© 2023 Linyi Li

CERTIFIABLY TRUSTWORTHY DEEP LEARNING SYSTEMS AT SCALE

BY

LINYI LI

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois Urbana-Champaign, 2023

Urbana, Illinois

Doctoral Committee:

Assistant Professor Bo Li, Chair
Professor Tao Xie, Co-Chair
Professor Carl A. Gunter,
Associate Professor J. Zico Kolter, Carnegie Mellon University

ABSTRACT

Great advances in deep learning (DL) have led to state-of-the-art performance on a wide range of challenging tasks. However, along with the rapid deployment of DL systems, several trustworthy threats arise, such as weak robustness against stealthy noise perturbations and natural transformations, bias across different subgroups, and lack of numerical reliability. These trustworthy threats have raised great concerns, especially when deploying DL systems in safety-critical scenarios such as autonomous driving and facial recognition for safeguarding. To defend against these common trustworthy threats, this thesis systematically proposes or enhances certification approaches and certified training approaches for DL systems, especially for large-scale DL systems.

A certification approach can guarantee some properties of the DL system under some trustworthiness properties. For instance, the robustness certification approach can guarantee the worst-case test accuracy when the attacker imposes any input perturbations or transformations within some bounded range. A certified training approach can improve the DL system's guaranteed trustworthiness under a certain property by training the DL model, e.g., improving the guaranteed test accuracy above.

This thesis begins with a systematic taxonomy of certification and certified training approaches. Then for several critical trustworthiness properties, this thesis proposes the corresponding certification and certified training approaches that lead to state-of-the-art tightness and scalability. These approaches are motivated by a few core principles, including dual problem analysis for randomized smoothing, general cutting planes for bound propagation, stratified sampling, subpopulation decomposition, and abstract interpretation. The effectiveness of the proposed approaches is supported by both theoretical analyses and empirical evaluations. The thesis is concluded with a discussion of limitations, challenges, and future directions towards achieving fully certifiable, reliable, and scalable machine learning.

In summary, this thesis enables certification of various trustworthy properties for DL systems up to millions of parameters, representing a major step in certified deep learning, an important research topic in machine learning, computer security, and software engineering.

To my parents, for their love and support.

ACKNOWLEDGMENTS

I would like to express my greatest gratitude to Prof. Bo Li, my advisor, and Prof. Tao Xie, my co-advisor. Before entering the PhD program, I had little experience in computer science research and was completely new to the field of trustworthy machine learning. It is my two advisors who admitted me into the program, taught me how to conduct research with unimaginable patience, enriched my immature research ideas and write-ups into high-quality publications in prestigious venues, and unconditionally supported and is supporting my career. This five-year journey was never easy, enriched by many frustrations from hard theoretical challenges, negative experimental results, dozens of submission rejections, and several rounds of revisions. Hence, only advisors can play a deterministic role to turn this tough journey into a fruitful, memorable, and enjoyable one, and they actually did. Inspired by their diligence and love of research, I plan to devote myself to academic research after graduation and follow their role models to help more students to flourish. All in one word, it is my lifetime treasure to have them as my advisors.

I am greatly thankful to other committee members: Prof. Carl A. Gunter and Prof. J. Zico Kolter. Carl's research is a textbook case of good computer security and privacy research which shaped my research taste. A visit to Zico's office in 2018 planted my interest in neural network robustness certification, and that seed has now grown into this thesis. As domain experts in this field, their insightful comments and questions inspired me to think about my research at a higher level and largely led to the discussion part of this thesis.

I sincerely thank Prof. Ce Zhang, for the pleasant collaborations that have led to several impact research works. I appreciate invaluable feedback from Prof. Varun Chandrasekaran, Prof. Robin Kravets, Prof. Darko Marinov, Prof. Ruta Mehta, Prof. Sasa Milosevic, Prof. Madhusudan Parthasarathy, and Prof. Gagandeep Singh that greatly improves the thesis presentation.

Besides research at school, summer industry internships are also an important part of my Ph.D. journey. Special thanks go to my research mentors: Adam Kalai, Mukul Prasad, Ripon Saha, and Neel Sundaresan.

The thesis would not be possible without collaborations with brilliant researchers and peers: Bhaskar Ray Chaudhury, Pin-Yu Chen, Wenda Chu, Hanjiang Hu, Junhao Hu, Mintong Kang, Huichen Li, Zhangheng Li, Xiangyu Qi, Wenyu Wang, Maurice Weber, Fan Wu, Chulin Xie, Xiaojun Xu, Zhuolin Yang, Huan Zhang, Chenhui Zhang, Jiawei Zhang, Zexuan Zhong, Yuhao Zhang, and many others. Thank you!

I am very fortunate to have lots of friends, who have enriched my life, especially during the tough Covid period: Shuyuan Chen, Yuting Du, Yuchen Fan, Zhongyu Jiang, Kaixiang Lei, Yunqi Li, Jacob Laurel, Jiaxin Shi, Yufeng Su, Wei Wang, Mengdi Xu, Kaiyuan Zhang, Xuan Zhang, Zirui Zhao, ... (list is never complete). I express my sincere gratitude to all of you.

Last and most importantly, thank you, my father Guihong Li and mother Jinbi Tang, for your invaluable love and support all the time. This thesis is dedicated to both of you.

TABLE OF CONTENTS

Part I

Introduction and Overview

In the past decade, we have witnessed deep learning and AI revolution in almost every aspect of our life. Deep learning (DL) has reshaped many fields such as image recognition[142, 191, 360] and generation [312, 349], language understanding and reasoning [44, 96, 378], program synthesis [60, 230] and software testing [222], and decision-making [265, 338].

Despite such success, the wide deployment of DL has exposed serious safety and social threats due to their lack of trustworthiness. For example, an adversary can rotate the camera [110, 124, 130, 416] or attach tiny stickers on road signs [112, 464] to fool DL-based autonomous driving systems, possibly leading to fatal traffic accidents. Such an adversary has been made practical to attack commercial DL systems [42, 219, 220, 449]. As noted in the recent *AI bill of rights* from the White House [279], such lack of trustworthiness in current DL-based tools threatens the rights of the public and challenges democracy.

Existing literature mainly evaluates the trustworthiness via detection approaches (i.e., "attacks" in computer security literature). These approaches find trustworthy vulnerabilities by searching trustworthiness-violating examples, e.g., specific rotation angles that fool the DL system[1] into making wrong predictions. Then, some empirical defense approaches enhance the DL system so that detection approaches cannot find trustworthiness issues. However, when a detection approach fails to find trustworthy vulnerabilities, it could be because the detection approach is not powerful enough, rather than the DL system is truly trustworthy. Indeed, many empirically defended DL systems are shown not trustworthy by stronger detection approaches [15, 371], which brings "a false sense of trustworthiness."

In contrast to detection approaches, **certification approaches** evaluate the trustworthiness of a DL system by computing a theoretical guarantee under certain trustworthiness properties and certain constraints; and the corresponding certified training approaches train DL system to improve such guarantees. As we can see, certified approaches, including certification and certified training approaches, provide a more reliable evaluation of DL trustworthiness in the form of a rigorous guarantee, and they are in urgent demand, especially in safety-critical applications such as autonomous driving, facial recognition, malware detection, and cyber-physical controllers. Figure 1.1 illustrates the relation between empirical and certified approaches, where the certified approach is the focus of this thesis.

[1] In this thesis, DL system refers to any system containing Deep Neural Networks (DNNs) as the machine learning (ML) models.

Certified approaches are usually tailored and optimized for the given trustworthiness property. In this thesis, we will propose novel certified approaches for the following commonly studied and widely concerned trustworthiness properties: robustness (robustness against ℓ_p-bounded perturbations, robustness against semantic transformations, robustness against state observation perturbations in reinforcement learning, robustness against training data perturbations in reinforcement learning), distributional fairness, and numerical reliability. For **all** these trustworthiness properties, we enable the construction and effective certification with non-trivial trustworthiness for DL systems with millions of parameters, significantly expanding the boundary of certified approaches. For example, for robustness against ℓ_p-bounded perturbations, our white-box approach is integrated into α-β-CROWN, the state-of-the-art tool that wins latest neural network verification competitions [20, 272], and our black-box approach is the first known approach that refutes the common belief ("black-box certification is intrinsically loose") and establishes the theoretical foundation for applying black-box certification to high-dimensional data domain.; for robustness against semantic transformations, we achieve 70.0% certified accuracy against any brightness change within $\pm$40% on the challenging ImageNet dataset; for numerical reliability, we generate certifications for over 70 diverse DNN architectures within 3 s running time per case, and fix the numerical defects in all these architectures within 55 s in total.

The rest of the thesis is organized as follows:

- Part I (Chapter 1) introduces background, terminologies, and taxonomy of certified approaches for DL and contextualizes this thesis.
- Part II (Chapters 2 to 6) is dedicated to certified approaches for robustness against ℓ_p-bounded perturbations. This property is widely studied in the literature, and this thesis pushes the boundary for two main branches of certified approaches: black-box and white-box approaches, from both certification and certified training sides.
- Part III (Chapters 7 to 10) presents two categories of generalizations of certified approaches for robustness beyond ℓ_p-bounded perturbations: certification against semantic transformations and robustness certification in reinforcement learning.
- Part IV (Chapters 11 to 13) further generalizes certified approaches to more trustworthiness properties beyond robustness: fairness and numerical reliability.
- Part V (Chapter 14) discusses limitations, challenging, and future directions towards achieving fully certifiable, reliable, and scalable machine learning.

Figure 1.2 illustrates the overall structure of the thesis.

CHAPTER 1: PRELIMINARIES AND OVERVIEW

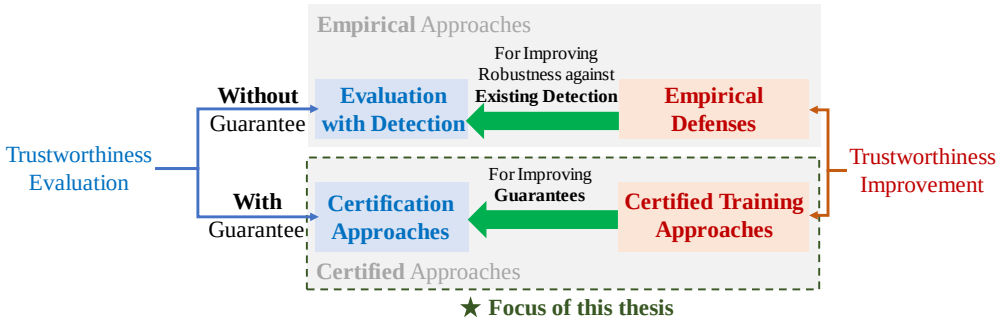


Figure 1.1: Relation between empirical approaches and certified approaches for trustworthy deep learning. The certified approach, including certification and certified training approach, is the focus of this thesis.

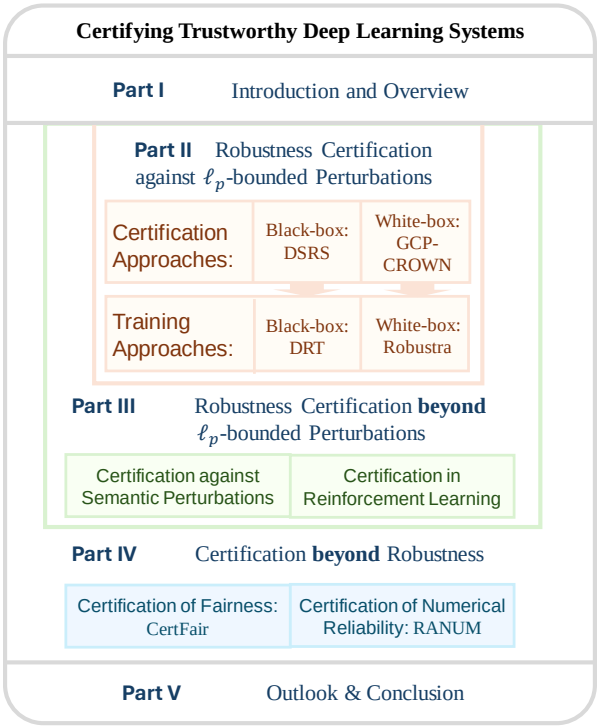


Figure 1.2: Thesis structure.

This chapter provides a brief introduction of key concepts in certifiably trustworthy deep learning and presents a taxonomy and characterization of certified approaches. Based on the taxonomy, we contextualize the approaches to present in this thesis.

1.1 PRELIMINARIES

In this thesis, unless otherwise specified, we use normal font (e.g., a, b, x, y) to represent normal objects and scalars, bolded font (e.g., $\boldsymbol{x}$) to represent scalars and scalar functions, bolded uppercase font (e.g., $\boldsymbol{M}$) to represent matrices, regular font (e.g., $\mathrm{x}, \mathbf{a}$) to represent random variables, and calligraphic font (e.g, $\mathcal{D}, \mathcal{X}, \mathcal{P}$) to represent sets (can also be represented by uppercase font) and distributions.

$\mathbb{R}$ is the set of real numbers, $\mathbb{R}_{\geqslant 0}$ is the set of non-negative real numbers, $\mathbb{R}^d$ is the d-dimensional vector space, $\mathbb{N}$ is the set of natural numbers, and $\mathbb{N}_+$ is the set of natural numbers without 0. We denote $[n]$ as set $\{1, 2, \ldots, n\}$. $\Pr[\cdot]$ refers to probability and $\mathbb{E}[\cdot]$ refers to expectation. $\|\cdot\|_p$ stands for the ℓ_p-norm of a vector ($p = 0$ or $p \geqslant 1$ or $p = \infty$). $\mathbb{I}[\cdot]$ is an indicator function, which equals to 1 if the predicate in the bracket holds otherwise 0. $[\cdot]_+$ and $[\cdot]_-$ stand for elementwise $\max\{\cdot, 0\}$ and $\min\{\cdot, 0\}$ respectively. ":=" or "=:" denotes to the definition of some quantity.

$\mathbb{1}_d$ is a d-dimentional all-one vector $(1, 1, \cdots, 1)^\intercal$. $\boldsymbol{I}_d$ is a $d \times d$ identity matrix. $\mathcal{N}(\boldsymbol{\mu}, \sigma^2 \boldsymbol{I}_d)$ refers to the d-dimensional multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\sigma^2 \boldsymbol{I}_d$. A special case is $\mathcal{N}(\mu, \sigma^2)$ which refers to single-dimensional normal distribution. $\Phi : \mathbb{R} \to (0, 1)$ is the cumulative density function (CDF) of standard normal distribution $\mathcal{N}(0, 1)$. $\mathrm{ReLU} : x \mapsto \max\{x, 0\}$ is the ReLU activation function, which expands to vector domain $\mathbb{R}^d \to \mathbb{R}^d_{\geqslant 0}$ naturally by applying the operation elementwise.

$f_{\boldsymbol{\theta}}(\cdot)$ or $F_{\boldsymbol{\theta}}(\cdot)$ usually refers to a deep neural network (DNN) model, where $\boldsymbol{\theta}$ denotes to all model parameters. When context is clear, $\boldsymbol{\theta}$ may be omitted. $f_{\boldsymbol{\theta}}$ and $F_{\boldsymbol{\theta}}$ can usually refer to the same model, where the image of $f_{\boldsymbol{\theta}}$ is usually a vector, indicating the predicted confidence for all classes; the image of $F_{\boldsymbol{\theta}}$ is usually a scalar value, indicating the predicted scalar value or class from $f_{\boldsymbol{\theta}}$.

Certified approaches are divided into two categories: certification approaches and certified training approaches, as illustrated in Figure 1.1.

Certification Approaches

A *certification approach* provides a guarantee of some specific trustworthiness property for the DL system in the form of whether some *predicate* always holds for all possible states within the underlying set, where the underlying set is given by the *threat model*. As we can see, there are two requisites to define a certification approach: threat model and property predicate.

Definition 1.1 (Threat Model)**.** A threat model $\mathcal{R} : \mathbb{R}_{\geqslant 0} \to 2^{\mathcal{S}}$ is a set of all possible states parameterized by a radius parameter $r \in \mathbb{R}_{\geqslant 0}$ satisfying:

(*1*) Set monotonicity with respect to r: $\forall r_1 \leqslant r_2$, $\mathcal{R}(r_1) \subseteq \mathcal{R}(r_2)$.

(*2*) State validness: Given a radius r, the threat model generates set of states $\mathcal{R}(r)$, such that any state within which $s \in \mathcal{R}(r)$ can determine the DL system's output.

The concrete definition of threat model $\mathcal{R}$ is determined by the trustworthiness property. From the perspective of computer security, the definition formalizes the exhaustive power of the underlying attacker that can use any state within $\mathcal{R}(r)$ as the input to undermine the DL system.

As an example, the property of robustness against ℓ_p-bounded perturbation leads to the threat model of ℓ_p-bounded adversary.

Example 1.1 (ℓ_p-bounded Adversary). For a given benign input $(\boldsymbol{x}_0, y_0)$, where $\boldsymbol{x}_0 \in \mathcal{X}$ is the input instance and $y_0 \in [C]$ is its true label, the ℓ_p-bounded adversary defines a state set of possible input: $\mathcal{R}(r) = B_{p,r}(\boldsymbol{x}_0) := \{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{x}_0\|_p \leqslant r\}$.

In the threat model of ℓ_p-bounded adversary, the attacker can add arbitrary perturbations to the input as long as the ℓ_p norm of the perturbation is within r. As one of the most widely studied trustworthiness properties, we will go back to this several times and introduce our certified approaches for this property proprietary in Part II.

Other threat model examples can be semantic adversary that exerts domain-specific semantic-preserving transformations to input data, data poisoning adversary that perturbs the training data, state observation adversary that alters the agent's observed states in reinforcement learning, distributional shifting, and more. We will define them respectively in corresponding chapters.

Another requisite is the property predicate.

Definition 1.2 (Property Predicate)**.** A property predicate $p(f, s)$ takes a DL system f and a state s as an input, and judges a binary result, where `True` indicates the property holds and `False` otherwise.

For example, for the robustness property, making a correct prediction for the perturbed or transformed input s is the goal and naturally defines the predicate. In many cases, the predicate's result is based on comparing some metric with a threshold (e.g., accuracy larger than some value, or mean loss smaller than some value, or confidence margin being positive for classification correctness). Hence, the goal of a certification approach naturally transforms into providing a lower or upper bound of that metric.

Now we are ready to define the certification approach.

Definition 1.3 (Certification Approach)**.** Given a DL system $f : \mathcal{S} \to \mathcal{Y}$, where $\mathcal{S}$ is the valid state set and $\mathcal{Y}$ is model output space, a threat model $\mathcal{R}$ defined in Definition 1.1, a radius parameter $r \in \mathbb{R}_{\geqslant 0}$, a predicate p defined in Definition 1.2, an algorithm $\mathcal{A}(f, \mathcal{R}, p, r)$ is called a *certification approach* if it satisfies this condition:

If $\mathcal{A}(f, \mathcal{R}, p, r) = \texttt{True}$, $\forall s \in \mathcal{R}(r), p(f, s)$ (when $\mathcal{A}$ is deterministic) or $\Pr[\forall s \in \mathcal{R}(r), p(f, s)] \geqslant 1 - \alpha$ where α is a pre-defined small threshold (when $\mathcal{A}$ is probabilistic, and the randomness should be independent of f, $\mathcal{R}$, p, and r).

As we can see, when the certification approach outputs `True`, the trustworthiness property holds for any state within the threat model set. Since the attacker is constrained to choose the state from the set, such output is a guarantee of the system's trustworthiness under an attack or under environment uncertainties, so $\mathcal{A}$ is called a certification approach.

Take the robustness certification for example, when the certification approach outputs `True`, any constrained perturbation to input $s \in \mathcal{R}$ cannot change the system's correct prediction, so the attacker cannot succeed. Usually, we measure a DL system's certified robustness by the ratio of test set samples where $\mathcal{A}$ outputs `True`, and this ratio, namely, certified robust accuracy, indicates a lower bound of system's accuracy under the constrained attack.

What happens if a certification approach outputs `False`? From the definition, the trustworthiness is unknown, i.e., $\forall s \in \mathbb{R}(r), p(f, s)$ could either hold or not. So a naive certification approach may barely output `False` which is useless. On the other hand, we hope that when the certification approach outputs `False`, the trustworthiness property does not hold, i.e., when the trustworthiness property holds, the approach outputs `True` as much as possible. We call this expectation *tightness*: if the approach is tighter, the approach outputs `True` as much as possible when the trustworthiness property holds. If an approach achieves perfect tightness, it implies that when the trustworthiness property holds, it will always output `True`, then we will call the approach *complete certification*. We will discuss this more in Section 1.3.1.

Certified Training Approaches

Though it sounds promising to use certification approaches to bring trustworthiness guarantees to DL systems, it is challenging. On the one hand, the DL system to certify could be intrinsically untrustworthy, so it is impossible to generate guarantees for it. On the

other hand, the DL system could be hard to certify due to its certification-unfriendly weight assignment or architecture design.

To mitigate these challenges, certified training approaches are developed both in the literature and in this thesis. Certified training approaches train DNNs so they can be certified with a high degree of trustworthiness. Certified training approaches significantly boost the certified trustworthiness by jointly improving the intrinsic trustworthiness and adapting the model architecture and weights in a certification-friendly direction.

Certified training approaches are usually strongly tied to certification approaches. They encourage desired properties from the certification approaches for the model to emerge during training. For the approaches to propose in this thesis, we will introduce them alongside the corresponding certification approaches due to this strong tie. In Section 1.3.2, we will categorize certified training approaches in detail.

Since certified training approaches are executed on the training dataset, and the degree of trustworthiness is measured by running certification approaches on the test dataset, a strong certified training approach under common notions also has a good degree of generalizability, though not explicitly emphasized.

1.2 TRUSTWORTHY PROPERTIES

In this section, we provide a brief description of trustworthy properties considered in this thesis, along with their corresponding definitions of threat models and property predicates. Detailed motivations and illustrations of these trustworthy properties are discussed in detail in the following chapters.

1.2.1 Robustness against ℓ_p-Bounded Perturbations

DL systems should be robust against arbitrary tiny perturbations directly added to normal input. For this trustworthy property, the threat model is defined with ℓ_p-bounded adversary (Example 1.1), and the property predicate depends on the task: for the classification task, the predicate is making a correct prediction, i.e., for input $\boldsymbol{x}_0$ with true label $y \in [C]$, the output for the perturbed input $F(\boldsymbol{x}) = y$; for the regression task, the predicate is the prediction error is upper bounded by some threshold, i.e., for input $\boldsymbol{x}_0$ with true value $y \in \mathbb{R}$, the output for the perturbed input satisfies $|F(\boldsymbol{x}) - y| \leqslant \epsilon$.

For the regression task, this property transforms into providing a lower and upper bound of $F(\boldsymbol{x})$ for any $\boldsymbol{x}$ in state set. For the classification task, since DL system F usually conducts label prediction by predicting a confidence score for each class and then choosing the most

probable class, i.e., $F(\boldsymbol{x}) = \arg\max_{i\in[C]} f(\boldsymbol{x})_i$ where $f(\boldsymbol{x})_i$ is the confidence score for the i-th class, the predicate equals to for any $i \in [C]\backslash\{y\}$, $f(\boldsymbol{x})_y - f(\boldsymbol{x})_i \geqslant 0$. Hence, computing a lower bound of $f(\boldsymbol{x})_y - f(\boldsymbol{x})_i$ for any $\boldsymbol{x}$ in state set is the goal. In summary, in both cases, the certification approach can generate the certification via computing a lower or upper bound for some quantity of interest.

When the robustness is certified for threat model $\mathcal{R}(r)$, we call r certified radius or robust radius (of model F at point $\boldsymbol{x}_0$). We study certification for this property in Part II.

1.2.2 Robustness against Semantic Transformations

DL systems should be robust against semantic-preserving transformations, such as small brightness change, contrast change, rotation, and scaling, to the image. For this trustworthy property, the threat model is defined with a predefined parameterized transformation function $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$, transforming an image $x \in \mathcal{X}$ with a $\mathcal{Z}$-valued parameter α. For example, we use $\phi_R(x, \alpha)$ to model a rotation of the image x by α degrees counter-clockwise with bilinear interpolation. Then, the threat model is defined as the range of transformation function when the parameter is bounded, e.g., $\mathcal{R}(r) = \{\phi(\boldsymbol{x}_0, \boldsymbol{\delta}) : \|\boldsymbol{\delta}\|_\infty \leqslant r\}$. The property predicate definition is the same as Section 1.2.1.

We study certification for this property in Chapter 7.

1.2.3 Robustness against State Observations in Reinforcement Learning

In reinforcement learning (RL), the DL system encodes a trained policy π to interact with the environment in multiple rounds (namely, "steps"). At each step, the policy observes the state input from the state space $s \in \mathcal{S}$ and chooses an action from the action space $a = \pi(s) \in \mathcal{A}$. Such action transforms the state to $P(s, a) \in \mathcal{S}$ (consider deterministic environment in this thesis) in the next round and receives a reward $r(s, a)$. However, the observed state by the policy agent can be maliciously perturbed by the attacker, e.g., by hacking the camerate perceptron module, and we hope the policy is still robust in this case.

Hence, we define the threat model to be the union of ℓ_p-bounded adversary at each step, i.e., denoting s_t to the true state at each step t, the state set $\mathcal{R}(r) = (S'_0, S'_1, \ldots, S'_H)$ where $S'_t = B_{p,r}(s_t)$. The property predicate is defined at two levels: step-level — per-state action stability, and episode-level — guaranteed cumulative reward. The former requires the action prediction does not change, i.e., $\pi(s'_t) = \pi(s_t)$ for $s'_t \in S'_t$. The latter requires the cumulative reward after H steps of interaction, i.e., $R = \min \sum_{i=0}^{H} r(s_t, \pi(s'_t))$, $s'_t \in S'_t$ is larger than some threshold.

We study certification for this property in Chapter 8.

1.2.4 Robustness against Poisoning Attacks in Reinforcement Learning

Besides state observation perturbations, in offline reinforcement learning, the performance of trained DL policy can also be negatively impacted by perturbations on the training data, i.e., the malicious attacker, as a training data provider, can destroy the policy performance by deleting, inserting, or replacing the training data samples.

Concretely, in offline RL, a training dataset $D = \{\tau_i\}_{i=1}^{N}$ consists of logged trajectories, where each trajectory $\tau = \{(s_j, r_j, a_j, s'_j)\}_{j=1}^{l} \in (\mathcal{S} \times \mathcal{A} \times \mathbb{R} \times \mathcal{S})^l$ consists of multiple tuples denoting the transitions (i.e., starting from state s_j, taking the action a_j, receiving reward r_j, and transitioning to the next state s'_j). Training dataset D can be poisoned in the following manner. For each trajectory $\tau \in D$, the adversary is allowed to replace it with an arbitrary trajectory $\widetilde{\tau}$, generating a manipulated dataset $\widetilde{D}$. We denote $D \ominus \widetilde{D} = (D \backslash \widetilde{D}) \bigcup (\widetilde{D} \backslash D)$ as the *symmetric difference* between two datasets D and $\widetilde{D}$. For instance, adding or removing one trajectory causes a symmetric difference of magnitude 1, while replacing one trajectory with a new one leads to a symmetric difference of magnitude 2. Hence, the threat model defines a set over manipulated datasets: $\mathcal{R}(r) = \{\widetilde{D} : D \ominus \widetilde{D} \leqslant r\}$.

The property predicates are the same as Section 1.2.3. We study certification for this property in Chapter 9.

1.2.5 Distributional Fairness

DL systems may inherit bias and disparity. Specifically, for data from different groups (identified by some protected or sensitive attribute $\mathcal{X}_s$), the model may have different performances. To identify the fairness issue, we expect that the model should have a low loss value on a perfectly fair distribution. In other words, when the distribution itself is fair, the model should encode such distribution very well. Otherwise, the model can have bias (if performed well on training distribution) or have bad performance (if not performed well either on training distribution). Either is not expected.

Hence, to quantify the fairness of a model from the distributional level, we propose to certify an upper bound of expected loss on any slightly shifted and perfectly fair distribution. The threat model is the set of all distributions (1) whose distance to the original training distribution is bounded; and (2) which is perfectly fair measured by group base rate. The property predicate is the model's expected loss on the distribution is smaller than some threshold.

A formal definition and our certification approach for this property is in Chapter 11.

1.2.6 Numerical Reliability

In deployment, a DL system may incur numerical failures by outputting `NaN` or `INF` instead of producing any meaningful prediction, resulting in system crashes. This is the numerical reliability issue for DL systems.

Numerical failures are triggered when some operators in the DL system receive invalid input, e.g., when log operator receives negative input. Hence, to certify the numerical reliability of a DL system, the threat model is defined as all valid input and all valid weights. For example, for image models, the threat model is the valid image domain $[0, 1]^d$ concatenated with the typical weight domain. Note that this threat model is not parameterized by some radius r, or it can be viewed as parameterized by a fixed radius r. Then, the property predicate is that for all operators within the DL system, their input falls into the valid range.

In contrast to previous trustworthy properties that certify a concrete DL system, certification of numerical reliability brings a guarantee for a group of DL systems sharing the same model architecture.

A formal definition and our certification approach is in Chapter 12.

1.3 TAXONOMY AND CHARACTERIZATION OF CERTIFIED APPROACHES

In this section, we outline important characteristics and features of certified approaches, and use them to construct a taxonomy for both certification and certified training approaches.

1.3.1 Taxonomy of Certification

For certification approaches, we typically care about five aspects: efficiency (or scalability), tightness, soundness, inference overhead, and generalizability. We will use them as the taxonomy criteria.

Efficiency (Scalability). The certification approach varies in its efficiency, and efficiency can usually be measured by time complexity with respect to model size (concretely, the number of neurons or parameters). An efficient certification approach can support larger models, so it is also called scalability. Given the trend of scaling up DL models, the more efficient a certification approach is, the broader its potential applicability. Hence, we use efficiency (scalability) as one taxonomy criterion.

We measure the efficiency with two metrics: (1) A qualitative measure: the largest dataset that has been demonstrated feasible to certify by existing work using the corresponding certification approach under a reasonable radius[2]. The dataset effectively measures scalability. For example, the approach scaling up to ImageNet is more scalable than the one to MNIST. (2) A quantitative measure: the best known time complexity for certifying an arbitrary input, given an arbitrary DNN with depth l, width w in terms of neurons, and sampling number S (for sampling-based probabilistic approaches and partition-based approaches). Note that the time complexity for DNN inference is $O(lw^2)$. All sampling-based probabilistic approaches have complexity $O(Slw^2)$, which is because the sampling time cost is much higher than the actual bound computation whose time complexity is subsumed. $\text{poly}(l, w)$ means a time complexity higher than $O(lw^3)$.

Tightness. As discussed in Section 1.1, besides being sound (when output `True`, truly trustworthy), we also expect a certification approach to be as tight as possible (when output `False`, not trustworthy as much as possible). Hence, tightness is one important criterion.

The highest level of tightness is "complete", where a `False` output means not trustworthy for certain. For other incomplete approaches, the tightness measurement is measured quantitatively by comparison with other approaches supporting the same architecture and inference protocol. The comparison is based on our benchmark results in [225] and theoretical results from the literature. For approaches that support generic DNNs, tightness are ranked by T_n; and for sampling-based probabilistic approaches, they are ranked by ST_n. The larger n means tighter approaches.

The *intrinsic trade-off between efficiency and tightness*, i.e., either scalability or tightness can be achieved but not both, constitutes the main obstacle for trustworthiness certification. For example, for robustness certification, all complete certification approaches have exponential time complexity $O(2^{lw})$ which is theoretically proved [180, 402].

Soundness. Though all approaches are sound by definition, the degree of soundness still varies: some approaches bring deterministic certification and others tolerate a small probability of making false claims. We use "deterministic/probabilistic" to distinguish these two classes in our taxonomy.

Inference Overhead. Some certification approaches do not support DNNs with normal inference procedures. Instead, they require a customized inference procedure, e.g., adding noise to the input and then applying majority voting. By doing so, they can require less

[2]For example, radius $r \geqslant 1/255$ for robustness certification against ℓ_p-bounded perturbations.

information from the DL system itself, e.g., require no information about model's architecture, to improve the efficiency. On the other hand, the customized inference procedure could be much more expensive than the normal one, bringing inference overhead which is critical for deployment. Hence, we outline the inference overhead as one taxonomy criterion and highlight approaches that incur inference overhead.

Generalizability. Different certification approaches require different degree knowledge from the DL system. As mentioned above, some approaches require no information about model's architecture but just an oracle access to the inference result. We call these approaches *black-box approaches*. On the other hand, other approaches require and support a particular set of model architectures, and they are called *white-box approaches*. Most common white-box approaches support models with ReLU as the activation functions (named "ReLU Nets" in Table 1.1). Some are more generic (named "generic DNNs" in Table 1.1), and some others are more restrictive, e.g., only supporting specific Lipschitz-bounded layers.

An interesting observation is that some most generalizable approaches (i.e., black-box approaches) have inference overhead, indicating a trade-off between efficiency and generalizability.

Table 1.1 presents our taxonomy results of certification approaches. Besides categorizing using the above five aspects, we use the certified trustworthy property as the first-level criterion since approaches are proposed to certify one trustworthy property at a time. We group approaches with the same core methodology together and use the methodology name as the identifier. Detail references are listed in the last column.

As we can observe, there is a large family of certification approaches in the literature and most of them are proposed within five years, reflecting the rapid development of this field. In Section 1.4, we will describe our approaches to propose using these taxonomy criteria and summarize how they advance the boundary of certification approaches.

More discussions and findings from the taxonomy can be found in [225]. A visualization of the taxonomy is in `https://sokcertifiedrobustness.github.io`.

Table 1.1: Taxonomy, characteristics, and references of trustworthy certification approaches. Details are explained in Section 1.3.1. Relative tightness levels T_n and ST_n are only listed among comparable approaches.

Trustworthy Property	Complete/ Incomplete	Soundness	Generalizability		Core Methodology			Scalability (Scale up to)	Scalability (Complexity)	Tightness	Inference Overhead	References
Robustness against ℓ_p-bounded Perturbations	Complete	Deterministic	White-box	for ReLU Nets	Solver-Based	SMT-Based		MNIST	$O(2^{lw})$	Complete		[298, 299]
						MIP-Based		CIFAR-10	$O(2^{lw})$	Complete		[66, 102, 240, 368]
					Extended Simplex Method			MNIST	$O(2^{lw})$	Complete		[180, 181]
					Branch-and-Bound			CIFAR-10	$O(2^{lw})$	Complete		[19, 48, 49, 91, 108, 115, 122, 123, 387] [175, 245, 388, 389, 425, 448]
	Incomplete	Deterministic	White-box	for General DNNs[1]	Linear Relaxation	Linear Programming (LP)		CIFAR-10	$O(\text{poly}(l,w))$	T_5[2]		[318, 402]
						Linear Inequality	Interval	Tiny ImageNet	$O(lw^2)$	T_2		[134]
							Polyhedra	Tiny ImageNet	$O(lw^3)$	T_3[2]		[246, 247, 343, 402, 424, 443]
							Zonotope	Tiny ImageNet	$O(lw^3)$	T_3[2]		[8, 262, 340, 341]
							Duality	Tiny ImageNet	$O(lw^3)$	T_4[2]		[103, 104, 406, 407]
						Multi-Neuron Relaxation		CIFAR-10	$O(lw^3)$ - $O(2^{lw})$[6]	T_7		[273, 286, 342, 367]
					Semidefinite Programming (SDP)			CIFAR-10	$O(\text{poly}(l,w))$	T_6		[89, 105, 114, 307, 308]
				for Lip-Bounded Nets	Lipschitz	General Lipschitz		Tiny ImageNet	$O(lw^2)$	T_1[3]		[206, 209, 345, 359, 374, 402, 445]
						Smooth Layers		Tiny ImageNet	$O(lw^2)$	[3]		[143, 227, 346, 373, 438, 439, 441]
				for Non-ReLU Nets[4]	Curvature			CIFAR-10	$O(lw^3)$	[4]		[344]
				Black-box[5]	Zeroth Order Smoothing	Lipschitz		ImageNet	$O(Slw^2)$	[5]	Exist	[214]
		Probabilistic	Black-box		Zeroth Order Smoothing	Differential Privacy Inspired		ImageNet	$O(Slw^2)$	ST_1	Exist	[204]
						Divergence Based		ImageNet	$O(Slw^2)$	ST_2	Exist	[106, 218]
						Neyman Pearson		ImageNet	$O(Slw^2)$	ST_3	Exist	[77]
						Level-Set Analysis		ImageNet	$O(Slw^2)$	ST_3	Exist	[364, 427, 442]
						Lipschitz		ImageNet	$O(Slw^2)$	ST_3	Exist	[16, 317]
					First Order Smoothing			ImageNet	$O(Slw^2)$	ST_4	Exist	[215, 267]
					Double Smoothing			ImageNet	$O(Slw^2)$	ST_5	Exist	[224]
Robustness against Semantic Transformations	Incomplete	Deterministic	White-box	for General DNNs[1]	Linear Relaxation	Linear Inequality	Interval	CIFAR-10	$O(lw^2)$	T_1		[343]
							Polyhedra	CIFAR-10	$O(lw^3)$	T_2		[22, 268]
	Complete			Black-box[7]	Parititon Enumeration			ImageNet	$O(Slw^2)$	Complete		[292]
	Incomplete	Probabilistic		Black-box	Partition Enumeration + Zeroth Order Smoothing			ImageNet	$O(S^2lw^2)$		Exist	[118, 223]
Robustness in RL aginst State Perturbations	Incomplete	Deterministic	White-box	for General DNNs[1]	Linear Relaxation	Liner Inequality	Polyhedra	CIFAR-10	$O(lw^3)$			[111]
		Probabilistic		Black-box	Zeroth Order Smoothing			CIFAR-10	$O(Slw^2)$		Exist	[196, 408]
Robustess in RL against Poisoning Attacks	Incomplete	Probabilistic		Black-box	Zeroth Order Smoothing			ImageNet	$O(Slw^2)$		Exist	[409]
Distributional Fairness	Incomplete	Probabilistic	White-box	for Wasserstein Distance	Lipschitz & Robust Optimization			MNIST	$O(lw^2)$			[347]
			Black-box	for Hellinger Distance	Gramian Bound			ImageNet	$O(Slw^2)$	ST_1		[399]
					Gramian Bound + Subpopulation Decomposition			ImageNet	$O(Slw^2)$	ST_2		[177]
Numerical Reliability	Incomplete	Deterministic	White-box	for General DNNs	Linear Relxation	Linear Programming (LP)		ImageNet	$O(\text{poly}(l,w))$	T_2		[460]
						Linear Inequality	Interval	ImageNet	$O(lw^2)$	T_1		[226]

1. Typical approaches mainly support ReLU networks, but extensions to general DNNs are available [40, 336, 443].
2. Tightness depends on intermediate layer bounds. If they share the same intermediate layer bounds, the tightness order is Zonotope < Polyhedra = Duality < LP [318].
3. Lipschitz bound is loose for typical DNNs, but can be tight for specially regularized DNNs which have small Lipschitz bounds.
4. Only available for networks whose activation functions have nonzero second-order derivatives, which exclude ReLU networks. Thus, tightness is incomparable with others.
5. The approach requires smoothing with some specific distributions as inference protocol.
6. Tunable time complexity dependent on the upper limit of number of linear constraints.
7. Only support discrete inputs and discrete transformations.

Table 1.2: Taxonomy and references of certified training approaches for trustworthy machine learning. *Suitable Certification* summarizes the certification approaches for which the training approach is designed. Details are explained in Section 1.3.2.

Robust Training Approaches	Suitable Verification	References
Regularization-Based	Complete, Incomplete and Deterministic (Lipschitz & Curvature)	[83, 84, 143, 206, 209, 344, 345, 346, 419, 438, 439]
Relaxation-Based	Incomplete and Deterministic (Linear Relaxation, SDP)	[21, 134, 221, 247, 262, 307, 337, 386, 406, 406, 446]
Augmentation-Based	Incomplete and Probabilistic	[16, 56, 77, 152, 204, 317, 427, 442]
Augmentation- and Regularization-Based	Incomplete and Probabilistic	[165, 166, 218, 430, 437]

1.3.2 Taxonomy of Certified Training

Compared to certification approaches, certified training approaches are relatively simpler, all sharing the same procedure in deep learning: gradient-descent-based optimization over training data in mini-batches. The main difference lies in the core methodologies in terms of data augmentation, pretraining, loss computation, and regularization. Hence, we use the core methodology as the taxonomy criterion. The taxonomy results are shown in Table 1.2.

Regularization-based Training. For complete certification, branch-and-bound (BaB) and mixed integer programming (MIP) are among the most efficient methodologies so far. Xiao et al. [419] find that for these certification approaches, the number of branches is upper bounded by the number of unstable ReLU neurons which motivates a regularization term to increase the ReLU neuron's stability for training. For complete certification based on linear region traversal, we can train with a regularization term maximizing the margin to non-robust regions [83, 85]. The Lipschitz and curvature certification favor small Lipschitz constant and small curvature bounds respectively. Therefore, the corresponding robust training approaches are very effective by explicitly penalizing large Lipschitz or curvature bounds [206, 209, 344, 374].

Relaxation-based Training. For linear relaxation based certification approaches, models with tight linear relaxation bounds are favored. To train such models, corresponding robust training approaches usually use the computed bounds from linear relaxation as the training objective to explicitly improve the bound tightness. This idea is similar to the powerful empirical defense named adversarial training (AT) [251] which uses effective attacks to approximately find "most adversarial" example $\max_{\boldsymbol{x}\in B_{p,r}(\boldsymbol{x}_0)} \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}), y_0)$ and minimize model weights $\boldsymbol{\theta}$ with respect to it, where $\mathcal{L}$ is a typical loss function such as cross-entropy loss. In relaxation-based training, instead, we compute an upper bound of $\max_{\boldsymbol{x}\in B_{p,r}(\boldsymbol{x}_0)} \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}), y_0)$ and minimize it. The bound can be derived from IBP [134, 337], polyhedra-based [21, 246, 446], zonotope-based [262], or duality-based certification [103, 221, 406]. Some useful training tricks are: combining relaxation-based loss with standard loss to improve benign

accuracy [134, 386, 446], applying relaxation on some layers but not all to balance benign accuracy and certified robustness [21], specialized weight initialization and training scheduling [337], and using reference space to guide the relaxation [221]. An intriguing phenomenon of relaxation-based training is that tighter relaxation, when used as the training objective, may not lead to more certifiably robust models [176], while the loosest IBP relaxation can achieve almost the highest certified robustness. A conjecture is that tighter relaxation may lead to a less smooth loss landscape containing discontinuities or sensitive regions which poses challenges for gradient-based training [176, 207]. Theoretical understanding of relaxation-based training is still lacking. Note that solver based and branch-and-bound based complete certification usually use linear relaxations for bounding. Therefore, models trained with these relaxation-based training approaches can usually be efficiently certified by these complete certification approaches [368, 389].

Augmentation-based Training. Since smoothing-based certification favors models to perform well for noisy inputs, to obtain high certified robustness, we can train the DNNs with noisy inputs, resulting in augmentation-based training [77, 204, 218]. Built upon such augmentation-based training, later approaches combine augmentation with regularization terms to encourage the prediction stability/consistency when the input noise is added [165, 166, 437]. Strategic training regularization combined with augmentation and ensemble is effective and achieves the state-of-the-art certified robustness against ℓ_2 adversary [152, 430]. Adversarial training combined with augmentation [317], and training unlabeled data [56] are also shown effective. Recently, diffusion models [349], which intrinsically possess the denoising ability, are leveraged to build models for randomized smoothing [55, 418]. They achieve superior or competitive certified robustness compared to above methods though require large model size which results in large inference overhead.

1.3.3 Extensions beyond Taxonomy

Besides the literature and related work covered by the taxonomy, there are a few extensions of certified approaches going further beyond. Here we illustrate some extending angles and discuss some example approaches. For related work specific to certain approaches, we will discuss them alongside the approach in the following chapters.

Extensions on Trustworthy Properties. The techniques of certified approaches discussed in this thesis are extended to certify other trustworthy properties. (1) **Robustness against local evasion attacks**: In local evasion attacks, the adversary slightly perturbs

the in-distribution data to mislead the model. The ℓ_p-bounded perturbations and semantic transformations are both special types of local evasion attacks. Now we elaborate on some other effective local evasion attacks and their certified approaches. (a) *Generative model based adversary* uses generative models such as GAN [131] to generate input perturbation. Similar to certification against the semantic adversary, smoothing-based approaches and linear relaxation approaches can be extended to provide certification against this adversary [264, 405]. (b) ℓ_0 *adversary* picks a bounded number of pixels to arbitrarily change and *patch adversary* picks a region of pixels with a bounded area to arbitrarily change. To defend against ℓ_0 adversary, smoothing-based approaches can be deployed [168, 205, 212]. To defend against patch adversary, the core idea of smoothing-based approaches, prediction aggregation on several noisy inputs which are patched inputs here, is leveraged to develop customized certification and corresponding training approaches. Starting from direct prediction aggregation [211], some recent certified defenses exploit or design model architectures and inference procedures with self-aggregation property, such as DNNs with small localized receptive fields [413], importance-score-based pruning [138], vision transformers [320], and two-round patch-masking [414], to improve the efficiency and tightness of robustness certification. These approaches [138, 211, 320, 413, 414] can provide robustness guarantees on the large-scale ImageNet dataset. (2) **Distributional evasion attacks**: In distributional evasion attacks, the attacker shifts the whole test data distribution within some bounded distance to maximize the expected loss. This threat model can be used to characterize the out-of-distribution generalization ability of ML models [334]. The certification under this threat model is an upper bound of the expected loss, which can be derived from duality under Lipschitz and curvature assumptions [347] or from extensions of smoothing-based approaches [197, 399]. Note that our distributional fairness property is similar to robustness against distributional evasion attacks, but in distributional fairness property we further require the perturbed distribution is perfectly fair. (3) **Global evasion attacks**: global evasion attacks can perturb any valid input example to mislead the model, whereas local evasion attacks can only perturb in-distribution data. Thus, the robustness against global evasion attacks means that the robustness property holds for the whole input domain. An example of a robustness property is that for any high-confident prediction, small perturbations cannot change the predicted label [209]. In the security domain, Chen et al. [64] recently proposed several domain-specific robustness properties such as requiring all low-cost features to be robust. To certify these properties, they propose a specific solver-based certification to verify logic ensemble models, and then use the found adversarial example as an augmentation for certified training. The certification and certified training *for DNNs* against global evasion attacks can be a promising direction.

Extensions on Other ML Models. There are efforts on generalizing existing certified approaches for DL systems to deal with more types of machine learning models. For example: (1) Some approaches that are designed for ReLU networks, such as linear relaxation based approaches, have been extended to support general DNNs [343, 407, 443], recurrent networks [99, 187, 315], transformers [39, 336], and generative models [263]. The main methodology is to derive the corresponding linear bounds for activation functions or attention mechanisms in these system models. Some complete certification approaches, e.g., branch-and-bound based ones [389], also support general DNNs. Note that these complete certification approaches become incomplete when applied on general DNNs. (2) Certification approaches for Lipschitz-bounded networks and non-ReLU networks have not been generalized to other model types yet. (3) Smoothing-based approaches typically need access to only the final prediction label, so they are applicable to any classification model. However, the model must follow the corresponding smoothing-based inference protocol. (4) There are also certification approaches for decision trees [10, 59, 393], decision stumps [393], nearest prototype classifiers [380], and logic ensembles [64]. However, there is no certification and certified training approach that supports all these system models yet. This is because certification and robust training approaches need to exploit properties (piecewise linearity, Lipschitz bound, smoothness, etc) of specific system models to achieve certified trustworthiness.

Extensions for Concrete Applications. Beyond the classification task, the discussed methodologies, such as linear relaxation and smoothing-based approaches, have been extended to certify DL systems in many concrete applications. In natural language processing, extensions include certification for recurrent neural networks against embedding perturbations [99, 187, 315], word substitutions [171], and word transformations [432, 459, 461]. Extensions have also been studied for object detection [70], segmentation [120], and point cloud models [75, 120, 236] in computer vision, and speech recognition [119, 281].

1.4 SCOPE OF THIS THESIS

In this thesis, we will propose certified approaches for all trustworthiness properties defined in Section 1.2. In Table 1.3, we summarize all approaches to propose and their characteristics. The characteristics strictly follow definitions in Section 1.3 and the table adapts a similar structure as the transpose of Table 1.1 and Table 1.2, so readers can make a clear comparison with the literature and the preceding taxonomy. The table also lists chapters that introduce the approaches in detail and reference links to corresponding publications, so readers can locate them if they are interested in details. Some certified training approaches are naturally

derived from the certification approaches, so they are introduced together under the same approach name. Then, for some trustworthiness properties, we do not propose certified training approaches yet. For proposed certification approaches, they achieve either state-of-the-art tightness or scalability among comparable ones; for proposed training approaches, they achieve state-of-the-art certified trustworthiness among approaches using similar costs.

Now we briefly outline why the proposed approaches inherit these characteristics.

For robustness against ℓ_p-bounded perturbations, we propose certified solutions for both black-box and white-box settings. The black-box setting is more scalable (to ImageNet) but induces inference overhead due to smoothing; the white-box setting is less scalable but requires no change to the model or inference protocol. Note that the certification approach GCP-CROWN is a part of α-β-CROWN that integrates a series of work in this field, and we will focus only on the novel methodology part, GCP-CROWN, in this thesis.

For robustness against semantic transformations, we propose black-box solution certification since it is the only feasible way to scale up to ImageNet-scale datasets, whereas existing black-box certification approaches can only support CIFAR-10-scale datasets to our knowledge.

For robustness against RL observation perturbations, though our approach CROP is a black-box certification approach, the methodology of tree search can be easily extended to allow white-box certification with no inference overhead.

For robustness against offline-RL poisoning attacks, our certification is based on the idea of partition aggregation, which is naturally black-box but incurs inference overhead. There is no white-box certification against poisoning yet for DNNs to our knowledge.

For distributional fairness, the distributional nature implies that all non-trivial certification of statistics should evolve confidence intervals, so the certification is probabilistic. Our proposed certification incurs no inference overhead and is black-box, meaning its optimality in terms of inference efficiency and generalizability.

For numerical reliability, since we aim at detecting and fixing the defect in an existing system, we are not able to modify the DL system. Since the system is not altered, we incur no inference overhead. Moreover, the numerical defect exists at the operator level inside the DL model, so we need to look into model architecture details and the approach can only be white-box.

More details are illustrated in the corresponding chapters.

Table 1.3: Certified approaches to propose in this thesis and their characteristics.

Trustworthy Property		Robustness against					Distributional Fairness	Numerical Reliability
		ℓ_p-bounded Perturabtions		Semantic Transformations	RL Observation Perturbations	Offline-RL Poisoning Attacks		
Certification Approach	Name	DSRS	GCP-CROWN	TSS	CROP	COPA	CertFair	RANUM
	Completeness		✓					
	Soundness	Probabilistic	Deterministic	Probabilistic	Probabilistic	Probabilistic	Probabilistic	Deterministic
	Generalizability	Black-box	White-box Generic DNNs	Black-box	Black-box	Black-box	Black-box	White-box Generic DNNs
	Core Method	Double Smoothing	Branch-and-Bound + Linear Relaxation	Partition + Zeroth-Order Smoothing	Tree Search + Zeroth-Order Smoothing		Gramian Bound + Decomposition	Interval Relaxation
	Scalability (up to)	ImageNet	TinyImageNet	ImageNet	CIFAR-10	ImageNet	ImageNet	ImageNet
	Scalability (complexity)	$O(Slw^2)$	$O(2^{lw})$	$O(S^2lw^2)$	$O(Slw^2)$	$O(Slw^2)$	$O(Slw^2)$	$O(lw^2)$
	Inference Overhead	Exist		Exist	Exist	Exist		
	Chapter	Chapter 2	Chapter 3	Chapter 7	Chapter 8	Chapter 9	Chapter 11	Chapter 12
	Publication	[224] (ICML 2022)	[448] (NeurIPS 2022)	[223] (CCS 2021)	[408] (ICLR 2022)	[409] (ICLR 2022)	[177] (NeurIPS 2022)	[226] (ICSE 2023)
Certified Training Approach	Name	DRT	Robustra	(same name as above, resp.)				(same name as above)
	Core Method	Augmentation + Transferability	Relaxation + Reduction	Augmentation		/	/	Relaxation
	Chapter	Chapter 4	Chapter 5	(same chapter as above, resp.)				(same chapter as above)
	Publication	[431] (ICLR 2022)	[221] (IJCAI 2019)	(same publication as above, resp.)				(same publication as above)

Part II

Robustness Certification against ℓ_p-bounded Perturbations

Probably, one of the most well-known, widely-concerned, and studied trustworthy threat of DL system is its robustness against ℓ_p-bounded perturbations. Extensive work has shown such vulnerability in normal DL systems is wide-spread [112, 132, 304, 359, 384], and could be turned into actual attacks into commercial DL systems [219, 449]. Hence, in this chapter, we focus on achieving certified robustness against ℓ_p-bounded perturbations.

In Chapter 2, we propose a black-box certification approach, DSRS, with state-of-the-art tightness among black-box approaches. In Chapter 3, we propose general cutting plane bound propagation GCP-CROWN, which is an important part of state-of-the-art white-box verifier α-β-CROWN that wins latest neural network certification competitions [20, 272]. Then, the following two chapters introduce suitable training approaches for black-box and white-box certification respectively. Both approaches share the same principle of reducing adversarial transferability among models. The last chapter concludes this part with a brief discussion.

The formal definition of robustness against ℓ_p-bounded perturbations is in Section 1.2.1. In the definition, there is a degree of freedom in ℓ_p norm types. To the best of our knowledge, existing ℓ_p-bounded attacks almost always consider ℓ_1, ℓ_2, and ℓ_∞. The inputs generated by these adversaries are within some radius r to clean input $\boldsymbol{x}_0$ measured by ℓ_1 norm (i.e., Manhattan distance), ℓ_2 norm (i.e., Euclidean distance), and ℓ_∞ norm (i.e., maximum difference among all dimensions) respectively. We illustrate the region from which the attacker picks the perturbed input in Figure 2.1 in the next page. In this part, we focus on ℓ_2 and ℓ_∞ adversaries. For approaches for ℓ_1 adversary, readers can refer to [225].

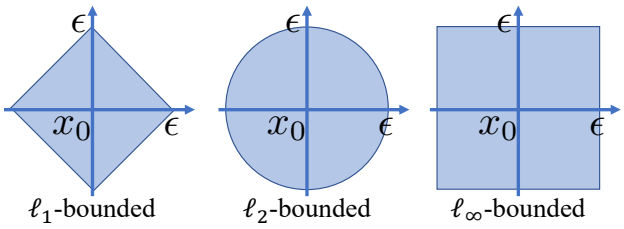


Figure 2.1: An ℓ_p-bounded adversary (Example 1.1) crafts perturbed input from ℓ_p-bounded region centered at clean input $\boldsymbol{x}_0$. From left to right are ℓ_1-, ℓ_2-, and ℓ_∞-bounded perturbation regions in 2D space with radius ϵ.

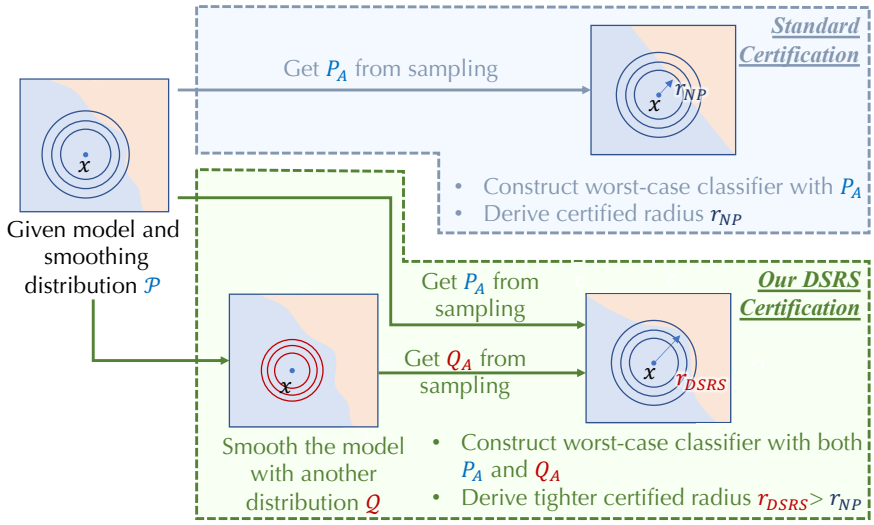


Figure 2.2: **Upper**: Standard certification for randomized smoothing leverages information from only one distribution (P_A) to compute robustness certification. **Lower**: DSRS leverages information from two distributions (P_A and Q_A) to compute certification for the smoothed classifier, yielding significantly larger certified radius.

Randomized smoothing [77, 218] has emerged as a popular technique to provide certified robustness for large-scale datasets. Concretely, it samples noise from a certain smoothing distribution to construct a smoothed classifier, and thus certifies the robust radius for the smoothed classifier. Compared to other techniques [134, 262, 406, 446], randomized smoothing is efficient and agnostic to the model as a black-box approach, and is applicable to a wide range of ML models, including large ResNet [142] on the ImageNet dataset.

To improve the certified robust radius, existing studies [77, 205, 218, 427] have explored different smoothing distributions. However, the improvement is limited. For example, ℓ_2

certified robust radius does not increase on large datasets despite that the input dimension d increases [77], resulting in a low ℓ_∞ certified radius on large datasets, theoretically shown as an intrinsic barrier of randomized smoothing ("curse of dimensionality" or "ℓ_∞ barrier") [38, 141, 195, 410, 427].

Given these challenges toward tight robustness certification, a natural question arises:

Q1: *Is it possible to circumvent the barrier of randomized smoothing by certifying with additional "information"?*

Q2: *What type of information is needed to provide tight robustness certification?*

To answer these questions, we propose a **Double Sampling Randomized Smoothing** (DSRS) framework to leverage the sampled noises from an *additional smoothing distribution* as additional information to tighten the robust certification. In theory, we show that (1) ideally, if the decision region of the base classifier is known, DSRS can provide tight robustness certification; (2) more practically, if the inputs, which can be correctly classified by the base classifier, satisfy the concentration property within an input-centered ball with constant mass under standard Gaussian measure, the standard Neyman-Pearson-based certification [77, 218, 317, 427] can certify only a dimension-independent ℓ_2 radius, whereas DSRS with generalized Gaussian smoothing can certify radius $\Omega(\sqrt{d})$ (under ℓ_2 norm), which would increase with the dimension d, leading to tighter certification. Under more general conditions, we provide numerical simulations to verify our theory. Our results provide a positive answer to Q1 and sufficient conditions for Q2, i.e., DSRS may be able to circumvent the barrier of randomized smoothing.

Motivated by the theory, we leverage a type of generalized Gaussian [442] as the smoothing distribution and truncated generalized Gaussian as an additional distribution. For this type of concretization, we propose an efficient and sound computation method to compute the certifiably robust radius for practical classifiers considering sampling error. Our method formulates the certification problem given additional information as a constrained optimization problem and leverages specific properties of the dual problem to decompose the effects of different dual variables to solve it. DSRS is fully scalable since the computational time is nearly independent of the size of the dataset, model, or sampling. Our extensive experimental evaluation on MNIST, CIFAR-10, and ImageNet shows that (1) under large sampling size ($2\times10^5-8\times10^5$), the certified radius of DSRS consistently increases as suggested by our theory; (2) under practical sampling size (10^5), DSRS can certify consistently higher robust radii than existing baselines, including standard Neyman-Pearson-based certification.

As further discussed in Appendix B.6, we believe that DSRS as a framework opens a wide

range of future directions for selecting or optimizing different forms of *additional information* to tighten the certification of randomized smoothing.

We summarize the main technical contributions as follows:

- We propose a general robustness certification framework DSRS, which leverages additional information by sampling from another smoothing distribution.
- We prove that under practical concentration assumptions, DSRS certifies $\Omega(\sqrt{d})$ radius under ℓ_2 norm with d the input dimension, suggesting a possible way to circumvent the intrinsic barrier of randomized smoothing.
- We concretize DSRS by generalized Gaussian smoothing mechanisms and propose a method to efficiently compute the certified radius for given classifiers.
- We conduct extensive experiments, showing that DSRS provides consistently tighter robustness certification than existing baselines, including standard Neyman-Pearson-based certification across different models on MNIST, CIFAR-10, and ImageNet.

2.1 RELATED WORK

For the certification method of randomized smoothing, most existing methods leverage only the true-class prediction probability to certify. In this case, the tightest possible robustness certification is based on the Neyman-Pearson lemma [276] as first proposed by Cohen et al. [77] for certifying ℓ_2 radius under Gaussian smoothing. Several methods extend this certification to accommodate different smoothing distributions and different ℓ_p norms [106, 214, 427, 442]. In randomized smoothing, the ℓ_2 certified robust radius r is similar across datasets of different scales, resulting in the vanishing ℓ_∞ certified radius $r/\sqrt{d}$ when input dimension increases. This limitation of existing certification methods of randomized smoothing is formally proved [38, 141, 195, 410, 427] and named "ℓ_∞ barrier" or "curse of dimensionality".

Recent work tries to incorporate additional information besides true-class prediction probability to tighten the certification and bypass the barrier. For ℓ_2 and ℓ_∞ certification, to the best of our knowledge, gradient magnitude is the only exploited additional information [215, 267]. However, in practice, the improvement is relatively marginal and requires a large number of samples (see Appendix B.5.5). Some other methods provide tighter certification given specific model structures [16, 70, 194, 205]. DSRS instead focuses on leveraging model-structure-agnostic additional information.

Recently, it is proposed that another potential way to improve the certified robustness of randomized smoothing is to dynamically change the smoothing distribution based on the input toward maximizing the certified radius [6, 109, 323]. In this scenario, the certification needs to take into account that the attacker may adaptively mislead the pipeline to choose a "bad" smoothing distribution. Therefore, additional costs such as memorizing training data need to be paid to defend such adaptive robustness vulnerabilities. A recent work [354] shows that input-dependent randomized smoothing may not bring substantial improvements in certified robustness. In DSRS, we select the additional smoothing distribution dynamically based on the input, which may appear like input-dependent randomized smoothing. However, we select such distribution only for certification purposes, and the original distribution that is used to construct the smoothed classifier remains static. Thus, we do not need to consider the existence of adaptive attackers.

To improve the certified robustness of randomized smoothing, besides certification, efforts have also been made on the training [165, 218, 317, 437] side. On the training side, data augmentation [77], regularization [165, 218, 437], and adversarial training [317] help to train stable base models under noise corruptions so that higher certified robustness for a smoothed classifier can be achieved. In this work, we focus on certification, and these training approaches can be used in conjunction with ours to provide higher certified robustness.

2.2 PRELIMINARIES AND BACKGROUND

Let $\mathbf{\Delta}^C$ be the C-dimensional probability simplex. We consider a multiclass classification model $F : \mathbb{R}^d \to [C]$ as the *base classifier*, where d is the input dimension, and the model outputs hard-label class prediction within $[C]$. The *original smoothing distribution* $\mathcal{P}$ and *additional smoothing distribution* $\mathcal{Q}$ are both supported on $\mathbb{R}^d$. We let $p(\cdot)$ and $q(\cdot)$ be their density functions respectively. We assume that both p and q are positive and differentiable almost everywhere, i.e., the set of singular points has zero measure under either $\mathcal{P}$ or $\mathcal{Q}$. These assumptions hold for common smoothing distributions used in the literature such as Gaussian distribution [77, 204, 218, 427].

Randomized smoothing constructs a smoothed classifier from a given base classifier by adding input noise following *original* smoothing distribution $\mathcal{P}$. For input $\boldsymbol{x} \in \mathbb{R}^d$, we define *prediction probability under* $\mathcal{P}$ by function $f^{\mathcal{P}} : \mathbb{R}^d \to \mathbf{\Delta}^C$:

$$f^{\mathcal{P}}(\boldsymbol{x})_c := \Pr_{\boldsymbol{\epsilon} \sim \mathcal{P}}[F(\boldsymbol{x} + \boldsymbol{\epsilon}) = c] \quad \text{where} \quad c \in [C]. \tag{2.1}$$

The *smoothed classifier* $\widetilde{F}^{\mathcal{P}} : \mathbb{R} \to [C]$ (or $\widetilde{F}$ when $\mathcal{P}$ is clear from the context) predicts the

class with the highest confidence after smoothing with $\mathcal{P}$:

$$\widetilde{F}^{\mathcal{P}}(\boldsymbol{x}) := \arg\max_{c\in[C]} f^{\mathcal{P}}(\boldsymbol{x})_c. \tag{2.2}$$

We focus on robustness certification against ℓ_p-bounded perturbations for smoothed classifier $\widetilde{F}$, where the standard certification method is called Neyman-Pearson-based certification [77] (details illustrated below). Concretely, certification methods compute robust radius r defined as below.

Definition 2.1 (Certified Robust Radius)**.** Under ℓ_p norm ($p \in \mathbb{R}_+ \cup \{+\infty\}$), for given smoothed classifier $\widetilde{F}^{\mathcal{P}}$ and input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with true label $y_0 \in [C]$, a radius $r \geqslant 0$ is called *certified (robust) radius* for $\widetilde{F}^{\mathcal{P}}$ if $\widetilde{F}^{\mathcal{P}}$ always predicts y_0 for any input within the r-radius ball centered at $\boldsymbol{x}_0$:

$$\forall \boldsymbol{\delta} \in \mathbb{R}^d, \|\boldsymbol{\delta}\|_p < r, \widetilde{F}^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta}) = y_0. \tag{2.3}$$

Neyman-Pearson Certification

The Neyman-Pearson-based robustness certification is the tightest certification given only prediction probability under $\mathcal{P}$ [77]. This certification and its equivalent variants are widely used for randomized smoothing. We use $r_{\mathsf{N-P}}$ to represent the certified radius from the Neyman-Pearson-based method.

If the smoothing distribution $\mathcal{P}$ is standard Gaussian, the following proposition gives the closed-form certified robust radius derived from the Neyman-Pearson lemma [276].

Proposition 2.1 ([77])**.** Under ℓ_2 norm, given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with true label y_0. Let $\mathcal{P} = \mathcal{N}(\sigma)$ be the smoothing distribution, then Neyman-Pearson-based certification yields certified radius $r_{\mathsf{N-P}} = \sigma\Phi^{-1}(f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0})$, where Φ^{-1} is the inverse CDF of unit-variance Gaussian.

For other smoothing distributions, the concretization of the Neyman-Pearson certification method can be found in [427].

Remark 2.1. In practice, the routine is to use Monte-Carlo sampling to obtain a high-confidence interval of $f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}$, which implies a high-confidence certification ($r_{\mathsf{N-P}}$) of robust radius. A tighter radius can be obtained when the runner-up prediction probability is known: $r'_{\mathsf{N-P}} = \frac{\sigma}{2}\left(\Phi^{-1}(f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}) - \max_{y\in[C]:y\neq y_0} \Phi^{-1}(f^{\mathcal{P}}(\boldsymbol{x}_0)_y)\right)$. However, due to efficiency concern (for C-way classification the sampling number needs to be more than C times if using $r'_{\mathsf{N-P}}$ for certification instead of $r_{\mathsf{N-P}}$), the standard routine is to only use top-class probability and $r_{\mathsf{N-P}}$ [77, Section 3.2.2]. DSRS follows this routine.

2.3 DSRS OVERVIEW

We propose Double Sampling Randomized Smoothing (DSRS), which leverages the prediction probability from an *additional* smoothing distribution $\mathcal{Q}$ (formally $Q_A := f^{\mathcal{Q}}(\boldsymbol{x}_0)_{y_0} = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[F(\boldsymbol{x}+\boldsymbol{\epsilon}) = y_0]$), along with the prediction probability from the original smoothing distribution $\mathcal{P}$ (formally $P_A := f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}$ as in Equation (2.1), also used in Neyman-Pearson-based certification), to provide robustness certification for $\mathcal{P}$-smoothed classifier $\widetilde{F}^{\mathcal{P}}$. Note that both P_A and Q_A can be obtained from Monte-Carlo sampling (see Sections 2.5.1 and 2.5.2). Formally, we let r_{DSRS} denote the tightest possible certified radius with prediction probability from $\mathcal{Q}$, then r_{DSRS} can be defined as below.

Definition 2.2 (r_{DSRS})**.** Given P_A and Q_A,

$$\begin{aligned} & r_{\mathsf{DSRS}} := \max r \quad \text{s.t.} \\ & \forall F : \mathbb{R} \to [C], f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0} = P_A, f^{\mathcal{Q}}(\boldsymbol{x}_0)_{y_0} = Q_A \\ \Rightarrow & \forall \boldsymbol{x}, \|\boldsymbol{x} - \boldsymbol{x}_0\|_p < r, \widetilde{F}^{\mathcal{P}}(\boldsymbol{x}) = y_0. \end{aligned} \tag{2.4}$$

Intuitively, r_{DSRS} is the maximum possible radius, such that any smoothed classifier constructed from base classifier satisfying P_A and Q_A constraints cannot predict other labels when the perturbation magnitude is within the radius.

In Section 2.4, we will analyze the theoretical properties of DSRS, including comparing r_{DSRS} and $r_{\mathsf{N-P}}$ under the concentration assumption. Computing r_{DSRS} is nontrivial, so in Section 2.5, we will introduce a practical computational method that exactly solves r_{DSRS} when $\mathcal{P}$ and $\mathcal{Q}$ are standard and generalized (truncated) Gaussian. In Appendix B.2, we will show method variants to deal with other forms of $\mathcal{P}$ and $\mathcal{Q}$ distributions. In Appendix B.6, we will further generalize the DSRS framework.

Smoothing Distributions. Now we formally define the smoothing distributions used in DSRS. We mainly consider standard Gaussian $\mathcal{N}$ [77, 427] and generalized Gaussian $\mathcal{N}^{\mathtt{g}}$ [442]. Let $\mathcal{N}(\sigma)$ to represent standard Gaussian distribution with covariance matrix $\sigma^2\boldsymbol{I}_d$ that has density function $\propto \exp(-\|\boldsymbol{\epsilon}\|_2^2/(2\sigma^2))$.[3] For $k \in \mathbb{N}$, we let $\mathcal{N}^{\mathtt{g}}(k,\sigma)$ to represent generalized Gaussian whose density function $\propto \|\boldsymbol{\epsilon}\|_2^{-2k} \exp(-\|\boldsymbol{\epsilon}\|_2^2/(2\sigma'^2))$ where $\sigma' = \sqrt{d/(d-2k)}\sigma$. Here we use σ' instead of σ to ensure that the expected noise $\sqrt{\mathbb{E}\,\|\boldsymbol{\epsilon}\|_2^2}$ of $\mathcal{N}^{\mathtt{g}}(k,\sigma)$ is the same as $\mathcal{N}(\sigma)$. The generalized Gaussian as the smoothing distribution overcomes the "thin shell" problem of standard Gaussian and improves certified robustness [442]; and we will reveal more of its theoretical advantages in Section 2.4.

[3] In this chapter, $\mathcal{N}(\sigma)$ is a shorthand of $\mathcal{N}(0, \sigma^2\boldsymbol{I}_d)$.

Table 2.1: Definitions of smoothing distributions in this chapter. In the table, $k \in \mathbb{N}$, $\sigma' = \sqrt{d/(d-2k)}\sigma$.

Name	Notation	Density Function
Standard Gaussian	$\mathcal{N}(\sigma)$	$\propto \exp\left(-\frac{\|\boldsymbol{\epsilon}\|_2^2}{2\sigma^2}\right)$
Generalized Gaussian	$\mathcal{N}^{\mathsf{g}}(k,\sigma)$	$\propto \|\boldsymbol{\epsilon}\|_2^{-2k} \exp\left(-\frac{\|\boldsymbol{\epsilon}\|_2^2}{2\sigma'^2}\right)$
Truncated Standard Gaussian	$\mathcal{N}_{\mathsf{trunc}}(T,\sigma)$	$\propto \exp\left(-\frac{\|\boldsymbol{\epsilon}\|_2^2}{2\sigma^2}\right) \cdot \mathbb{I}[\|\boldsymbol{\epsilon}\|_2 \leqslant T]$
Truncated Generalized Gaussian	$\mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k,T,\sigma)$	$\propto \|\boldsymbol{\epsilon}\|_2^{-2k} \exp\left(-\frac{\|\boldsymbol{\epsilon}\|_2^2}{2\sigma'^2}\right) \cdot \mathbb{I}[\|\boldsymbol{\epsilon}\|_2 \leqslant T]$

As the additional smoothing distribution $\mathcal{Q}$, we will mainly consider truncated distributions within a small ℓ_2 radius ball. Specially, truncated standard Gaussian is denoted by $\mathcal{N}_{\mathsf{trunc}}(T,\sigma)$ with density function $\propto \exp(-\|\boldsymbol{\epsilon}\|_2^2/(2\sigma^2)) \cdot \mathbb{I}[\|\boldsymbol{\epsilon}\|_2 \leqslant T]$; and truncated generalized Gaussian is denoted by $\mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k,T,\sigma)$ with density function $\propto \|\boldsymbol{\epsilon}\|_2^{-2k} \exp(-\|\boldsymbol{\epsilon}\|_2^2/(2\sigma'^2)) \cdot \mathbb{I}[\|\boldsymbol{\epsilon}\|_2 \leqslant T]$.

In Table 2.1, we summarize these distribution definitions.

2.4 THEORETICAL ANALYSIS OF DSRS

In this section, we theoretically analyze DSRS to answer the following core question: *Does Q_A, the prediction probability under additional smoothing distribution, provide sufficient information for tightening the robustness certification?* We first show that if the support of $\mathcal{Q}$ is the decision region of true class, DSRS can certify the smoothed classifier's maximum possible robust radius. Then, under concentration assumption, we show the ℓ_2 certified radius of DSRS can be $\Omega(\sqrt{d})$ that is asymptotically optimal for bounded inputs. Finally, under more general conditions, we conduct both numerical simulations and real-data experiments to verify that the certified radius of DSRS increases with data dimension d. These analyses provide a positive answer to the above core question.

DSRS can certify the tightest possible robust radius.

Given an original smoothing distribution $\mathcal{P}$ and a base classifier F_0. At input point $\boldsymbol{x}_0 \in \mathbb{R}^d$ with true label y_0, we define the tightest possible certified robust radius r_{tight} to be the largest ℓ_p ball that contains no adversarial example for *smoothed* classifier $\widetilde{F}_0^{\mathcal{P}}$:

$$r_{\mathsf{tight}} := \max r \quad \text{s.t.} \quad \forall \boldsymbol{\delta} \in \mathbb{R}^d, \|\boldsymbol{\delta}\|_p < r, \widetilde{F}_0^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta}) = y_0. \tag{2.5}$$

Then, for binary classification, if we choose an additional smoothing distribution $\mathcal{Q}$ whose support is the decision region or its complement, then DSRS can certify robust radius r_{tight}.

Theorem 2.1. Suppose the original smoothing distribution $\mathcal{P}$ has non-zero density everywhere, i.e., $p(\cdot) > 0$. For binary classification with base classifier F_0, at point $\boldsymbol{x}_0 \in \mathbb{R}^d$, let $\mathcal{Q}$ be an additional distribution that satisfies: (1) its support is the decision region of an arbitrary class $c \in [C]$ shifted by $\boldsymbol{x}_0$: $\mathrm{supp}(\mathcal{Q}) = \{\boldsymbol{x} - \boldsymbol{x}_0 : F_0(\boldsymbol{x}) = c\}$; (2) for any $\boldsymbol{x} \in \mathrm{supp}(\mathcal{Q})$, $0 < q(\boldsymbol{x})/p(\boldsymbol{x}) < +\infty$. Then, plugging $P_A = f_0^{\mathcal{P}}(\boldsymbol{x}_0)_c$ and $Q_A = f_0^{\mathcal{Q}}(\boldsymbol{x}_0)_c$ (see Equation (2.1)) into Definition 2.2, we have $r_{\mathsf{DSRS}} = r_{\mathsf{tight}}$ under any ℓ_p ($p \geqslant 1$).

Proof sketch. We defer the proof to Appendix B.1.1. At a high level, with this type of $\mathcal{Q}$, we have $Q_A = 1$ or $Q_A = 0$. Then, from the mass of the $\mathcal{Q}$'s support on $\mathcal{P}$ and P_A, we can conclude that the $\mathcal{Q}$'s support is exactly the decision region of label c or its complement. Thus, the DSRS constraints (in Equation (2.4)) are satisfied iff F differs from F_0 in a zero-measure set, and thus we exactly compute the smoothed classifier $\widetilde{F}_0^{\mathcal{P}}$'s maximum certified robust radius in DSRS. An extension to multiclass setting is in Appendix B.1.2. QED.

Remark 2.2. For any base classifier F_0, $\mathcal{Q}$ that satisfies conditions in Theorem 2.1 exists, implying that with DSRS, certifying a strictly tight robust radius is possible. In contrast, Neyman-Pearson-based is proved to certify tight robust radius for linear base classifiers [77, Section 3.1], but for arbitrary base classifiers, its tightness is not guaranteed. This result suggests that, to certify a tight radius, just one additional smoothing distribution $\mathcal{Q}$ is sufficient rather than multiple ones.

On the other hand, it is challenging to find $\mathcal{Q}$ whose support (or its complement) exactly matches the decision region of an NN classifier. In the following, we analyze the tightness of DSRS under weaker assumptions.

DSRS can certify $\Omega(\sqrt{d})$ ℓ_2 radius under concentration assumption.

We begin by defining the concentration property.

Definition 2.3 ($(\sigma, P_{\mathsf{con}})$-Concentration)**.** Given a base classifier F_0, at input $\boldsymbol{x}_0 \in \mathbb{R}$ with true label y_0, we call F_0 satisfies $(\sigma, P_{\mathsf{con}})$-concentration property, if for within P_{con}-percentile of small ℓ_2 magnitude Gaussian $\mathcal{N}(\sigma)$ noise, the adversarial example occupies zero measure. Formally, $(\sigma, P_{\mathsf{con}})$-concentration means

$$\Pr_{\boldsymbol{\epsilon} \sim \mathcal{N}(\sigma)}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = y_0 \,|\, \|\boldsymbol{\epsilon}_0\|_2 \leqslant T] = 1 \tag{2.6a}$$

$$\text{where} \quad T \text{ satisfies } \Pr_{\boldsymbol{\epsilon} \sim \mathcal{N}(\sigma)}[\|\boldsymbol{\epsilon}\|_2 \leqslant T] = P_{\mathsf{con}}. \tag{2.6b}$$

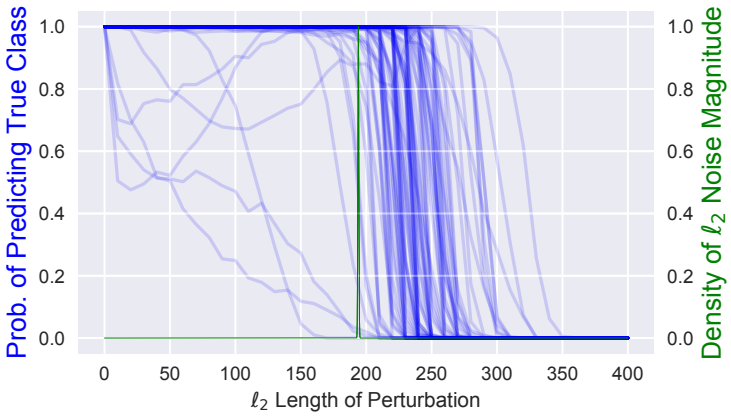


Figure 2.3: Blue curves: Probability of true-prediction w.r.t. ℓ_2 length of perturbations for a base classifier from [317] on ImageNet. Each line corresponds to one of 100 uniformly drawn samples from test set (detailed setup in Appendix B.5.1). Green curve: Normalized density of ℓ_2 noise magnitude for ImageNet standard Gaussian $\mathcal{N}(\sigma)$ with $\sigma = 0.5$, which highly concentrates on $\sigma\sqrt{d}$. *Thus, for constant P_{con}, $(\sigma, P_{\mathsf{con}})$-concentration can be satisfied for a significant portion of input samples.*

Intuitively, $(\sigma, P_{\mathsf{con}})$-concentration implies that the base classifier has few adversarial examples for small magnitude noises during standard Gaussian smoothing. In Figure 2.3, we empirically verified that a well-trained base classifier on ImageNet may satisfy this property for a significant portion of inputs. Furthermore, Salman et al. [317] show that promoting this concentration property by adversarially training the smoothed classifier improves the certified robustness. With this concentration property, DSRS certifies the radius $\Omega(\sqrt{d})$ under ℓ_2 norm, as the following theorem shows.

Theorem 2.2. Let d be the input dimension and F_0 be the base classifier. For an input point $\boldsymbol{x}_0 \in \mathbb{R}^d$ with true class y_0, suppose F_0 satisfies $(\sigma, P_{\mathsf{con}})$-Concentration property. Then, for any sufficiently large d, for the classifier $\widetilde{F}_0^{\mathcal{P}'}$ smoothed by generalized Gaussian $\mathcal{P}' = \mathcal{N}^{\mathsf{g}}(k, \sigma)$ with $d/2 - 15 \leqslant k < d/2$, DSRS with additional smoothing distribution $\mathcal{Q} = \mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$ can certified ℓ_2 radius

$$r_{\mathsf{DSRS}} \geqslant 0.02\sigma\sqrt{d} \tag{2.7}$$

where $T = \sigma\sqrt{2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}})}$ and $\Gamma\mathrm{CDF}_{d/2}$ is the CDF of gamma distribution $\Gamma(d/2, 1)$.

Proof sketch. We defer the proof to Appendix B.1.3. At high level, based on the standard Gaussian distribution's property (Proposition B.1), we find $Q_A = 1$ under concentration property (Lemma B.1). With $Q_A = 1$, we derive a lower bound of r_{DSRS} in Lemma B.2. We then use: (1) the concentration of beta distribution $\mathrm{Beta}(\frac{d-1}{2}, \frac{d-1}{2})$ (see Lemma B.3) for large d; (2) the relative concentration of gamma $\Gamma(d/2, 1)$ distribution around mean for

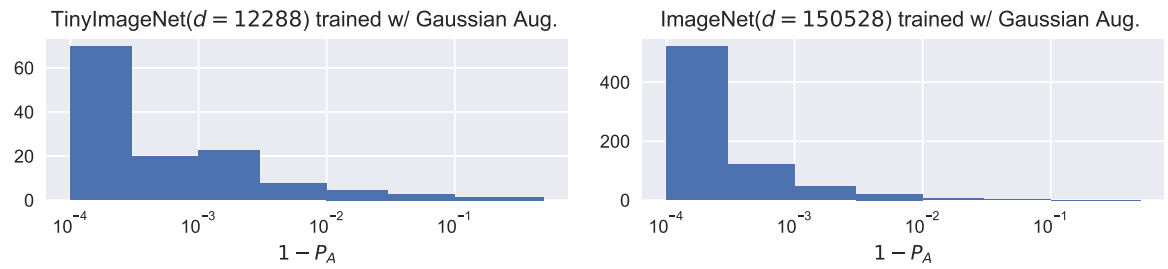


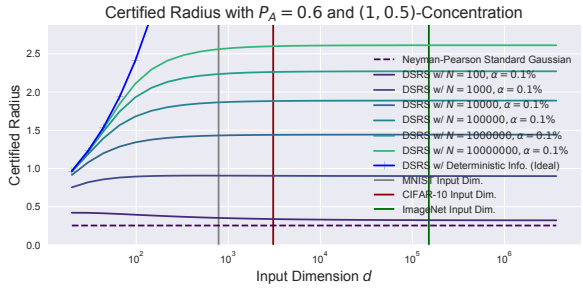
Figure 2.4: P_A histograms for models trained on TinyImageNet (left) and ImageNet (right) with same $\sigma = 0.50$. P_A histograms are based on $1,000$ test samples from TinyImageNet and ImageNet. Note that TinyImageNet images are downscaled ImageNet images, so the data distribution only differs in the input dimension d. As we can observe, though d varies, the P_A distribution is highly similar, so P_A is roughly unchanged along with the increase of input dimension d.

large d (see Proposition B.2 and resulting Fact B.1); and (3) the misalignment of gamma distribution $\Gamma(d/2 - k, 1)$'s mean and median for small $(d/2 - k)$ (see Proposition B.3) to lower bound the quantity in Lemma B.2 and show it is large or equal to 0.5. Then, using the conclusion in Section 2.5 we conclude that $r_{\mathsf{DSRS}} \geqslant 0.02\sigma\sqrt{d}$. QED.

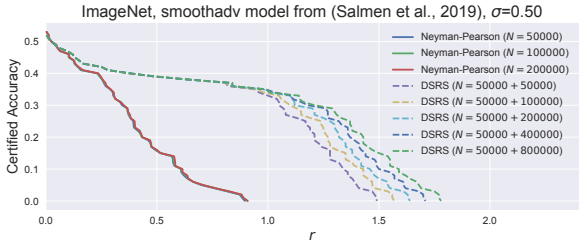
Remark 2.3. (1) For standard Neyman-Pearson based certificaton, $r_{\mathsf{N-P}} = \sigma\Phi^{-1}(f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0})$. Along with the increase of input dimension d, to achieve growing ℓ_2 certified radius, one needs the prediction probability of true class under $\mathcal{P}$, namely $f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}$, to grow simultaneously, which is challenging. Indeed, across different datasets, $f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}$ is almost a constant, which leads to a constant ℓ_2 certified radius and shrinking ℓ_∞ radius for large d. We further empirically illustrate this property in Figure 2.4.

(2) In contrast, as long as the model satisfies concentration property, which may be almost true on large datasets as reflected by Figure 2.3, with our specific choices of $\mathcal{P}$ and $\mathcal{Q}$, DSRS can achieve $\Omega(\sigma\sqrt{d})$ ℓ_2 radius on large datasets. This rate translates to a constant $\Omega(\sigma)$ ℓ_∞ radius on large datasets and thus breaks the curse of dimensionality of randomized smoothing. We remark that this $\sqrt{d}$ rate is optimal when dataset input is bounded such as images (otherwise, the $\omega(1)$ ℓ_∞ radius leads the radius to exceed the constant ℓ_∞ diameter for large d). Therefore, *under the assumption of concentration property, DSRS provides asymptotically optimal certification for randomized smoothing.*

(3) Smoothing with generalized Gaussian distribution and choosing a parameter k that is close to $d/2$ play an essential role in proving the $\Omega(\sigma\sqrt{d})$ certified radius. Otherwise, in Appendix B.1.4 we have Theorem B.1 that shows any certification methods cannot certify an ℓ_2 radius $c\sqrt{d}$ for any $c > 0$. This adds another theoretical evidence for the superiority of generalized Gaussian that is cross-validated by Zhang et al. [442].



(a) When holding probability in Equation (2.6a) is obtained from sampling N times and confidence level $1 - \alpha = 99.9\%$, relation between certified radius (y-axis) and input data dimension d (x-axis). Different curves correspond to different Ns.



(b) Relation between certified radius (x-axis) and certified accuracy (y-axis) on ImageNet models. Different curves correspond to Neyman-Pearson and DSRS certification with different Ns. Sampling error considered, confidence level = 99.9%.

Figure 2.5: Tendency of DSRS certified robust radius considering sampling error. In both **(a)** and **(b)**, DSRS certified radius grows along with the increase of sampling number N but Neyman-Pearson radius is almost fixed.

DSRS certifies tighter radius under general scenarios.

When the concentration property does not absolutely hold, a rigorous theoretical analysis becomes challenging, since the impact of a noninfinite dual variable needs to be taken into account. This dual variable is inside a Lambert W function where typical approximation bounds are too loose to provide non-trivial convergence rates. Thus, we leverage the numerical computational method introduced in Section 2.5 to provide numerical simulations and real-data experiments. We generalize the concentration assumption by changing the holding probability in Equation (2.6a) from 1 to $\alpha^{1/N}$, which corresponds to $(1-\alpha)$-confident lower bound of Q_A given N times of Monte-Carlo sampling, where we set $\alpha = 0.1\%$ following the convention [77]. In this scenario, we compare DSRS certification with Neyman-Pearson certification numerically in Figure 2.5 (numerical simulations in Figure 2.5(a) and ImageNet experiments in Figure 2.5(b)).

In Figure 2.5(a), we assume $(\sigma, P_{\mathsf{con}})$-concentration with $\sigma = 1$, $P_{\mathsf{con}} = 0.5$ and different sampling number Ns. We further assume $P_A = f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0} = 0.6$ as the true-class prediction probability under $\mathcal{P}$. In Figure 2.5(b), we take the model weights trained by Salman et al. [317] on ImageNet and apply generalized Gaussian smoothing with $d/2-k = 4$ and $\sigma = 0.50$. We uniformly pick 100 samples from the test set and compute $(1-\alpha)$-confident certified radius for each sample. We report certified accuracy (under different ℓ_2 radius r) that is the fraction of certifiably correctly classified samples by the smoothed classifier.

Remark 2.4. When the sampling error and confidence interval come into play, they quickly suppress the $\Omega(\sqrt{d})$ growth rate of DSRS certified radius (blue curve) as shown in Fig-

ure 2.5(a). Nonetheless, DSRS still certifies a larger radius than the standard Neyman-Pearson method and increasing the sampling number further enlarges the gap.

Instead of generalizing the concentration assumption by replacing the holding probability 1 in Equation (2.6a) by probability considering sampling confidence, we replace the holding probability in Equation (2.6a) by $\exp(-d^\alpha)$ for $\alpha \in \{0.1, 0.2, 0.3, 0.4, 0, 5\}$.

With this relaxation, we conduct numerical simulations using the same settings as above, and the corresponding results are shown in Figure 2.6. Note that some solid curves terminate when d is large, which is due to the limitation of floating-point precision in numerical simulations, and we use dashed lines of the same color to plot the projected radius when d is large.

Remark 2.5. When the concentration property holds with probability $\exp(-d^\alpha)$ $(0 < \alpha \leqslant 0.5)$ other than 1, from Figure 2.6, we observe that $r_{\mathsf{N-P}} d^{\alpha/1.18}$ predicts the certified radius of DSRS well where $r_{\mathsf{N-P}}$ is Neyman-Pearson certified radius. Therefore, although the $\sqrt{d}$ growth rate of ℓ_2 certified radius does not hold, the radius still increases along with the dimension d. Interestingly, along with the increase of dimension d, the vanishing probability $\exp(-d^\alpha)$ still implies the increasing volume of adversarial examples, and smoothed classifier is still certifiably robust with increasing radius reflected by DSRS despite the increasing adversarial volume.

2.5 DSRS COMPUTATIONAL METHOD

The theoretical analysis in Section 2.4 implies that *additional smoothing distribution* $\mathcal{Q}$ helps to tighten the robustness certification over standard Neyman-Pearson-based certification significantly. In this section, we propose an efficient computational method to compute this tight certified robust radius r_{DSRS} (see Definition 2.2) when $\mathcal{P}$ is generalized Gaussian and $\mathcal{Q}$ is truncated $\mathcal{P}$ as suggested by Theorems 2.2 and B.1.

Compared with the classical certification for randomized smoothing or its variants (cf. [194]), incorporating additional information raises a big challenge: the Neyman-Pearson lemma ([276]) can no longer be served as the foundation of the certification algorithm due to its incapability to handle the additional information.

Thus, we propose a novel DSRS computational method by formalizing robustness certification as a constrained optimization problem and proving its strong duality (§2.5.1). Then, we propose an efficient algorithm to solve this specific dual optimization problem considering sampling error. The detailed algorithm can be found in Algorithm 2.1 in Section 2.5.4: 1)

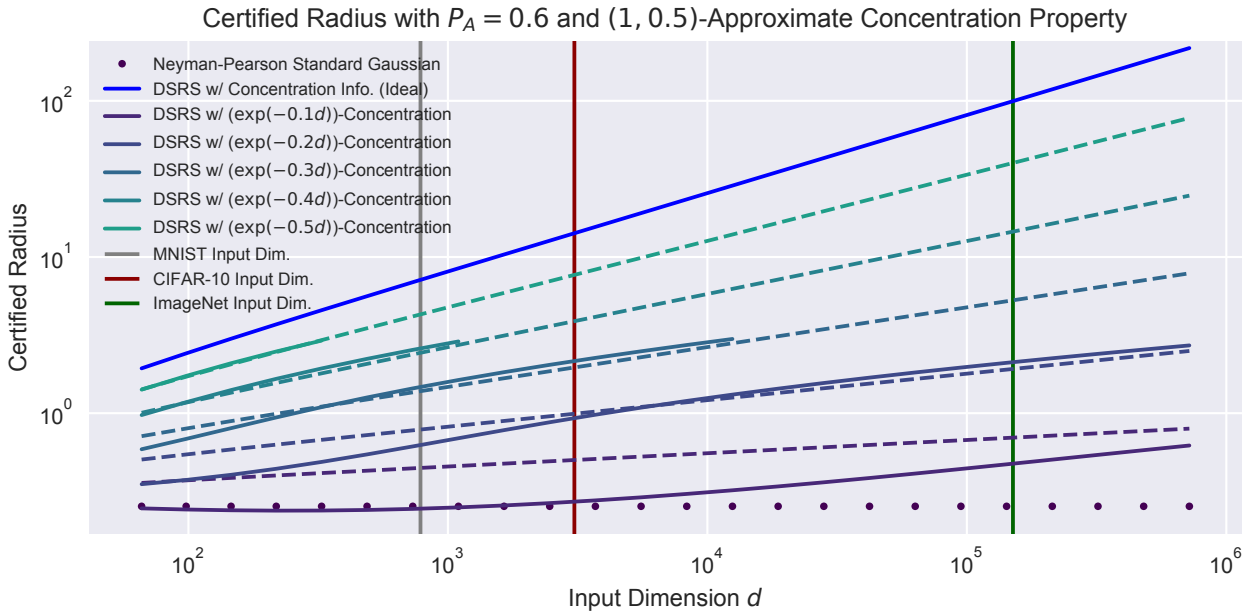


Figure 2.6: Tendency of DSRS certified robust radius with different input dimensions d under relaxed concentration assumption: when holding probability in Equation (2.6a) is $\exp(-d^\alpha)$ with α from 0.1 to 0.5;. Blue line: DSRS when holding probability in Equation (2.6a) is 1. Dotted line: Neyman-Pearson certification. Other solid lines: DSRS when holding probability in Equation (2.6a) is $\exp(-d^\alpha)$. Other dashed lines: DSRS projected radius by $r^{\text{proj}}_{\text{DSRS}} = r_{\text{N-P}} d^{\alpha/1.18}$. $\alpha \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$. Both x- and y-axes are logarithmic.

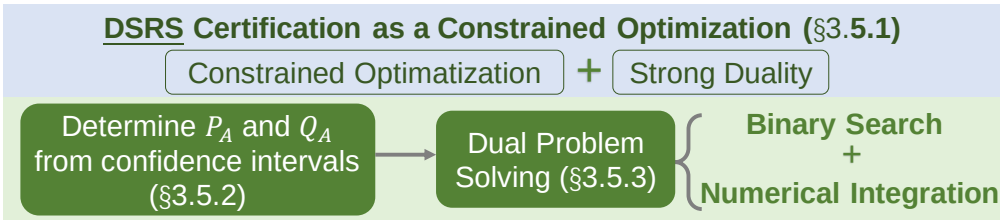


Figure 2.7: Overview of DSRS computational method.

we first perform a binary search on the certified radius r to determine the maximum radius that we can certify; 2) for current r, we determine the smoothed prediction confidence P_A and Q_A from the confidence intervals of predicting the true class (§2.5.2); 3) then, for current r we solve the dual problem by quick binary search for dual variables λ_1 and λ_2 (see Equation (2.11)) along with numerical integration (§2.5.3). To guarantee the soundness of numerical-integration-based certification, we take the maximum possible error into account during the binary search. We will discuss further extensions in §2.5.7.

2.5.1 DSRS as Constrained Optimization

We first formulate the robustness certification as a constrained optimization problem and then show several foundational properties of the problem.

Following the notation of Definition 2.2, from the given base classifier F_0, we can use Monte-Carlo sampling to obtain

$$P_A = f_0^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}, \quad Q_A = f_0^{\mathcal{Q}}(\boldsymbol{x}_0)_{y_0}. \tag{2.8}$$

In §2.5.2 we will discuss how to handle confidence intervals of P_A and Q_A. For now, we assume P_A and Q_A are fixed.

Given perturbation vector $\boldsymbol{\delta} \in \mathbb{R}^d$, to test whether smoothed classifier $\widetilde{F}_0^{\mathcal{P}}$ still predicts true label y_0, we only need to check whether the prediction probability $f_0^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta})_{y_0} > 0.5$. This can be formulated as a constrained optimization problem ($\mathbf{C}$):

$$\begin{aligned}
\underset{f}{\text{minimize}} \quad & \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon} + \boldsymbol{\delta})] && \text{(2.9a)}\\
\text{s.t.} \quad & \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon})] = P_A, \quad \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[f(\boldsymbol{\epsilon})] = Q_A, && \text{(2.9b)}\\
& 0 \leqslant f(\boldsymbol{\epsilon}) \leqslant 1 \quad \forall \boldsymbol{\epsilon} \in \mathbb{R}^d. && \text{(2.9c)}
\end{aligned}$$

Remark 2.6. ($\mathbf{C}$) seeks for the minimum possible $f^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta})_{y_0}$ given Equation (2.8)'s constraint. Concretely, we let f represent whether the base classifier predicts label y_0: $f(\cdot) = \mathbb{I}[F(\cdot + \boldsymbol{x}_0) = y_0]$, and accordingly impose $f \in [0, 1]$ in Equation (2.9c). Then, Equations (2.9a) and (2.9b) unfold $f^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta})_{y_0}, f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}$, and $f^{\mathcal{Q}}(\boldsymbol{x}_0)_{y_0}$ respectively and impose Equation (2.8)'s constraint.

We let $\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A)$ denote the optimal value of Equation (2.9) when feasible. Thus, under norm p, to certify the robustness within radius r, we only need to check whether

$$\forall \boldsymbol{\delta}, \|\boldsymbol{\delta}\|_p < r \Rightarrow \mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) > 0.5. \tag{2.10}$$

This formulation yields the tightest robustness certification given information from $\mathcal{P}$ and $\mathcal{Q}$ under the binary setting. Under the multiclass setting, there are efforts towards tighter certification by using "> maximum over other classes" instead of "> 0.5" in Equation (2.10) [106]. For saving the sampling cost and also to follow the convention [77, 165, 427, 437], we mainly consider "> 0.5" for multiclass setting, and extension to the other form is straightforward.

Since our choices of $\mathcal{P}$ and $\mathcal{Q}$ (standard/generalized (truncated) Gaussian) are isotropic and centered around origin, when certifying radius r, for ℓ_2 certification we only need to test

$\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) > 0.5$ with $\boldsymbol{\delta} = (r, 0, \ldots, 0)^\intercal$; and for ℓ_∞ we only need to divide ℓ_2 radius by $\sqrt{d}$. This trick can also be extended for ℓ_1 case [442].

Directly solving ($\mathbf{C}$) is challenging. Thus, we construct the Lagrangian dual problem ($\mathbf{D}$):

$$\underset{\lambda_1, \lambda_2 \in \mathbb{R}}{\text{maximize}} \ \Pr_{\boldsymbol{\epsilon} \sim \mathcal{P}}[p(\boldsymbol{\epsilon}) < \lambda_1 p(\boldsymbol{\epsilon} + \boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{\epsilon} + \boldsymbol{\delta})] \tag{2.11a}$$

$$\begin{aligned} \text{s.t.} \ \Pr_{\boldsymbol{\epsilon} \sim \mathcal{P}}[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] &= P_A, \\ \Pr_{\boldsymbol{\epsilon} \sim \mathcal{Q}}[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] &= Q_A. \end{aligned} \tag{2.11b}$$

In Equation (2.11), $p(\cdot)$ and $q(\cdot)$ are the density functions of distributions $\mathcal{P}$ and $\mathcal{Q}$ respectively. We let $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$ denote the optimal objective value to Equation (2.11a) when it is feasible.

Theorem 2.3. For given $\boldsymbol{\delta} \in \mathbb{R}^d$, P_A, and Q_A, if $\mathbf{C}$ and $\mathbf{D}$ are both feasible, then $\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) = \mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$.

The theorem states the strong duality between ($\mathbf{C}$) and ($\mathbf{D}$). We defer the proof to Appendix B.2.1. The proof is based on min-max inequality and feasibility condition of ($\mathbf{D}$). Intuitively, we can view ($\mathbf{C}$), a functional optimization over f, as a linear programming (LP) problem over infinite number of variables $\{f(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^d\}$ so that the strong duality holds, which guarantees the tightness of DSRS in the primal space.

2.5.2 Dealing with Confidence Intervals

It is practically intractable to know the exact P_A and Q_A in Equation (2.8) by only querying the model's prediction for finite times. The common practice is using Monte-Carlo sampling, which gives confidence intervals of P_A and Q_A with a predefined confidence level $1 - \alpha$.

Suppose we have confidence intervals $[\underline{P_A}, \overline{P_A}]$ and $[\underline{Q_A}, \overline{Q_A}]$. To derive a sound certification, we need to certify that for *any* $P_A \in [\underline{P_A}, \overline{P_A}]$ and *any* $Q_A \in [\underline{Q_A}, \overline{Q_A}]$, $\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) > 0.5$. Given the infinite number of possible P_A and Q_A, the brute-force method is intractable. Here, *without* computing $\mathbf{C}_{\boldsymbol{\delta}}$, we show how to solve

$$(P_A, Q_A) = \underset{P'_A \in [\underline{P_A}, \overline{P_A}], Q'_A \in [\underline{Q_A}, \overline{Q_A}]}{\arg\min} \mathbf{C}_{\boldsymbol{\delta}}(P'_A, Q'_A). \tag{2.12}$$

If solved P_A and Q_A satisfy $\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) > 0.5$, then for any P_A and Q_A within the confidence intervals, we can certify the robustness against perturbation $\boldsymbol{\delta}$. We observe the following

two properties of $\mathbf{C}_{\boldsymbol{\delta}}$.

Proposition 2.2. $\mathbf{C}_{\boldsymbol{\delta}}(\cdot, \cdot)$ is convex in the feasible region.

Proposition 2.3. With respect to $x \in [0, 1]$, functions $x \mapsto \min_y \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ and $x \mapsto \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ are monotonically non-decreasing. Similarly, with respect to $y \in [0, 1]$, functions $y \mapsto \min_x \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ and $y \mapsto \arg\min_x \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ are monotonically non-decreasing.

These two propositions characterize the landscape of $\mathbf{C}_{\boldsymbol{\delta}}(\cdot, \cdot)$—convex and monotonically non-decreasing along both x and y axes. Thus, desired (P_A, Q_A) (location of minima within the bounded box) lies on the box boundary, and we only need to compute the location of boundary-line-sliced minima and compare it with box constraints to solve Equation (2.12). Formally, we propose an efficient algorithm (Algorithm 2.2 in Section 2.5.4) to solve (P_A, Q_A).

Theorem 2.4. If Equation (2.12) is feasible, the P_A and Q_A returned by Algorithm 2.2 solve Equation (2.12).

The above results are proved in Appendix B.2.2. On a high level, we prove Proposition 2.2 by definition; we prove Proposition 2.3 via a reduction to classical Neyman-Pearson-based certification and analysis of this reduced problem; and we prove Theorem 2.4 based on Propositions 2.2 and 2.3 along with exhaustive and nontrivial analyses of all possible cases.

2.5.3 Solving the Dual Problem

After the smoothed prediction confidences P_A and Q_A are determined from the confidence intervals, now we solve the dual problem $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$ as defined in Equation (2.11). We solve the problem based on the following theorem:

Theorem 2.5 (Numerical Integration for DSRS with Generalized Gaussian Smoothing)**.** In $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$, let $r = \|\boldsymbol{\delta}\|_2$, when $\mathcal{P} = \mathcal{N}^{\mathsf{g}}(k, \sigma)$ and $\mathcal{Q} = \mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$, let $\sigma' := \sqrt{d/(d-2k)}$ and let $\nu := \Gamma\mathrm{CDF}_{d/2-k}(T^2/(2\sigma'^2))$,

$$R(\lambda_1, \lambda_2) := \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon}) < \lambda_1 p(\boldsymbol{\epsilon}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{\epsilon}+\boldsymbol{\delta})] = \begin{cases} \mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_1(t), & \lambda_1 \leqslant 0 \\ \mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_1(t) + u_2(t), & \lambda_1 > 0 \end{cases} \tag{2.13}$$

$$\text{where} \quad u_1(t) = \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{\min\{T^2, 2\sigma'^2 k W(\frac{t}{k}e^{\frac{t}{k}}(\lambda_1+\nu\lambda_2)^{\frac{1}{k}})\}}{4r\sigma'\sqrt{2t}} - \frac{(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right), \tag{2.14}$$

$$u_2(t) = \max\left\{\mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{2\sigma'^2 k W(\frac{t}{k}e^{\frac{t}{k}}\lambda_1^{\frac{1}{k}})(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right) - \right. \tag{2.15}$$

$$-\text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2-(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right), 0\Bigg\},$$

$$P(\lambda_1,\lambda_2) := \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] = \mathop{\mathbb{E}}_{t\sim\Gamma(d/2-k,1)} \begin{cases} u_3(t,\lambda_1), & t \geqslant T^2/(2\sigma'^2) \\ u_3(t,\lambda_1+\nu\lambda_2), & t < T^2/(2\sigma'^2). \end{cases} \tag{2.16}$$

$$\text{where} \quad u_3(t,\lambda) = \text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{(r+\sigma'\sqrt{2t})^2}{4r\sigma'\sqrt{2t}} - \frac{2k\sigma'^2 W(\frac{t}{k}e^{\frac{t}{k}}\lambda^{-\frac{1}{k}})}{4r\sigma'\sqrt{2t}}\right), \tag{2.17}$$

$$Q(\lambda_1,\lambda_2) := \Pr_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] = \nu \mathop{\mathbb{E}}_{t\sim\Gamma(d/2-k,1)} u_3(t,\lambda_1+\nu\lambda_2)\cdot\mathbb{I}[t \leqslant T^2/(2\sigma'^2)]. \tag{2.18}$$

In above equations, $\Gamma(d/2-k,1)$ is gamma distribution and $\Gamma\text{CDF}_{d/2-k}$ is its CDF, $\text{BetaCDF}_{\frac{d-1}{2}}$ is the CDF of distribution $\text{Beta}(\frac{d-1}{2},\frac{d-1}{2})$, and W is the principal branch of Lambert W function.

When $\mathcal{P}$ is standard Gaussian and $\mathcal{Q}$ is truncated standard Gaussian, we derive similar expressions as detailed in Appendix B.3.1. We prove Theorem 2.5 in Appendix B.2.3. The proof extends the level-set sliced integration and results from [427]. With the theorem, we can rewrite the dual problem $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$ as

$$\max_{\lambda_1,\lambda_2\in\mathbb{R}} R(\lambda_1,\lambda_2) \,\text{s.t.}\, P(\lambda_1,\lambda_2) = P_A,\, Q(\lambda_1,\lambda_2) = Q_A, \tag{2.19}$$

Given concrete λ_1 and λ_2, from the theorem, these function values $P(\lambda_1,\lambda_2)$, $Q(\lambda_1,\lambda_2)$, and $R(\lambda_1,\lambda_2)$ can be easily computed with one-dimensional numerical integration using `SciPy` package.

Now, solving $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$ reduces to finding dual variables λ_1 and λ_2 such that $P(\lambda_1,\lambda_2) = P_A$ and $Q(\lambda_1,\lambda_2) = Q_A$. Generally, we find that there is only one unique feasible pair (λ_1,λ_2) for Equation (2.19), so finding out such a pair is sufficient. We prove the uniqueness and discuss how we deal with edge cases where multiple feasible pairs exist in Appendix B.2.4.

Normally, such solving process is expensive. However, we find a particularly efficient method to solve λ_1 and λ_2 and the algorithm description is in Algorithm 2.3 (in Section 2.5.4). At a high level, from Theorem 2.5, we observe that $Q(\lambda_1,\lambda_2)$ is determined only by the sum $(\lambda_1+\nu\lambda_2)$ and non-decreasing w.r.t. this sum. Therefore, we apply binary search to find

out $(\lambda_1 + \nu\lambda_2)$ that satisfies $Q(\lambda_1, \lambda_2) = Q_A$. Then, we observe that

$$P(\lambda_1, \lambda_2) - \frac{Q(\lambda_1, \lambda_2)}{\nu} = \overbrace{\mathop{\mathbb{E}}_{\mathrm{t}\sim\Gamma(d/2-k,1)} u_3(\mathrm{t}, \lambda_1) \cdot \mathbb{I}\left[\mathrm{t} \geqslant \frac{T^2}{2\sigma'^2}\right]}^{:=h(\lambda_1)}. \tag{2.20}$$

Thus, we need to find λ_1 such that $h(\lambda_1) = P_A - Q_A/\nu$. We observe that $h(\lambda_1)$ is non-decreasing w.r.t. λ_1, and we use binary search to solve λ_1. Combining with the value of $(\lambda_1 + \nu\lambda_2)$, we also obtain λ_2. We lastly leverage numerical integration to compute $R(\lambda_1, \lambda_2)$ following Theorem 2.5 to solve the dual problem $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$.

To this point, we have introduced the DSRS computational method.

2.5.4 Algorithm Pseudocode

Algorithm 2.1 is the pseudocode of the whole DSRS computational method.

Algorithm 2.1: DSRS computational method.

Data: clean input $\boldsymbol{x}_0$, base classifier F_0; distributions $\mathcal{P}$ and $\mathcal{Q}$; norm type p; confidence level α; numerical integration error bound Δ

Result: Certified radius r

1 Query prediction $y_0 \leftarrow \widetilde{F}_0^{\mathcal{P}}(\boldsymbol{x}_0)$;
2 Sample and estimate the intervals of smoothed confidence $[\underline{P_A}, \overline{P_A}]$ under $\mathcal{P}$ and $[\underline{Q_A}, \overline{Q_A}]$ under $\mathcal{Q}$ with confidence $(1-\alpha)$ following [77];
3 Initialize: $r_l \leftarrow 0, r_u \leftarrow r_{\max}$;
4 **while** $r_u - r_l >$ eps **do** /* Binary search on radius r */
5 $\quad r_m \leftarrow (r_l + r_u)/2$;
6 $\quad \boldsymbol{\delta} \leftarrow (r_m, 0, \ldots, 0)^\intercal$; /* for ℓ_2 certification with ℓ_2 symmetric $\mathcal{P}$ and $\mathcal{Q}$; for ℓ_∞ or ℓ_1, can be adjusted following [442] */
7 $\quad$ Determine $P_A \in [\underline{P_A}, \overline{P_A}]$ and $Q_A \in [\underline{Q_A}, \overline{Q_A}]$; /* See Section 2.5.2 and Algorithm 2.2 */
8 $\quad (\lambda_1, \lambda_2) \leftarrow$ DualBinarySearch(P_A, Q_A) ; /* See Section 2.5.3 and Algorithm 2.3 */
9 $\quad v \leftarrow R(\lambda_1, \lambda_2) - \Delta$; /* Using Theorem 2.5 */
10 $\quad$ **if** $v > 0.5$ **then**
11 $\quad\quad r_l \leftarrow r_m$
12 $\quad$ **else**
13 $\quad\quad r_u \leftarrow r_m$;
14 $\quad$ **end**
15 **end**
16 **return** r_l;

Algorithm 2.2 is a subroutine (Line 7) of Algorithm 2.1.

Algorithm 2.2: Determining P_A and Q_A from confidence intervals (see §2.5.2).

Data: Distributions $\mathcal{P}$ and $\mathcal{Q}$; $\boldsymbol{\delta}$; $[\underline{P_A}, \overline{P_A}]$ and $[\underline{Q_A}, \overline{Q_A}]$

Result: P_A and Q_A satisfying Equation (2.12)

1 Compute $\underline{q} \leftarrow \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A}, y)$;

2 **if** $\underline{q} > \underline{Q_A}$ **then**

3 | **return** $(\underline{P_A}, \min\{\underline{q}, \overline{Q_A}\})$

4 **else**

5 | Compute $\underline{p} \leftarrow \arg\min_x \mathbf{C}_{\boldsymbol{\delta}}(x, \underline{Q_A})$;

6 | **return** $(\max\{\min\{\underline{p}, \overline{P_A}\}, \underline{P_A}\}, \underline{Q_A})$;

Algorithm 2.3: DualBinarySearch for λ_1 and λ_2.

Data: Query access to $P(\cdot, \cdot)$ and $Q(\cdot, \cdot)$; P_A; Q_A; ν; precision parameter ϵ; numerical integration error bound Δ

Result: λ_1 and λ_2 satisfying constraints $P(\lambda_1, \lambda_2) = P_A, Q(\lambda_1, \lambda_2) = Q_A$ (see Equation (2.19))

1 $a^L \leftarrow 0$, $a^U \leftarrow M$; /* search for $a = \lambda_1 + \nu\lambda_2$, M is a large positive number */

2 **while** $a^U - a^L > \epsilon$ **do**

3 | $a^m \leftarrow (a^L + a^U)/2$;

4 | **if** $Q(a^m, 0) < Q_A$ **then**

5 | | $a^L \leftarrow a^m$

6 | **else**

7 | | $a^U \leftarrow a^m$

8 **end**

/* Following while-loop enlarges a^L and a^U until $[a^L, a^U]$ covers a^* such that $Q(a^*, 0) = Q_A$ under numerical integration error */

9 **while** $(Q(a^L, 0) + \Delta > Q_A)$ *or* $(Q(a^U, 0) - \Delta < Q_A)$ **do**

10 | $t \leftarrow a^U - a^L$;

11 | $a^L \leftarrow a^L - t/2$;

12 | $a^U \leftarrow a^U + t/2$;

13 **end**

14 $\lambda_1^L \leftarrow 0$, $\lambda_1^U \leftarrow M$; /* search for λ_1, M is a large positive number */

15 **while** $\lambda_1^U - \lambda_1^L > \epsilon$ **do**

16 | $\lambda_1^m \leftarrow (\lambda_1^L + \lambda_1^U)/2$;

17 | **if** $h(\lambda_1^m) - \Delta < P_A - Q_A/\nu$ **then**

18 | | $\lambda_1^L \leftarrow \lambda_1^m$

19 | **else**

20 | | $\lambda_1^U \leftarrow \lambda_1^m$

21 **end**

/* Following while-loop enlarges λ_1^L and λ_1^U until $[\lambda_1^L, \lambda_1^U]$ covers λ_1^* such that $h(\lambda_1^*) = P_A - Q_A/\nu$ under numerical integration error */

22 **while** $(h(\lambda_1^L) + \Delta > P_A - Q_A/\nu)$ *or* $(h(\lambda_1^U) - \Delta < P_A - Q_A/\nu)$ **do**

23 | $t \leftarrow \lambda_1^U - \lambda_1^L$;

24 | $\lambda_1^L \leftarrow \lambda_1^L - t/2$;

25 | $\lambda_1^U \leftarrow \lambda_1^U + t/2$;

26 **end**

27 **return** $(\lambda_1^L, (a^L - \lambda_1^U)/\nu)$; /* for soundness, choose the left endpoint of λ_1 and λ_2 range */

Note that Lines 1 and 5 of Algorithm 2.2 solve the constrained optimization with only

one constraint (either one of Equation (2.9b)), reducing to the well-studied and solvable Neyman-Pearson-based certification.

Note that we do not need to evaluate any value of $\mathbf{C}_{\boldsymbol{\delta}}$ in Algorithm 2.2. Although $\underline{q}$ and $\underline{p}$ in the algorithm are "$\arg\min_{x \text{ or } y}$" over $\mathbf{C}_{\boldsymbol{\delta}}$, the free choices of x or y leave $\mathbf{C}_{\boldsymbol{\delta}}$'s constrained optimization with only one constraint and then $\underline{q}$ and $\underline{p}$ can be solved by Neyman-Pearson instead of evaluating $\mathbf{C}_{\boldsymbol{\delta}}$ directly.

Algorithm 2.3 is the dual variable search algorithm described in Section 2.5.3, and it is a subroutine (Line 8) of Algorithm 2.1.

From Line 1 to 8, we conduct binary search for quantity $\lambda_1 + \nu\lambda_2$; from Line 14 to 21, we conduct binary search for quantity λ_1. Notice that our binary search interval is initialized to be the non-negative interval. This is because $Q(a^m, 0) = 0$ and $h(\lambda_1^m) = 0$ if a^m and λ_1^m are non-positive observed from Theorem 2.5.

Implementation details are in Appendix B.4.1.

2.5.5 Guaranteeing Numerical Soundness

As a practical certification method, we need to guarantee the certification soundness in the presence of numerical error. In DSRS, there are two sources of numerical error: numerical integration error when computing $P(\lambda_1, \lambda_2)$, $Q(\lambda_1, \lambda_2)$, and $R(\lambda_1, \lambda_2)$, and the finite precision of binary search on λ_1 and λ_2. For numerical integration, we notice that typical numerical integration packages such as `scipy` support setting an absolute error threshold Δ and raising warnings when such threshold cannot be reached. We set the absolute threshold $\Delta = 1.5 \times 10^{-8}$ in `scipy.integrad.quad` function, and abstain when the threshold cannot be reached (which never happens in our experimental evaluation). Then, when computing P, Q, and R, suppose the numerical value is v, we use the lower bound $(v - \Delta)$ and upper bound $(v + \Delta)$ in the corresponding context to guarantee the soundness. For the finite precision in binary search, we use the left endpoint or the right endpoint of the final binary search interval to guarantee soundness. For example, we use the left endpoint of λ_1 in R computation, and use the left endpoint of $(\lambda_1 + \nu\lambda_2)$ minus right endpoint of λ_1 to get the lower bound of λ_2 to use in R computation. Therefore, both λ_1 and λ_2 are underestimated and by the monotonicity of $R(\cdot, \cdot)$, the actual $R(\lambda_1, \lambda_2)$ would be underestimated to guarantee the soundness.

2.5.6 Complexity and Efficiency Analysis

Suppose the binary search precision is ϵ, and each numerical integration costs C time. First, the search of certified robust radius costs $O(\log(\sqrt{d}/\epsilon))$. For each searched radius, we first determine P_A and Q_A by running Neyman-Pearson-based certification, which has cost $O(\log(1/\epsilon)C)$. Then, solving dual variables takes two binary search rounds, which has cost $O(\log(1/\epsilon)C)$. The final one-time integration of $R(\lambda_1, \lambda_2)$ has cost $O(C)$. Thus, overall time complexity is $O(\log(\sqrt{d}/\epsilon)\log(1/\epsilon)C)$, which is the same as classical Neyman-Pearson certification and grows slowly (in logarithmic factor) w.r.t. input dimension d.

In practice, the certification time is on average 5 s to 10 s per sample across different datasets. For example, with $\sigma = 0.50$ as the smoothing variance parameter, the certification time, as an overhead over Neyman-Pearson-based certification, is 10.53 s, 4.53 s, and 3.21 s on MNIST, CIFAR-10, and ImageNet respectively. This overhead is almost negligible compared with the sampling time for estimating P_A and Q_A which is around 200 s on ImageNet and is the shared cost of all randomized smoothing certification methods. In summary, compared with standard Neyman-Pearson-based certification, the running time of DSRS is roughly the same.

2.5.7 Extensions

We mainly discussed DSRS computational method for generalized Gaussian $\mathcal{P}$ and truncated generalized Gaussian $\mathcal{Q}$ under ℓ_2 norm. Can we extend it to other settings? Indeed, DSRS is a general framework. In appendices, we show following extensions: (1) DSRS for generalized Gaussian with different variances as $\mathcal{P}$ and $\mathcal{Q}$ (in Appendix B.3.2); (2) DSRS for other ℓ_p norms (in Appendix B.3.3); and (3) DSRS that leverages other forms of additional information covering gradient magnitude information [215, 267] (in Appendix B.6).

2.6 EXPERIMENTAL EVALUATION

In this section, we systematically evaluate DSRS and demonstrate that it achieves tighter certification than the classical Neyman-Pearson-based certification against ℓ_2 perturbations on MNIST, CIFAR-10, and ImageNet. We focus on ℓ_2 certification because additive randomized smoothing is not optimal for other norms (e.g., ℓ_1 [214]) or the certification can be directly translated from ℓ_2 certification (e.g., ℓ_∞ [427] and semantic transformations [223]).

Table 2.2: Certified robust accuracy under different radii r with different certification approaches.

Dataset	Training Method	Certification Approach	Certified Accuracy under Radius r											
			0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00
MNIST	Gaussian Aug. [77]	Neyman-Pearson	**97.9%**	96.9%	94.6%	88.4%	78.7%	57.6%	41.0%	25.5%	13.6%	6.2%	2.1%	0.9%
		DSRS	**97.9%**	**97.0%**	**95.0%**	**89.8%**	**83.4%**	**61.6%**	**48.4%**	**34.1%**	**21.0%**	**10.6%**	**4.4%**	**1.2%**
	Consistency [165]	Neyman-Pearson	**98.4%**	**97.5%**	**96.0%**	92.3%	83.8%	67.5%	49.1%	35.6%	21.7%	10.4%	4.1%	1.9%
		DSRS	**98.4%**	**97.5%**	**96.0%**	**93.5%**	**87.1%**	**71.8%**	**55.8%**	**41.9%**	**31.4%**	**17.8%**	**8.6%**	**2.8%**
	SmoothMix [166]	Neyman-Pearson	**98.6%**	97.6%	96.5%	91.9%	85.1%	73.0%	51.4%	40.2%	31.5%	22.2%	12.2%	4.9%
		DSRS	**98.6%**	**97.7%**	**96.8%**	**93.4%**	**87.5%**	**76.6%**	**54.4%**	**46.2%**	**37.6%**	**29.2%**	**18.5%**	**7.2%**
CIFAR-10	Gaussian Aug. [77]	Neyman-Pearson	56.1%	41.3%	27.7%	18.9%	14.9%	10.2%	7.5%	4.1%	2.0%	0.7%	0.1%	**0.1%**
		DSRS	**57.4%**	**42.7%**	**30.6%**	**20.6%**	**16.1%**	**12.5%**	**8.4%**	**6.4%**	**3.5%**	**1.8%**	**0.7%**	**0.1%**
	Consistency [165]	Neyman-Pearson	61.8%	50.9%	38.0%	32.3%	23.8%	19.0%	16.4%	13.8%	11.2%	9.0%	7.1%	5.1%
		DSRS	**62.5%**	**52.5%**	**38.7%**	**35.2%**	**28.1%**	**20.9%**	**17.6%**	**15.3%**	**13.1%**	**10.9%**	**8.9%**	**6.5%**
	SmoothMix [166]	Neyman-Pearson	63.9%	53.3%	40.2%	34.2%	26.7%	20.4%	17.0%	13.9%	10.3%	7.8%	4.9%	2.3%
		DSRS	**64.7%**	**55.5%**	**42.1%**	**35.9%**	**29.4%**	**22.1%**	**18.7%**	**16.1%**	**13.2%**	**10.2%**	**7.1%**	**3.9%**
ImageNet	Guassian Aug. [77]	Neyman-Pearson	57.1%	47.0%	39.3%	33.2%	24.8%	21.4%	17.6%	13.7%	10.2%	7.8%	5.7%	3.6%
		DSRS	**58.4%**	**48.4%**	**41.4%**	**35.3%**	**28.8%**	**23.3%**	**21.3%**	**18.7%**	**14.2%**	**11.0%**	**9.0%**	**5.7%**
	Consistency [165]	Neyman-Pearson	59.8%	49.8%	43.3%	36.8%	31.4%	25.6%	22.1%	19.1%	16.1%	14.0%	10.6%	8.5%
		DSRS	**60.4%**	**52.4%**	**44.7%**	**39.3%**	**34.8%**	**28.1%**	**25.4%**	**22.6%**	**19.6%**	**17.4%**	**14.1%**	**10.4%**
	SmoothMix [166]	Neyman-Pearson	46.7%	38.2%	28.8%	24.6%	18.1%	14.2%	11.8%	10.1%	8.9%	7.2%	6.0%	4.6%
		DSRS	**47.4%**	**40.0%**	**30.3%**	**26.8%**	**21.6%**	**15.7%**	**14.0%**	**12.1%**	**9.9%**	**8.4%**	**7.2%**	**5.3%**

2.6.1 Experimental Setup

Smoothing Distributions. Following Theorem 2.2, we use generalized Gaussian $\mathcal{N}^{\mathrm{g}}(k, \sigma)$ as smoothing distribution $\mathcal{P}$ where $d/2 - 15 \leqslant k < d/2$. Specifically, we set k to be $d/2 - 12$ on MNIST, $d/2 - 6$ on CIFAR-10, and $d/2 - 4$ on ImageNet. We use three different σ's: 0.25, 0.50, and 1.00.

In terms of the additional smoothing distribution $\mathcal{Q}$, on MNIST and CIFAR-10, we empirically find that using generalized Gaussian with the same k but different variance yields tighter robustness certification, and therefore we choose σ_g to be 0.2, 0.4, and 0.8 corresponding to $\mathcal{P}$'s σ being 0.25, 0.50, and 1.0, respectively. On ImageNet, the concentration property (see Definition 2.3) is more pronounced (detail study in Appendix B.5.1) and thus we use truncated generalized Gaussian $\mathcal{N}^{\mathrm{g}}_{\mathsf{trunc}}(k, T, \sigma)$ as $\mathcal{Q}$. We apply a simple but effective algorithm as explained in Appendix B.4 to determine hyperparameter T in $\mathcal{N}^{\mathrm{g}}_{\mathsf{trunc}}(k, T, \sigma)$.

Models and Training. We consider three commonly-used or state-of-the-art training methods: Gaussian augmentation [77], Consistency [165], and SmoothMix [166]. We follow the default model architecture on each dataset respectively. We train the models with augmentation noise sampled from the corresponding generalized Gaussian smoothing distribution $\mathcal{P}$. More training details can be found in Appendix B.4.

Baselines. We consider the Neyman-Pearson-based certification method as the baseline. This certification is widely used and is the tightest given only prediction probability under $\mathcal{P}$ [77, 165, 223, 427]. We remark that although there are certification methods that leverage more information, to the best of our knowledge, they are not visibly better than the Neyman-Pearson-based method on ℓ_2 certification under practical sampling number (10^5). More comparisons in Appendix B.5.5 show DSRS is also better than these baselines.

For both baseline and DSRS, following the convention, the certification confidence is

$1 - \alpha = 99.9\%$, and we use 10^5 samples for estimating P_A and Q_A. Neyman-Pearson certification does not use the information from additional distribution, and all 10^5 samples are used to estimate the interval of P_A. In DSRS, we use 5×10^4 samples to estimate the interval of P_A with confidence $1 - \frac{\alpha}{2} = 99.95\%$ and the rest 5×10^4 samples for Q_A with the same confidence. By union bound, the whole certification confidence is 99.9%.

Metrics. We uniformly draw 1000 samples from the test set and report *certified robust accuracy* (under each ℓ_2 radius r) that is the fraction of samples that are both correctly classified and have certified robust radii larger than or equal to r. Under each radius r, we report the highest certified robust accuracy among the three variances $\sigma \in \{0.25, 0.50, 1.00\}$ following [77, 317]. We also report evaluation results with ACR metric [437] in Appendix B.5.3.

2.6.2 Evaluation Results

We show results in Table 2.2. The corresponding curves and separated tables for each variance σ are in Appendix B.5.3.

For *almost all* models and radii r, DSRS yields significantly higher certified accuracy. For example, for Gaussian augmented models, when $r = 2.0$, on MNIST the robust accuracy increases from 25.5% to 34.1% (+8.6%), on CIFAR-10 from 4.1% to 6.4% (+2.3%), and on ImageNet from 13.7% to 18.7% (+5.0%). On average, on MNIST the improvements are around 6% - 9%; on CIFAR-10, the improvements are around 1.5% - 3%; and on ImageNet the improvements are around 2% - 5%. Thus, DSRS can be used in conjunction with different training approaches and provides consistently tighter robustness certification.

The improvements in the robust radius are not as substantial as those in Figure 2.5(b) (which is around 2×). We investigate the reason in Appendix B.5.1. In summary, the model in Figure 2.5(b) is trained with standard Gaussian smoothing augmentation and smoothed with generalized Gaussian. The models in this section are trained with generalized Gaussian augmentation. Such training gives higher certified robustness, but in the meantime, gives more advantage to Neyman-Pearson-based certification. This finding implies that there may be a large space for exploring training approaches that favor DSRS certification since all existing training methods are designed for Neyman-Pearson-based certification. Nevertheless, even with these "unsuitable" training methods, DSRS still achieves significantly tighter robustness certification than the baseline.

On the other hand, all the above results are restricted to generalized Gaussian smoothing. We still observe that standard Gaussian smoothing combined with strong training methods [165, 317] and Neyman-Pearson certification (the SOTA setting) yields similar or slightly higher certified robust accuracy than generalized Gaussian smoothing even with

DSRS certification. Though DSRS with its suitable generalized Gaussian smoothing does not achieve SOTA certified robustness yet, given the theoretical advantages, we believe that with future tailored training approaches, DSRS with generalized Gaussian smoothing can bring strong certified robustness. More discussion is in Appendix B.6.3.

Ablation Studies. We present several ablation studies in the appendix. and verify: (1) Effectiveness of our simple heuristic for selecting hyperparameter for $\mathcal{Q}$: We propose a simple heuristic to select the hyperparameter T in smoothing distribution $\mathcal{Q} = \mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$. In Appendix B.5.2, we propose a gradient-based optimization method to select such $\mathcal{Q}$. We find that our simple heuristic has similar performance compared to the more complex optimization method but is more efficient. (2) Comparison of different types of $\mathcal{Q}$: by choosing different types of $\mathcal{Q}$ distributions (truncated Gaussian or Gaussian with different variance), DSRS has different performance as mentioned in Section 2.6.1. In Appendix B.5.4, we investigate the reason. In summary, when concentration property (see Definition 2.3) is better satisfied, using truncated Gaussian as $\mathcal{Q}$ is better; otherwise, using Gaussian with different variance is better.

2.7 SUMMARY

We propose a general DSRS framework that exploits information based on an additional smoothing distribution to tighten the robustness certification. We theoretically analyze and compare classical Neyman-Pearson and DSRS certification, showing that DSRS has the potential to break the curse of dimensionality of randomized smoothing.

CHAPTER 3: WHITE-BOX CERTIFICATION: GCP-CROWN

Neural network (NN) certification aims to formally prove or disprove certain properties (e.g., correctness and safety properties) of a NN under a certain set of inputs. These methods can provide worst-case performance guarantees of a NN, and have been applied to mission-critical applications that involve neural networks, such as automatic aircraft control [19, 181], learning-enabled cyber-physical systems [372], and NN based algorithms in an operating system [362].

The NN certification problem is generally NP-complete [180]. For piece-wise linear networks, it can be encoded as a mixed integer programming (MIP) [368] problem with the non-linear ReLU neurons described by binary variables. Thus, fundamentally, the NN certification problem can be solved using the branch and bound (BaB) [49] method similar to generic MIP solvers, by branching some binary variables and relaxing the rest into a convex problem such as linear programming (LP) to obtain bounds on the objective. Although early neural network verifiers relied on off-the-shelf CPU-based LP solvers [48, 245] for bounding in BaB, LP solvers do not scale well to large NNs. Thus, many recent verifiers are instead based on efficient and GPU-accelerated algorithms customized to NN certification, such as bound propagation methods [389, 425], Lagrangian decomposition methods [47, 91] and others [62, 286]. Bound propagation methods, presented in a few different formulations [104, 148, 343, 388, 406, 443], empower state-of-the-art NN verifiers such as α-β-CROWN [389, 425, 443] and VeriNet [20], and can achieve two to three orders of magnitudes speedup compared to solving the NN certification problem using an off-the-shelf solver directly [389], especially on large networks.

Despite the success of existing NN verifiers, we experimentally find that state-of-the-art NN verifiers may timeout on certain hard instances which a generic MIP solver can solve relatively quickly, sometimes even without branching. Compared to an MIP solver, a crucial factor missing in most scalable NN verifiers is the ability to efficiently generate and solve general cutting planes (or "cuts"). In generic MIP solvers, cutting planes are essential to strengthen the convex relaxation, so that much less branching is required. Advanced cutting planes are among the most important factors in modern MIP solvers [37]; they can strengthen the convex relaxation without removing any valid integer solution from the MIP formulation. In the setting of NN certification, cutting planes reflects complex intra-layer and inter-layer dependencies between multiple neurons, which cannot be easily captured by existing bound propagation methods with single neuron relaxations [318]. This motivates us to seek the combination of efficient bound propagation method with effective cutting planes to further

increase the power of NN verifiers.

A few key factors make the inclusion of *general* cutting planes in NN verifiers quite challenging. First, existing efficient bound propagation frameworks such as CROWN [443] and β-CROWN [389] cannot solve general cutting plane constraints that may involve variables across *different layers* in the MIP formulation. Particularly, these frameworks do not explicitly include the *integer variables* in the MIP formulation that are crucial when encoding many classical strong cutting planes, such as Gomory cuts and mixed integer rounding (MIR) cuts. Furthermore, although some existing works [273, 342, 367] enhanced the basic convex relaxation used in NN certification (such as the Planet relaxation [108]), these enhanced relaxations involve only one or a few neurons in a single layer or two adjacent layers, and are not general enough. In addition, an LP solver is often required to handle these additional cutting plane constraints [273], for which the efficient and GPU-accelerated bound propagation cannot be used, so the use of these tighter relaxations may not always bring improvements.

In this thesis, we achieve major progress in using general cutting planes in bound-propagation-based NN verifiers. To mitigate the challenge of efficiently solving general cuts, *our first contribution* is to generalize existing bound propagation methods to their most general form, enabling constraints involving variables from neurons of any layer as well as integer variables that encode the status of a ReLU neuron. This allows us to consider any cuts during bound propagation without relying on a slow LP solver, and opens up the opportunity for using advanced cutting plane techniques efficiently for the NN certification problem. *Our second contribution* involves combining a cutting-plane-focused, off-the-shelf MIP solver with our GPU-accelerated, branching-focused bound propagation method capable of handling general cuts. We entirely disable branching in the MIP solver and use it only for generating high-quality cutting planes not restricting to neurons within adjacent layers. Although an MIP solver often cannot certify large neural networks, we find that they can generate high-quality cutting planes within a short time, significantly helping bound propagation to achieve better bounds.

Our experiments show that general cutting planes can bring significant improvements to NN verifiers: we are the first verifier that *completely* solves *all* instances in the `oval20` benchmark, with an average time of *less than 5 seconds* per instance; on the even harder `oval21` benchmark in VNN-COMP 2021 [20], we can certify *twice as many instances* compared to the competition winner. We also outperform existing state-of-the-art bound-propagation-based methods including those using multi-neuron relaxations [115] (a limited form of cutting planes).

3.1 RELATED WORK

Cutting plane method is a classic technique to strengthen the convex relaxation of an integer programming problem. Generic cutting planes such as Gomory's cut [125, 126], Chvátal–Gomory cut [76], implied bound cut [151], lift-and-project [243], reformulation-linearization techniques [335] and mixed integer rounding cuts [256, 275] can be applied to almost any LP relaxed problems, and problem specific cutting planes such as Knapsack cut [87], Flow-cover cut [285] and Clique cut [174] require specific problem structures. Modern MIP solvers typically uses a branch-and-cut strategy, which tends to generate a large number of cuts before starting the next iteration of branching, and solve the LP relaxation of the MIP problem with cutting planes with an exact method such as the Simplex method. Our GCP-CROWN is a specialized solver for the NN certification problem, which can quickly obtain a lower bound of the LP relaxation with cutting planes specially for the NN certification problem.

The certification of piece-wise linear NNs can be formulated as a MIP problem, so early works [180, 181, 368] solve an integer or combinatorial formulation directly. For efficiency reasons, most recent works use a convex relaxation such as linear relaxation [108, 406] or semidefinite relaxation [89, 308]. Salman et al. [318] discussed the limitation of many convex relaxation based NN certification algorithms and coined the term "convex relaxation barrier", specifically for the popular single-neuron "Planet" relaxation [108]. Several works developed novel techniques to break this barrier. Singh et al. [342] added constraints that depends on the aggregation of multiple neurons, and these constraints were passed to an LP solver. Müller et al. [273] enhanced the multi-neuron formulation of [342] to obtain tighter relaxations. Anderson et al. [9] studied stronger convex relaxations for a ReLU neuron after an affine layer, and Tjandraatmadja et al. [367] constructed efficient algorithms based on this relaxation for incomplete certification. De Palma et al. [91] extended the formulation in [9] to a dual form and combined it with branch and bound to achieve completeness. Botoeva et al. [41] proposed specialized cuts by considering neuron dependencies and solve them using a MIP solver. Ferrari et al. [115] combined the multi-neuron relaxation [273] with branch and bound. Although these works can be seen as a special form of cutting planes, they mostly focused on enhancing the relaxation for several neurons within a single layer or two adjacent layers. GCP-CROWN can efficiently handle general cutting plane constraints with neurons from any layers in a bound propagation manner, and the cutting planes we find from a MIP solver can be seen as tighter convex relaxations encoding multi-neuron and multi-layer correlations.

NN Cerification Problem. We consider the white-box certification problem for an L-layer ReLU-based Neural Network (NN) with inputs $\hat{\boldsymbol{x}}^{(0)} := \boldsymbol{x} \in \mathbb{R}^{d_0}$, weights $\mathbf{W}^{(i)} \in \mathbb{R}^{d_i \times d_{i-1}}$, and biases $\boldsymbol{b}^{(i)} \in \mathbb{R}^{d_i}$ ($i \in [L]$). We can get the NN outputs $f(\boldsymbol{x}) = \boldsymbol{x}^{(L)} \in \mathbb{R}^{d_L}$ by sequentially propagating the input $\boldsymbol{x}$ through affine layers with $\boldsymbol{x}^{(i)} = \mathbf{W}^{(i)}\hat{\boldsymbol{x}}^{(i-1)} + \boldsymbol{b}^{(i)}$ and ReLU layer with $\hat{\boldsymbol{x}}^{(i)} = \text{ReLU}(\boldsymbol{x}^{(i)})$. We also let scalars $\hat{x}_j^{(i)}$ and $x_j^{(i)}$ denote the post-activation and pre-activation, respectively, of j-th ReLU neuron in i-th layer. We use $\mathbf{W}_{:,j}^{(i)}$ to denote the j-th column of $\mathbf{W}^{(i)}$.

Commonly, the input $\boldsymbol{x}$ is bounded within a perturbation set $\mathcal{C}$ (such as an ℓ_p norm ball) and the certification specification defines a property of the output $f(\boldsymbol{x})$ that should hold for any $\boldsymbol{x} \in \mathcal{C}$, e.g., whether the true label's logit $f_y(\boldsymbol{x})$ will be always larger than another label's logit $f_j(\boldsymbol{x})$, (i.e., checking if $f_y(\boldsymbol{x}) - f_j(\boldsymbol{x})$ is always positive). Since we can append the certification specification (such as a linear function on neural network output) as an additional layer of the network, canonically, the NN certification problem requires one to solve the following one-dimensional ($d_L = 1$) optimization objective on $f(\boldsymbol{x})$:

$$f^* = \min_{\boldsymbol{x}} f(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathcal{C} \tag{3.1}$$

with the relevant property defined to be proven if the optimal solution $f^* \geqslant 0$. Throughout this work, we consider the ℓ_∞ norm ball $\mathcal{C} := \{\boldsymbol{x} : \|\boldsymbol{x} - \boldsymbol{x}_0\|_\infty \leqslant \epsilon\}$ where $\boldsymbol{x}_0$ is a predefined constant (e.g., a clean input image), although it is possible to extend to other norms or specifications [303, 424].

MIP and LP Formulation for NN Certification. The mixed integer programming (MIP) formulation is the root of many NN certification algorithms. This formulation uses binary variables $\boldsymbol{z}$ to encode the non-linear ReLU neurons to make the non-convex optimization problem (3.1) tractable. Additionally, we assume that we know sound pre-activation bounds $\boldsymbol{l}^{(i)} \leqslant \boldsymbol{x}^{(i)} \leqslant \boldsymbol{u}^{(i)}$ for $\boldsymbol{x} \in \mathcal{C}$ which can be obtained via cheap bound propagation methods such as IBP [134] or CROWN [443]. Then ReLU neurons for each layer i can be classified into three classes [406], namely "active" ($\mathcal{I}^{+(i)}$), "inactive" ($\mathcal{I}^{-(i)}$) and "unstable" ($\mathcal{I}^{(i)}$) neurons, respectively:

$$\mathcal{I}^{+(i)} := \{j : \ l_j^{(i)} \geqslant 0\}; \quad \mathcal{I}^{-(i)} := \{j : \ u_j^{(i)} \leqslant 0\}; \quad \mathcal{I}^{(i)} := \{j : \ l_j^{(i)} \leqslant 0, \ u_j^{(i)} \geqslant 0\}. \tag{3.2}$$

Based on the definition of ReLU, activate and inactive neurons are linear functions, so only unstable neurons require binary encoding. The *MIP formulation* of (3.1) is:

$$f^* = \min_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{z}} f(\boldsymbol{x}) \qquad \text{s.t. } f(\boldsymbol{x}) = \boldsymbol{x}^{(L)}; \quad \hat{\boldsymbol{x}}^{(0)} = \boldsymbol{x}; \quad \boldsymbol{x} \in \mathcal{C}; \tag{3.3}$$

$$\boldsymbol{x}^{(i)} = \mathbf{W}^{(i)}\hat{\boldsymbol{x}}^{(i-1)} + \boldsymbol{b}^{(i)}; \quad i \in [L], \tag{3.4}$$

$$\hat{x}_j^{(i)} \geqslant 0; \quad j \in \mathcal{I}^{(i)}, i \in [L-1] \tag{3.5}$$

$$\hat{x}_j^{(i)} \geqslant x_j^{(i)}; \quad j \in \mathcal{I}^{(i)}, i \in [L-1] \tag{3.6}$$

$$\hat{x}_j^{(i)} \leqslant u_j^{(i)} z_j^{(i)}; \quad j \in \mathcal{I}^{(i)}, i \in [L-1] \tag{3.7}$$

$$\hat{x}_j^{(i)} \leqslant x_j^{(i)} - l_j^{(i)}(1 - z_j^{(i)}); \quad j \in \mathcal{I}^{(i)}, i \in [L-1] \tag{3.8}$$

$$z_j^{(i)} \in \{0, 1\}; \quad j \in \mathcal{I}^{(i)}, i \in [L-1] \tag{3.9}$$

$$\hat{x}_j^{(i)} = x_j^{(i)}; \quad j \in \mathcal{I}^{+(i)}, i \in [L-1] \tag{3.10}$$

$$\hat{x}_j^{(i)} = 0; \quad j \in \mathcal{I}^{-(i)}, i \in [L-1]. \tag{3.11}$$

Since a MIP problem is slow or intractable to solve, it is commonly relaxed. When the integer variables are relaxed to continuous ones, we obtain the *LP relaxation* of NN certification problem:

$$f^*_{\text{LP}} = \min_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{z}} f(\boldsymbol{x})$$
$$\text{s.t. } (3.4), (3.3), (3.5), (3.6), (3.7), (3.8), (3.10), (3.11), \quad 0 \leqslant z_j^{(i)} \leqslant 1; \quad j \in \mathcal{I}^{(i)},\ i \in [L-1]. \tag{3.12}$$

The ReLU constraints involving z is often projected out, leading to the well-known *Planet relaxation* used in many NN verifiers, replacing (3.7), (3.8) and (3.9) with a single constraint to get an equivalent LP:

$$f^*_{\text{LP}} = \min_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{z}} f(\boldsymbol{x})$$
$$\text{s.t. } (3.4), (3.3), (3.5), (3.6), (3.10), (3.11), \quad \hat{x}_j^{(i)} \leqslant \frac{u_j^{(i)}}{u_j^{(i)} - l_j^{(i)}}(x_j^{(i)} - l_j^{(i)});\ j \in \mathcal{I}^{(i)},\ i \in [L-1]. \tag{3.13}$$

Due to the relaxations, the objective of the LP formulation is always a lower bound of the MIP formulation: $f^*_{\text{LP}} \leqslant f^*$. A verifier using this formulation is incomplete: if $f^*_{\text{LP}} \geqslant 0$, then $f^* \geqslant 0$ and the property is verified; otherwise, we cannot conclude the sign of f^* so the verifier must return "unknown". Branch and bound can be used to improve the lower bound

and achieve completeness [49, 389]. However, in this chapter, we work on an orthogonal direction of strengthening the LP formulation by adding cutting planes to obtain larger bounds.

Bound Propagation Methods Instead of solving the LP formulation directly using a LP solver, bound propagation methods aims to quickly give a lower bound for f^*_{LP}. For example, CROWN [443] and β-CROWN [389] propagate a sound linear lower bound backwards for $f_L(\boldsymbol{x})$ with respect to each intermediate layer. For example, suppose we know

$$\min_{\boldsymbol{x}\in\mathcal{C}} f_L(\boldsymbol{x}) \geqslant \min_{\boldsymbol{x}\in\mathcal{C}} \mathbf{a}^{(i)\top}\hat{\boldsymbol{x}}^{(i)} + c^{(i)}. \tag{3.14}$$

With $i = L-1$ the above is trivially hold with $\mathbf{a}^{(L-1)} = \mathbf{W}^{(L)}$, $c^{(L-1)} = \boldsymbol{b}^{(L)}$. In bound propagation methods, a propagation rule propagates an inequality (3.14) through a previous layer $\hat{\boldsymbol{x}}^{(i)} := \text{ReLU}(\mathbf{W}^{(i)}\hat{\boldsymbol{x}}^{(i-1)} + \boldsymbol{b}^{(i)})$ to obtain a *sound* inequality with respect to $\hat{\boldsymbol{x}}^{(i-1)}$:

$$\min_{\boldsymbol{x}\in\mathcal{C}} f_L(\boldsymbol{x}) \geqslant \min_{\boldsymbol{x}\in\mathcal{C}} \mathbf{a}^{(i-1)\top}\hat{\boldsymbol{x}}^{(i-1)} + c^{(i-1)}. \tag{3.15}$$

Here $\mathbf{a}^{(i-1)}$, $c^{(i-1)}$ can be calculated in close-form via $\mathbf{a}^{(i)}$, $c^{(i)}$, $\mathbf{W}^{(i)}$, $\boldsymbol{b}^{(i)}$, $\boldsymbol{l}^{(i)}$ and $\boldsymbol{u}^{(i)}$ such that the bound still holds (see Lemma 1 in [389]). Applying the procedure repeatedly will eventually reach the input layer:

$$\min_{\boldsymbol{x}\in\mathcal{C}} f_L(\boldsymbol{x}) \geqslant \min_{\boldsymbol{x}\in\mathcal{C}} \mathbf{a}^{(0)\top}\boldsymbol{x} + c^{(0)}. \tag{3.16}$$

The minimization on linear layer can be solved easily when $\mathcal{C}$ is a ℓ_p norm ball to obtain a valid lower bound of f^*. Since the bounds propagate layer-by-layer, this process can be implemented efficiently on GPUs [424] without relying on a slow LP solver, which greatly improves the scalability and solving time. Additionally, it is often used to obtain intermediate layer bounds $\boldsymbol{l}^{(i)}$ and and $\boldsymbol{u}^{(i)}$ required for the MIP formulation (3.7)(3.8), by treating each $x_j^{(i)}$ as the output neuron. The bound propagation rule can either be derived in primal space [443], dual space [406] or abstract interpretations [343]. In Section 3.3.1, we will discuss the our bound propagation procedure with general cutting plane constraints.

3.3 NEURAL NETWORK CERTIFICATION WITH GENERAL CUTTING PLANES

3.3.1 GCP-CROWN: General Cutting Planes in Bound Propagation

In this section, we generalize existing bound propagation method to handle general cutting plane constraints. Our goal is to derive a bound propagation rule similar to CROWN and β-CROWN discussed in Section 3.2, however considering additional constraints among any variables within the LP relaxation. To achieve this, we first derive the dual problem of the LP; inspired by the dual formulation, we derive the bound prorogation rule in a layer by layer manner that takes all cutting plane constraints into consideration. The derivation process is inspired by [318, 406] and [389].

LP Relaxation with Cutting Planes. In this section, we derive the bound propagation procedure under the presence of general cutting plane constraints. A cutting plane is a constraint involving any variables $\boldsymbol{x}^{(i)}$ (pre-activation), $\hat{\boldsymbol{x}}^{(i)}$ (post-activation), $\boldsymbol{z}^{(i)}$ (ReLU indicators) from any layer i:

$$\sum_{i=1}^{L-1} \left(\boldsymbol{h}^{(i)\top} \boldsymbol{x}^{(i)} + \boldsymbol{g}^{(i)\top} \hat{\boldsymbol{x}}^{(i)} + \boldsymbol{q}^{(i)\top} \boldsymbol{z}^{(i)} \right) \leqslant d. \tag{3.17}$$

Here $\boldsymbol{h}^{(i)}$, $\boldsymbol{g}^{(i)}$ and $\boldsymbol{q}^{(i)}$ are coefficients for this cut constraint. The difference between a valid cutting plane and an arbitrary constraint is that a valid cutting plane should not remove any valid integer solution from the MIP formulation. Our new bound propagation procedure can work for any constraints, although in this work we focus on studying the impacts of cutting planes. When there are N cutting planes, we write them in a matrix form:

$$\sum_{i=1}^{L-1} \left(\boldsymbol{H}^{(i)} \boldsymbol{x}^{(i)} + \boldsymbol{G}^{(i)} \hat{\boldsymbol{x}}^{(i)} + \boldsymbol{Q}^{(i)} \boldsymbol{z}^{(i)} \right) \leqslant \boldsymbol{d} \tag{3.18}$$

where $\boldsymbol{H}^{(i)}, \boldsymbol{G}^{(i)}, \boldsymbol{Q}^{(i)} \in \mathbb{R}^{N \times d_i}$. The LP relaxation with all cutting planes is:

$$f^*_{\text{LP-cut}} = \min_{\boldsymbol{x}, \hat{\boldsymbol{x}}, \boldsymbol{z}} f(x)$$

$$\text{s.t. } (3.4), (3.3), (3.5), (3.6), (3.7), (3.8), (3.10), (3.11), \quad 0 \leqslant z_j^{(i)} \leqslant 1; \quad j \in \mathcal{I}^{(i)},\ i \in [L-1],$$

$$\sum_{i=1}^{L-1} \left(\boldsymbol{H}^{(i)} \boldsymbol{x}^{(i)} + \boldsymbol{G}^{(i)} \hat{\boldsymbol{x}}^{(i)} + \boldsymbol{Q}^{(i)} \boldsymbol{z}^{(i)} \right) \leqslant \boldsymbol{d}. \tag{3.19}$$

Since an additional constraint is added, $f^*_{\text{LP-cut}} \geqslant f^*_{\text{LP}}$ and we get closer to f^*. Unlike the original LP where each constraint only contains variables from two consecutive layers, our general cutting plane constraint may involve any variable from any layer in a single constraint.

Dual Problem with Cutting Planes. We first show the dual problem for the above LP. The dual problem we consider here is different from existing works in two ways: first, we have constraints with integer variables to support potential cutting planes on $\boldsymbol{z}$. Additionally, we have cutting planes constraints that may involve variables in *any* layer, so the dual in previous works such as [406] cannot be directly reused. Our dual problem is given below (derivation details in Appendix C.1):

$$
\begin{aligned}
f^*_{\text{LP-cut}} = & \max_{\substack{\boldsymbol{\nu},\boldsymbol{\mu}\geqslant 0,\boldsymbol{\tau}\geqslant 0\\ \boldsymbol{\gamma}\geqslant 0,\boldsymbol{\pi}\geqslant 0,\boldsymbol{\beta}\geqslant 0}} -\epsilon\|\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}\boldsymbol{x}_0\|_1 - \boldsymbol{\beta}^\top \boldsymbol{d} - \sum_{i=1}^{L}\boldsymbol{\nu}^{(i)\top}\boldsymbol{b}^{(i)} \\
& + \sum_{i=1}^{L-1}\sum_{j\in\mathcal{I}^{(i)}}\left[\pi_j^{(i)} l_j^{(i)} - \text{ReLU}(u_j^{(i)}\gamma_j^{(i)} + l_j^{(i)}\pi_j^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{Q}^{(i)}_{:,j})\right] \\
\text{s.t. } & \boldsymbol{\nu}^{(L)} = -1; \text{ and for each } i \in [L-1]: \\
& \nu_j^{(i)} = \boldsymbol{\nu}^{(i+1)\top}\mathbf{W}^{(i+1)}_{:,j} - \boldsymbol{\beta}^\top(\boldsymbol{H}^{(i)}_{:,j} + \boldsymbol{G}^{(i)}_{:,j});\ j \in \mathcal{I}^{+(i)} \\
& \nu_j^{(i)} = -\boldsymbol{\beta}^\top \boldsymbol{H}^{(i)}_{:,j};\ j \in \mathcal{I}^{-(i)} \\
& \quad \text{and for each } j \in \mathcal{I}^{(i)} \text{ the two equalities below hold:} \\
& \nu_j^{(i)} = \pi_j^{(i)} - \tau_j^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{H}^{(i)}_{:,j}; \quad \left(\pi_j^{(i)} + \gamma_j^{(i)}\right) - \left(\mu_j^{(i)} + \tau_j^{(i)}\right) = \boldsymbol{\nu}^{(i+1)\top}\mathbf{W}^{(i+1)}_{:,j} - \boldsymbol{\beta}^\top \boldsymbol{G}^{(i)}_{:,j}.
\end{aligned}
\tag{3.20}
$$

Instead of solving the dual problem exactly, we use it to obtain a lower bound of $f^*_{\text{LP-cut}}$. Intuitively, due to the definition of dual problem, any valid setting of dual variables leads to a lower bound of $f^*_{\text{LP-cut}}$. Informally, starting from $\boldsymbol{\nu}^{(L)} = -1$, by applying the constraints in this dual formulation, we can compute $\boldsymbol{\nu}^{(L-1)}, \boldsymbol{\nu}^{(L-2)}, \cdots$ until $\boldsymbol{\nu}^{(1)}$. The final objective is a function of $\boldsymbol{\nu}^{(i)}$, $i \in [N]$ and other dual variables. Precisely, our GCP-CROWN bound propagation procedure with general cutting plane constraint is presented in the theorem below (proof in Appendix C.1):

Theorem 3.1 (Bound propagation with general cutting planes)**.** Given any optimizable parameters $0 \leqslant \alpha_j^{(i)} \leqslant 1$ and $\boldsymbol{\beta} \geqslant 0$, $f^*_{\text{LP-cut}}$ is lower bounded by the following objective

function, ${\pi_j^{(i)}}^*$ is a function of $\boldsymbol{Q}_{:,j}^{(i)}$:

$$g(\boldsymbol{\alpha}, \boldsymbol{\beta}) = -\epsilon \|\boldsymbol{\nu}^{(1)\top} \mathbf{W}^{(1)} \boldsymbol{x}_0\|_1 - \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top} \boldsymbol{b}^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{d} + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{(i)}} h_j^{(i)}(\boldsymbol{\beta}) \tag{3.21}$$

where variables $\boldsymbol{\nu}^{(i)}$ are obtained by propagating $\boldsymbol{\nu}^{(L)} = -1$ throughout all $i \in [L-1]$:

$$\nu_j^{(i)} = \boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top (\boldsymbol{H}_{:,j}^{(i)} + \boldsymbol{G}_{:,j}^{(i)}), \ \ j \in \mathcal{I}^{+(i)} \tag{3.22}$$

$$\nu_j^{(i)} = -\boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)}, \ \ j \in \mathcal{I}^{-(i)} \tag{3.23}$$

$$\nu_j^{(i)} = {\pi_j^{(i)}}^* - \alpha_j^{(i)} [\hat{\nu}_j^{(i)}]_- - \boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)}, \ \ j \in \mathcal{I}^{(i)}. \tag{3.24}$$

Here $\hat{\nu}_j^{(i)}$, ${\pi_j^{(i)}}^*$ and $h_j^{(i)}(\boldsymbol{\beta})$ are defined for each unstable neuron $j \in \mathcal{I}^{(i)}$.

$$\hat{\nu}_j^{(i)} := \boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top \boldsymbol{G}_{:,j}^{(i)}, \tag{3.25}$$

$${\pi_j^{(i)}}^* = \max\left(\min\left(\frac{u_j^{(i)}[\hat{\nu}_j^{(i)}]_+ + \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}}{u_j^{(i)} - l_j^{(i)}}, [\hat{\nu}_j^{(i)}]_+\right), 0\right), {\pi_j^{(i)}}^* \text{ is a function of } \boldsymbol{Q}_{:,j}^{(i)}, \tag{3.26}$$

$$h_j^{(i)}(\boldsymbol{\beta}) = \begin{cases} l_j^{(i)}, {\pi_j^{(i)}}^* & \text{if} \ \ l_j^{(i)}[\hat{\nu}_j^{(i)}]_+ \leqslant \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \leqslant u_j^{(i)}[\hat{\nu}_j^{(i)}]_+ \\ 0, & \text{if} \ \ \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \geqslant u_j^{(i)}[\hat{\nu}_j^{(i)}]_+ \\ \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}. & \text{if} \ \ \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \leqslant l_j^{(i)}[\hat{\nu}_j^{(i)}]_+ \end{cases} \tag{3.27}$$

Based on Theorem 3.1, to obtain an lower bound of $f^*_{\text{LP-cut}}$, we start with any valid setting of $0 \leqslant \boldsymbol{\alpha} \leqslant 1$ and $\boldsymbol{\beta} \geqslant 0$ and $\boldsymbol{\nu}^{(L)} = -1$. According to the bound propagation rule, we can compute each $\boldsymbol{\nu}^{(i)}$, $i \in [L-1]$, in a layer by layer manner. Then objective $g(\boldsymbol{\alpha}, \boldsymbol{\beta})$ can be evaluated based on all $\boldsymbol{\nu}^{(i)}$ to give an lower bound of $f^*_{\text{LP-cut}}$. Since any valid setting of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ lead to a valid lower bound, we can optimize $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ using gradient ascent in a similar manner as in [389, 425] to tighten this lower bound. The entire procedure can also run on GPU for great acceleration.

Connection to Convex Outer Adversarial Polytope. In convex outer adversarial polytope [406], a bound propagation rule was developed in a similar manner in the dual space without considering cutting plane constraints, and is a special case of ours. We denote their bound propagation objective function as g_{WK} which also contains optimizable parameters $\boldsymbol{\alpha}_{\text{WK}}$.

Proposition 3.1. Given the same input $\boldsymbol{x}$, perturbation set $\mathcal{C}$, network weights, and N cutting plane constraints,

$$\max_{\boldsymbol{\alpha},\boldsymbol{\beta}} g(\boldsymbol{\alpha},\boldsymbol{\beta}) \geqslant \max_{\boldsymbol{\alpha}_{\text{WK}}} g_{\text{WK}}(\boldsymbol{\alpha}_{\text{WK}}). \tag{3.28}$$

Proof. In Theorem 3.1, when all $\boldsymbol{\beta}$ are set to 0, then $\pi_j^{(i)*} = \frac{u_j^{(i)}[\hat{\nu}_j^{(i)}]_+}{u_j^{(i)}-l_j^{(i)}}$ and $h_j^{(i)}(\boldsymbol{\beta}) = \pi_j^{(i)*} l_j^{(i)}$, we recover exactly the same bound propagation equations as in [406]. However, since we allow the addition of cutting plane methods and we can maximize over the additional parameter $\boldsymbol{\beta}$, the objective given by our bound propagation is always at least as good as g_{WK}. QED.

Connection to CROWN-like Bound Propagation Methods. CROWN [443] and α-CROWN [425] use the same bound propagation rule as [406] so Proposition 3.1 also applies, although they were derived from primal space without explicitly formulating the problem as a LP. Salman et al. [318] showed that many other bound propagation methods [343, 387] are equivalent to or weaker than [406]. Recently, Wang et al. [389] extend CROWN to β-CROWN to handle split constraints (e.g., $x_j^{(i)} \geqslant 0$). It can be seen a special case as GCP-CROWN where all $\boldsymbol{H}$, $\boldsymbol{G}$ and $\boldsymbol{Q}$ matrices are zeros except:

$$\boldsymbol{H}_{j,j}^{(i)} = 1, j \in \mathcal{I}^{-(i)} \qquad \text{for } x_j^{(i)} \leqslant 0 \text{ split;} \qquad \boldsymbol{H}_{j,j}^{(i)} = -1, j \in \mathcal{I}^{+(i)} \qquad \text{for } x_j^{(i)} \geqslant 0 \text{ split.} \tag{3.29}$$

In addition, Ferrari et al. [115] encode multi-neuron relaxations using sparse $\boldsymbol{H}^{(i)}$ and $\boldsymbol{G}^{(i)}$ and each cut contains a small number of neurons involving $\boldsymbol{x}^{(i)}$ and $\hat{\boldsymbol{x}}^{(i)}$ for the same layer i. Wang et al. [389] derive bound propagation rules from both the dual LP and the primal space with a Lagrangian without LP. However, in our case, it is not intuitive to derive bound propagation without LP due to the potential cutting planes on relaxed integer variables $\boldsymbol{z}$, which do not appear without the explicit LP formulation. Furthermore, although we derived cutting planes for bound propagation methods, it is technically also possible to derive them using other bounding frameworks such as Lagrangian decomposition [47].

3.3.2 Branch-and-bound with GCP-CROWN and MIP Solver Generated Cuts

To build a complete NN verifier, we follow the popular branch-and-bound (BaB) procedure [48, 49] in state-of-the-art NN verifiers with GPU accelerated bound propagation method [47, 286, 389, 425], and our GCP-CROWN is used as the bounding procedure in BaB. We refer the readers to Appendix C.2 for a more detailed background on branch-

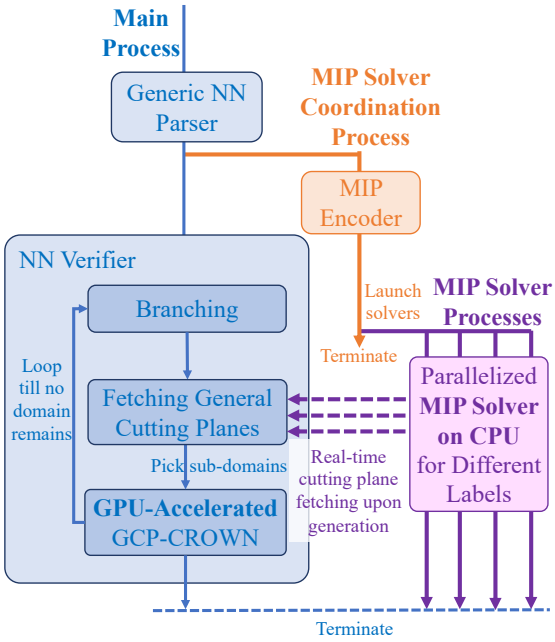


Figure 3.1: Overview of our cutting-plane-enhanced, fully-parallelized NN verifier.

and-bound. Having the efficient bound propagation procedure with general cutting plane constraints, we now need to find a good set of general cutting planes $\boldsymbol{H}^{(i)}, \boldsymbol{G}^{(i)}, \boldsymbol{Q}^{(i)}$ to accelerate NN certification. Since GCP-CROWN can adopt any cutting planes, to fully exploit its power, we propose to use off-the-shelf MIP solvers to generate cutting planes and create an NN verifier combining GPU-accelerated bound propagation with strong cuts generated by a MIP solver. We make the bound propagation on GPU and the MIP solver on CPU run in parallel with cuts added on-the-fly, so the original strong performance of bound-propagation-based NN verifier will never be affected by a potentially slow MIP solver. This allows us to make full use of available computing resource (GPU + CPU). The architecture of our verifier is shown in Figure 3.1.

MIP Solvers for Cutting Plane Generation. Generic MIP solvers such as `cplex` [160] and `gurobi` [137] also apply a branch-and-bound strategy, conceptually similar to state-of-the-art NN verifiers. They often tend to be slower than specialized NN verifiers because MIP solvers rely on slower bounding procedures (e.g., Simplex or barrier method) and cannot apply an GPU-accelerated method such as bound propagation or Lagrangian decomposition. However, we still find that MIP solvers are a strong baseline when *combined with tight intermediate layer bounds*. For example, in the `oval21` benchmark in Table 3.2, α-CROWN+MIP (MIP solver combined with tight intermediate layer bound computed by α-CROWN) is able

to solve 4 more instances compared to all other tools in the competition. Our investigation found that α-CROWN+MIP explores much less branches than other state-of-the-art branch-and-bound based NN verifiers, however before branching starts, the MIP solver produces very effective cutting planes that can often certify an instance with little branching. MIP solvers is able to discover sophisticated cutting planes involving several NN layers reflecting complex correlations among neurons, while existing NN verifiers only exploit specific forms of constraints within same layer or between adjacent layers [273, 342, 367]. This motivates us to combine the strong cutting planes generated by MIP solvers with GPU-accelerated bound-propagation-based BaB.

In this work, we use `cplex` as the MIP solver for cutting plane generation since `gurobi` cannot export cuts. We entirely disable all branching features in `cplex` and make it focus on finding cutting planes only. These cutting planes are generated only for the root node of the BaB search tree so they are sound for any subdomains with neuron splits in BaB. We conduct branching using our generalized bound propagation procedure in Section (3.3.1) with the cutting planes generated by `cplex`.

Fully-parallelized NN Verifier Design. We design our verifier as shown in Figure 3.1. After parsing the NN under certification, we launch a separate process to encode the NN certification problem as MIP problem and start multiple MIP solvers, one for each target label to be verified. At the same time, the main NN verifier process executes branch-and-bound without waiting for the MIP solver process. In each iteration of branch and bound, we query the MIP solving processes and fetch any newly generated cutting planes. If any cutting planes are produced, they are added as $\boldsymbol{H}^{(i)}, \boldsymbol{G}^{(i)}, \boldsymbol{Q}^{(i)}$ in GCP-CROWN and tighten the bounds for subsequent branching and bounding. If no cutting planes are produced, GCP-CROWN reduces to β-CROWN [389]. Since our verifier is based on strengthening the bounds in β-CROWN with sound cutting planes, it is also sound and complete.

Adjustments to Existing Branch and Bound Method. We implement GCP-CROWN into the α-β-CROWN verifier [389, 424, 443], the winning verifier in VNN-COMP 2021, 2022, and 2023 [20, 272], as the backbone for BaB with bound propagation. To better exploit the power of cutting planes under a fully parallel and asynchronous design, we made a key changes to the BaB procedure. When the number of BaB subdomains are greater than batch size, we rank the subdomains by their lower bounds and choose the *easiest domains* with largest lower bounds first to certify with GCP-CROWN, unlike most existing verifiers which solve the worst domains first. We use such an order because the MIP solver generates cutting planes incrementally. Solving these easier subdomains tend to require no or fewer

Table 3.1: Average runtime and average number of branches on `oval20` benchmarks with 100 properties per model. Timeout is set to 3,600 seconds (consistent with other literature results). GCP-CROWN is the only method that can completely solve *all* instances (0% timeout) and the average time per-instance is less than 5 seconds on all three networks.

	CIFAR-10 Base			CIFAR-10 Wide			CIFAR-10 Deep		
Method	time(s)	branches	%timeout	time(s)	branches	%timeout	time(s)	branches	%timeout
MIPplanet [108]	2849.69	-	68.00	2417.53	-	46.00	2302.25	-	40.00
BaBSR [48]	2367.78	1020.55	36.00	2871.14	812.65	49.00	2750.75	401.28	39.00
GNN-online [245]	1794.85	565.13	33.00	1367.38	372.74	15.00	1055.33	131.85	4.00
BDD+ BaBSR [47]	807.91	195480.14	20.00	505.65	74203.11	10.00	266.28	12722.74	4.00
Fast-and-Complete [425]	695.01	119522.65	17.00	495.88	80519.85	9.00	105.64	2455.11	1.00
OVAL (BDD+ GNN)*[47, 245]	662.17	67938.38	16.00	280.38	17895.94	6.00	94.69	1990.34	1.00
A.set BaBSR [286]	381.78	12004.60	7.00	165.91	2233.10	3.00	190.28	2491.55	2.00
BigM+A.set BaBSR [286]	390.44	11938.75	7.00	172.65	4050.59	3.00	177.22	3275.25	2.00
BaDNB (BDD+ FSB)[91]	309.29	38239.04	7.00	165.53	11214.44	4.00	10.50	368.16	**0.00**
ERAN*[340, 341, 342, 343]	805.94	-	5.00	632.20	-	9.00	545.72	-	**0.00**
β-CROWN [389]	118.23	208018.21	3.00	78.32	116912.57	2.00	5.69	41.12	**0.00**
α-CROWN+MIP†	335.50	8523.37	3.00	203.87	2029.60	**0.00**	76.90	1364.24	**0.00**
GCP-CROWN with MIP cuts	**4.07**	2580.53	**0.00**	**3.02**	2095.18	**0.00**	**3.87**	110.92	**0.00**

* Results from VNN-COMP 2020 report [233].

† A new baseline proposed and evaluated in this work, not presented in previous publications.

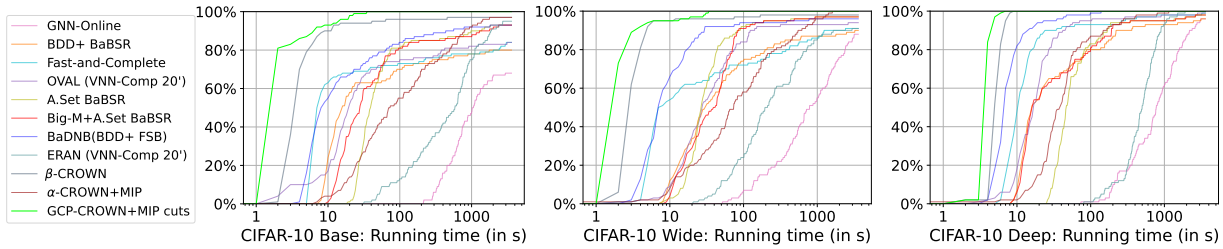


Figure 3.2: Percentage of solved properties on the `oval20` benchmark vs. running time (timeout 1 hour).

cutting planes, so we solve them at earlier stages where cutting planes have not been generated or are relatively weak. On the other hand, if we split worst subdomains first, the number of subdomains will grow quickly, and it can take a long time to certify these domains when stronger cuts become available later. Under a similar rationale, when certifying a multi-class model and BaB needs to certify each target class label one by one, we start BaB from the easiest label first (α-CROWN bound closest to 0), allowing the MIP solver to run longer for harder labels, generating stronger cuts for harder labels.

3.4 EXPERIMENTS

We now evaluate our verifier, GCP-CROWN with MIP cuts, on a few popular certification benchmarks. Since our verifier uses a MIP solver in our pipeline, we also include a new baseline, α-CROWN+MIP, which uses `gurobi` as the MIP solver with the tightest possible

Table 3.2: VNN-COMP 2021 benchmarks: `oval21` and `cifar10-resnet`. Results marked with VNN-COMP are from publicly available benchmark data on VNN-COMP 2021 Github. "-" indicates unsupported model.

	`oval21` (30 properties; PGD upper bound 27)			`cifar10-resnet` (72 properties)		
Method	time(s)	# verified	%timeout	time(s)	# verified	%timeout
nnenum* [18, 19]	630.06	2	86.66	-	-	-
Marabou* [181]	429.13	5	73.33	157.70	39	45.83
ERAN*[271, 273]	233.84	6	70.00	129.48	43	40.28
OVAL* [91, 286]	393.14	11	53.33	-	-	-
VeriNet* [148, 149]	414.61	11	53.33	105.91	48	33.33
α,β-CROWN* [389, 425, 443]	395.32	11	53.33	99.87	58	19.44
MN-BaB [115]	435.46	10	56.66	-	-	-
Venus2† [41, 190]	386.71	17	33.33	-	-	-
α-CROWN+MIP	301.23	15	40.00	125.48	46	36.11
GCP-CROWN with MIP cuts	**145.26**	**23**	**13.33**	**53.49**	**63**	**12.5**

* Results from VNN-COMP 2021 report [20]. † We use the latest code of Venus2 in the `vnncomp` branch, committed on Jul 18, 2022. Older versions cannot run these convolutional networks.

Table 3.3: **Verified accuracy (%)** and avg. per-example certification time (s) on 7 models from SDP-FO [89].

Dataset	Model	SDP-FO [89]*		PRIMA [273]		β-CROWN [389]		MN-BaB [115]		Venus2 [41, 190]		α-CROWN+MIP		GCP-CROWN		Upper
$\epsilon = 0.3$ and $\epsilon = 2/255$		Verified%	Time (s)	Ver.%	Time(s)	Ver.%	Time(s)	Ver.%	Time(s)	Ver.%	Time(s)	Ver.%	Time(s)	Ver.%	Time(s)	bound
MNIST	CNN-A-Adv	43.4	>20h	44.5	135.9	70.5	21.1	-	-	35.5	148.4	56.5	224.3	**72.0**	19.9	76.5
CIFAR	CNN-B-Adv	32.8	>25h	38.0	343.6	46.5	32.2	-	-	-	-	27.0	360.6	**48.5**	57.8	65.0
	CNN-B-Adv-4	46.0	>25h	53.5	43.8	54.0	11.6	-	-	-	-	52.5	129.5	**59.0**	21.5	63.5
	CNN-A-Adv	39.6	>25h	41.5	4.8	44.0	5.8	42.5	68.3	47.5	26.0	46.0	63.1	**48.5**	9.8	50.0
	CNN-A-Adv-4	40.0	>25h	45.0	4.9	46.0	5.6	46.0	37.7	47.5	13.1	**48.5**	16.4	**48.5**	5.7	49.5
	CNN-A-Mix	39.6	>25h	37.5	34.3	41.5	49.6	35.0	140.3	33.5	72.4	32.5	231.3	**47.5**	29.2	53.0
	CNN-A-Mix-4	47.8	>25h	48.5	7.0	50.5	5.9	49.0	70.9	49.0	37.3	52.5	77.7	**55.5**	12.4	57.5

* We run α-CROWN+MIP and MN-BaB with 600s timeout threshold for all models. "-" indicates that we could not run a model due to unsupported model structure or other errors. We run our GCP-CROWN with MIP cuts with a shorter 200s timeout for all models and it achieves better verified accuracy than all other baselines. Other results are reported from [389].

intermediate layer bounds from α-CROWN [424]. We use the same branch and bound algorithm as in β-CROWN and we use filtered smart branching (FSB) [286] as the branching heuristic in all experiments. Without cutting planes, GCP-CROWN becomes vanilla β-CROWN as we share the same code base as β-CROWN. We include model information and detailed setup for our experiments in Appendix C.3. GCP-CROWN has been integrated into the α-β-CROWN verifier, and the instructions to reproduce results in this chapter are available at `http://PaperCode.cc/GCP-CROWN`.

Results on `oval20` Benchmark in VNN-COMP 2020. `oval20` is a popular benchmark consistently used in huge amount of NN verifiers and it perfectly reflects the progress of NN verifiers. We include literature results for many baselines in Table 3.1. We are the only verifier that can *completely solve* all three models without any timeout. Our average runtime is significantly lower compared to the time of baselines because we have no timeout (counted as 3600s), and our slowest instance only takes about a few minutes while easy ones only take a few seconds, as shown in Figure 3.2. Additionally, we often use less number

of branches compared to the state-of-the-art verifier, β-CROWN, since our strong cutting planes help us to eliminate many hard to solve subdomains in BaB. Furthermore, we highlight that α-CROWN+MIP also achieves a low timeout rate, although it is much slower than our bound propagation based approach combined with cuts from a MIP solver.

Results on VNN-COMP 2021 Benchmarks. Among the eight scoring benchmarks in VNN-COMP 2021 [20], only two (`oval21` and `cifar10-resnet`) are most suitable for the evaluation of this work. Among other benchmarks, `acasxu` and `nn4sys` have low input dimensionality and require input space branching rather than ReLU branching; `verivital`, `mnistfc`, and `eran` benchmarks consist of small MLP networks that can be solved directly by MIP solvers; `marabou` contains mostly adversarial examples, making it a good benchmark for falsifiers rather than verifiers. We present our results in Table 3.2. Besides results from 6 VNN-COMP 2021 participants, we also include two additional baselines, α-CROWN+MIP (same as in Table 3.1), and MN-BaB [115], a recently proposed branch and bound framework with multi-neuron relaxations [271], which can be viewed as a restricted form of cutting planes. On the `oval21` benchmark, OVAL, VeriNet and α-β-CROWN are the best performing tools, verified 11 out of 27 instances, while we can certify *twice more* instances (22 out of 27) on this benchmark. On the `cifar10-resnet` benchmark, our verifier also solves the most number of instances and achieves the lowest average time. In fact, α-CROWN+MIP is also a strong baseline, solving 4 more instances than all competition participants, showing the importance of strong cutting planes. Our GCP-CROWN with MIP cuts combines the benefits of fast bound propagation on GPU with the strong cutting planes generated by a MIP solver and achieves the best performance. We present a more detailed analysis on the cutting planes used in this benchmark in Appendix C.3.2.

The `oval21` benchmark was also included as part of VNN-COMP 2022, concluded in July 2022, with a different sample of 30 instances. GCP-CROWN is part of the winning tool in VNN-COMP 2022, α-β-CROWN, which verified 25 out of 30 instances in this benchmark, outperforming the second place tool (MN-BaB [115] with multi-neuron relaxations) with 19 verified instances by a large margin. More results on VNN-COMP 2022 can be found in these slides[4].

Results on SDP-FO Benchmarks. We further evaluate our method on the SDP-FO benchmarks in [89, 389]. This benchmark contains 7 mostly adversarially trained MNIST and CIFAR models with 200 instances each, which are hard for many existing verifiers. Beyond

[4]A formal competition report is under preparation by VNN-COMP organizers; scores were presented in FLoC 2022: `https://drive.google.com/file/d/1nnRWSq3plsPvOT3V-drAF5D8zWGuO2VF/view`.

the baselines reported in [389], we also include two additional baselines, α-CROWN+MIP (same as in Table 3.1) and MN-BaB [115] (same as in Table 3.2). Table 3.3 shows that our method improves the percentage of verified images ("verified accuracy") on all models compared to state-of-the-art verifiers, further closing the gap between verified accuracy and empirical robust accuracy obtained by PGD attack (reported as "upper bound" in Table 3.3).

3.5 SUMMARY

In this chapter, we propose GCP-CROWN, an efficient and GPU-accelerated bound propagation method for NN certification capable of handling any cutting plane constraints. We combine GCP-CROWN with branch and bound and high quality cutting planes generated by a MIP solver to tighten the convex relaxation for NN certification. The combination of fast bound propagation and strong cutting planes lead to state-of-the-art certification performance on multiple benchmarks. Our work opens up a great opportunity for studying more efficient and powerful cutting planes for NN certification.

Limitations. Our work generalizes existing bound propagation methods that can handle only simple constraints (such as neuron split constraints in β-CROWN [389]) to general constraints, and we share a few common limitations as in previous works [49, 286, 425]: the branch-and-bound procces and bound propagation procedure are developed on ReLU networks, and it can be non-trivial to extend it to neural networks with non-piecewise-linear operations. In addition, we currently directly use cutting planes generated by a generic MIP solver, and there might exist stronger and faster cutting plane methods that can exploit the structure of the neural network certification problem. We hope these limitations can be addressed in future works.

Potential Negative Societal Impact. Our work focuses on formally proving desired properties of a neural network under investigation such as safety and robustness, which is an important direction of trustworthy machine learning and has overall positive societal impact. Since our verifier is a complete verifier, it might be possible to use it to find weakness of a neural network and guide adversarial attacks. However, we believe that formally characterizing a model's behavior and potential weakness is important for building robust models and preventing real-world malicious attack.

CHAPTER 4: TRAINING FOR ENHANCING BLACK-BOX CERTIFIED ROBUSTNESS: DRT

Certied approaches [77, 406] have been proposed to provide the robustness guarantees for given ML models, so that no additional attack can break the model under certain adversarial constraints. For instance, randomized smoothing has been proposed as an effective defense providing certified robustness [77, 204, 427]. Among different certified robustness approaches [225, 402, 424, 440], randomized smoothing provides a model-independent way to smooth a given ML model and achieves state-of-the-art certified robustness on large-scale datasets such as ImageNet.

Currently, almost all the existing certified approaches focus on the robustness of a single ML model. Given the observations that ensemble ML models are able to bring additional benefits in standard learning [284, 311], in this work we aim to ask: *Can an ensemble ML model provide additional benefits in terms of the certified robustness compared with a single model? If so, what are the sufficient and necessary conditions to guarantee such certified robustness gain?*

Empirically, we first find that *standard* ensemble models only achieve marginally higher certified robustness by directly appling randomized smoothing: with ℓ_2 perturbation radius 1.5, a single model achieves certified accuracy as 21.9%, while the average aggregation based ensemble of three models achieves certified accuracy as 24.2% on CIFAR-10 (Table 4.2). Given such observations, next we aim to answer: *How to improve the certified robustness of ensemble ML models? What types of conditions are required to improve the certified robustness for ML ensembles?*

In particular, from the theoretical perspective, we analyze the standard Weighted Ensemble (WE) and Max-Margin Ensemble (MME) protocols, and prove the sufficient and necessary conditions for the certifiably robust ensemble models under model-smoothness assumption. Specifically, we prove that: (1) an ensemble ML model is more certifiably robust than each single base model; (2) *diversified gradients* and *large confidence margins* of base models are the sufficient and necessary conditions for the certifiably robust ML ensembles. We show that these two key factors would lead to higher certified robustness for ML ensembles. We further propose *Ensemble-before-Smoothing* as the model smoothing strategy and prove the bounded model-smoothness with such strategy, which realizes our model-smoothness assumption.

Inspired by our theoretical analysis, we propose **D**iversity-**R**egularized **T**raining (DRT), a lightweight regularization-based ensemble training approach. DRT is composed of two simple yet effective and general regularizers to promote the diversified gradients and large

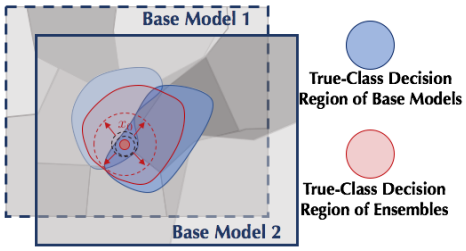


Figure 4.1: Illustration of a robust ensemble.

confidence margins respectively. DRT can be easily combined with existing ML approaches for training smoothed models, such as Gaussian augmentation [77] and adversarial smoothed training [317], with negligible training time overhead while achieves *significantly* higher certified robustness than state-of-the-art approaches consistently.

We conduct extensive experiments on a wide range of datasets including MNIST, CIFAR-10, and ImageNet. The experimental results show that DRT can achieve significantly higher certified robustness compared to baselines with similar training cost as training a single model. Furthermore, as DRT is flexible to integrate any base models, by using the pretrained robust single ML models as base models, DRT achieves the highest certified robustness so far to our best knowledge. For instance, on CIFAR-10 under ℓ_2 radius 1.5, the DRT-trained ensemble with three base models improves the certified accuracy from SOTA 24.2% to 30.3%; and under ℓ_2 radius 2.0, DRT improves the certified accuracy from SOTA 16.0% to 20.3%.

Technical Contributions. In this chapter, we conduct the *first* study for the sufficient and necessary conditions of certifiably robust ML ensembles and propose an efficient training algorithm DRT to achieve the state-of-the-art certified robustness. We make contributions on both theoretical and empirical fronts.

- We provide the *necessary* and *sufficient* conditions for robust ensemble ML models including Weighted Ensemble (WE) and Max-Margin Ensemble (MME) under the model-smoothness assumption. In particular, we prove that the *diversified gradients* and *large confidence margins* of base models are the sufficient and necessary conditions of certifiably robust ensembles. We also prove the bounded model-smoothness via proposed *Ensemble-before-Smoothing* strategy, which realizes our model-smoothness assumption.

- To analyze different ensembles, we prove that when the adversarial transferability among base models is low, WE is more robust than MME. We also prove that the ML ensemble is more robust than a single base model under the model-smoothness assumption.

- Based on the theoretical analysis of the sufficient and necessary conditions, we propose DRT, a lightweight regularization-based training approach that can be easily combined with different training approaches and ensemble protocols with small training cost overhead.
- We conduct extensive experiments to evaluate the effectiveness of DRT on various datasets, and we show that to the best of your knowledge, DRT can achieve the *highest* certified robustness, outperforming all existing baselines.

Related work. DNNs are known vulnerable to adversarial examples [359]. To defend against such attacks, several empirical defenses have been proposed [252, 288]. For ensemble models, existing work mainly focuses on empirical robustness [67, 229, 287] where the robustness is measured by accuracy under existing attacks and no certified robustness guarantee could be provided or enhanced; or certify the robustness for a standard weighted ensemble [234, 444] using either LP-based [443] verification or randomized smoothing without considering the model diversity [234] to boost their certified robustness. In this chapter, we aim to prove that the diversified gradient and large confidence margin are the sufficient and necessary conditions for certifiably robust ensemble ML models. Moreover, to our best knowledge, we propose the *first* training approach to boost the *certified* robustness of ensemble ML models.

Randomized smoothing [77, 204] has been proposed to provide certified robustness for a single ML model. It achieved the state-of-the-art certified robustness on large-scale dataset such as ImageNet and CIFAR-10 under ℓ_2 norm. Several approaches have been proposed to further improve it by: (1) choosing different smoothing distributions for different ℓ_p norms [106, 427, 442], and (2) training more robust smoothed classifiers, using data augmentation [77], unlabeled data [56], adversarial training [317], regularization [218, 437], and denoising [319]. In this chapter, we propose a suitable smoothing strategy and training approach to improve the certified robustness of ML ensembles.

4.1 CHARACTERIZING ML ENSEMBLE ROBUSTNESS

In this section, we prove the sufficient and necessary robustness conditions for both general and smoothed ML ensemble models. Based on these robustness conditions, we discuss the key factors for improving the certified robustness of an ensemble, compare the robustness of ensemble models with single models, and outline several findings based on additional theoretical analysis.

4.1.1 Preliminaries

Notations. Throughout the chapter, we consider the classification task with C classes. We first define the classification scoring function $f : \mathbb{R}^d \to \mathbf{\Delta}^C$, which maps the input to a *confidence vector*, and $f(\boldsymbol{x})_i$ represents the confidence for the ith class. We mainly focus on the confidence after normalization, i.e., $\mathbf{\Delta}^C$ is the probability simplex. To characterize the *confidence margin* between two classes, we define $f^{y_1/y_2}(\boldsymbol{x}) := f(\boldsymbol{x})_{y_1} - f(\boldsymbol{x})_{y_2}$. The corresponding *prediction* $F : \mathbb{R}^d \to [C]$ is defined by $F(\boldsymbol{x}) := \arg\max_{i\in[C]} f(\boldsymbol{x})_i$. We are also interested in the *runner-up prediction* $F^{(2)}(\boldsymbol{x}) := \arg\max_{i\in[C]:i\neq F(\boldsymbol{x})} f(\boldsymbol{x})_i$.

r-Robustness. For brevity, we consider the model's certified robustness, against the ℓ_2-bounded perturbations as defined below. Our analysis can be generalizable for ℓ_1 and ℓ_∞ perturbations, leveraging existing work [214, 218, 427].

Definition 4.1 (r-Robustness)**.** For a prediction function $F : \mathbb{R}^d \to [C]$ and input $\boldsymbol{x}_0$, if all instance $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 < r\}$ satisfies $F(\boldsymbol{x}) = F(\boldsymbol{x}_0)$, we say model F is *r-robust* (at point $\boldsymbol{x}_0$).

Remark 4.1. Using the notation in Section 1.1, model F being r-robust at point $\boldsymbol{x}_0$ means that for the property of robustness against ℓ_p-bounded perturbations, model is trustworthy under threat model $\mathcal{R}(r)$, i.e., certified radius is no smaller than r.

Ensemble Protocols. An ensemble model contains N *base models* $\{F_i\}_{i=1}^N$, where $F_i(\boldsymbol{x})$ and $F_i^{(2)}(\boldsymbol{x})$ are their top and runner-up predictions for given input $\boldsymbol{x}$ respectively. The ensemble prediction is denoted by $\mathcal{M} : \mathbb{R}^d \to [C]$, which is computed based on outputs of base models following certain ensemble protocols. In this chapter, we consider both Weighted Ensemble (WE) and Maximum Margin Ensemble (MME).

Definition 4.2 (Weighted Ensemble (WE))**.** Given N base models $\{F_i\}_{i=1}^N$, and the weight vector $\{w_i\}_{i=1}^N \in \mathbb{R}_+^N$, the weighted ensemble $\mathcal{M}_{\text{WE}}$: $\mathbb{R}^d \to [C]$ is defined by

$$\mathcal{M}_{\text{WE}}(\boldsymbol{x}_0) := \arg\max_{i\in[C]} \sum_{j=1}^{N} w_j f_j(\boldsymbol{x}_0)_i. \tag{4.1}$$

Definition 4.3 (Max-Margin Ensemble (MME))**.** Given N base models $\{F_i\}_{i=1}^N$, for input $\boldsymbol{x}_0$, the max-margin ensemble model $\mathcal{M}_{\text{MME}} : \mathbb{R}^d \to [C]$ is defined by

$$\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) := F_c(\boldsymbol{x}_0) \quad \text{where} \quad c = \arg\max_{i\in[N]} \left(f_i(\boldsymbol{x}_0)_{F_i(\boldsymbol{x}_0)} - f_i(\boldsymbol{x}_0)_{F_i^{(2)}(\boldsymbol{x}_0)} \right). \tag{4.2}$$

The commonly-used WE [234, 444] sums up the weighted confidence of base models $\{F_i\}_{i=1}^N$ with weight vector $\{w_i\}_{i=1}^N$, and predicts the class with the highest weighted confidence. The standard average ensemble can be viewed as a special case of WE (where all w_i's are equal). MME chooses the base model with the largest confidence margin between the top and the runner-up classes, which is a direct extension from max-margin training [155].

Randomized Smoothing. Randomized smoothing [77, 204] provides certified robustness by constructing a smoothed model from a given model. Formally, let $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ be a Gaussian random variable, for any given model $F : \mathbb{R}^d \to [C]$ (can be an ensemble), we define *smoothed confidence* function $g_F^\epsilon : \mathbb{R}^d \to \boldsymbol{\Delta}^C$ such that

$$g_F^\epsilon(\boldsymbol{x})_j := \mathop{\mathbb{E}}_{\epsilon \sim \mathcal{N}(0,\sigma^2 \boldsymbol{I}_d)} \mathbb{I}[F(\boldsymbol{x}+\epsilon) = j] = \Pr_{\epsilon \sim \mathcal{N}(0,\sigma^2 \boldsymbol{I}_d)}(F(\boldsymbol{x}+\epsilon) = j). \tag{4.3}$$

Intuitively, $g_F^\epsilon(\boldsymbol{x})_j$ is the probability of base model F's prediction on the jth class given Gaussian smoothed input. The smoothed classifier $G_F^\epsilon : \mathbb{R}^d \to [C]$ outputs the class with highest smoothed confidence: $G_F^\epsilon(\boldsymbol{x}) := \arg\max_{j \in [C]} g_F^\epsilon(\boldsymbol{x})_j$. Let c_A be the predicted class for input $\boldsymbol{x}_0$, i.e., $c_A := G_F^\epsilon(\boldsymbol{x}_0)$. Cohen et al. show that G_F^ϵ is $(\sigma \Phi^{-1}(g_F^\epsilon(\boldsymbol{x}_0)_{c_A}))$-robust at input $\boldsymbol{x}_0$, i.e., the certified radius is $\sigma \Phi^{-1}(g_F^\epsilon(\boldsymbol{x}_0)_{c_A})$ (more discussed in Proposition 2.1). In practice, we will leverage the smoothing strategy together with Monte-Carlo sampling to certify ensemble robustness.

4.1.2 Robustness Conditions for General Ensemble Models

We will first provide sufficient and necessary conditions for robust ensembles under the model-smoothness assumption.

Definition 4.4 (β-Smoothness)**.** A differentiable function $f : \mathbb{R}^d \to \mathbb{R}^C$ is β-smooth, if for any $\boldsymbol{x}_1$, $\boldsymbol{x}_2 \in \mathbb{R}^d$ and any output dimension $j \in [C]$, $\frac{\|\nabla_{\boldsymbol{x}_1} f(\boldsymbol{x}_1)_j - \nabla_{\boldsymbol{x}_2} f(\boldsymbol{x}_2)_j\|_2}{\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2} \leqslant \beta$.

The definition of β-smoothness is inherited from optimization theory literature, and it is equivalent to the curvature bound in certified robustness literature [344]. β quantifies the non-linearity of function f, where higher β indicates more rigid functions/models and smaller β indicates smoother ones. When $\beta = 0$ the function/model is linear.

For Weighted Ensemble (WE), we have the following robustness conditions.

Theorem 4.1 (Gradient and Confidence Margin Conditions for WE Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$, and $\mathcal{M}_{\text{WE}}$ as a WE defined over base models $\{F_i\}_{i=1}^N$ with weights $\{w_i\}_{i=1}^N$. $\mathcal{M}_{\text{WE}}(\boldsymbol{x}_0) = y_0$. All base models F_i's are β-smooth.

- (Sufficient Condition) The $\mathcal{M}_{\text{WE}}$ is r-robust at point $\boldsymbol{x}_0$ if for any $y_i \neq y_0$,

$$\Big\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^{N} w_j f_j^{y_0/y_i}(\boldsymbol{x}_0) - \beta r \sum_{j=1}^{N} w_j, \tag{4.4}$$

- (Necessary Condition) If $\mathcal{M}_{\text{WE}}$ is r-robust at point $\boldsymbol{x}_0$, for any $y_i \neq y_0$,

$$\Big\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^{N} w_j f_j^{y_0/y_i}(\boldsymbol{x}_0) + \beta r \sum_{j=1}^{N} w_j. \tag{4.5}$$

The proof follows from Taylor expansion at $\boldsymbol{x}_0$ and we leave the detailed proof in Appendix D.1.2. When it comes to Max-Margin Ensemble (MME), the derivation of robust conditions is more involved. In Theorem D.1 (Appendix D.1.1) we derive the robustness conditions for MME composed of two base models. The robustness conditions have highly similar forms as those for WE in Theorem 4.1. Thus, for brevity, we focus on discussing Theorem 4.1 for WE hereinafter and similar conclusions can be drawn for MME (details are in Appendix D.1.1).

To analyze Theorem 4.1, we define *Ensemble Robustness Indicator* (ERI) as such:

$$I_{y_i} := \Big\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \Big/ \|\boldsymbol{w}\|_1 - \frac{1}{r\|\boldsymbol{w}\|_1} \sum_{j=1}^{N} w_j f_j^{y_0/y_i}(\boldsymbol{x}_0). \tag{4.6}$$

ERI appears in both sufficient (Equation (4.4)) and necessary (Equation (4.5)) conditions. In both conditions, *smaller* ERI means *more* certifiably robust ensemble. Note that we can analyze the robustness under different attack radius r by directly varying r in Equations (4.4) and (4.5). When r becomes larger, the gap between the RHS of two inequalities ($2\beta r \sum_{j=1}^{N} w_j$) also becomes larger, and thus it becomes harder to determine robustness via Theorem 4.1. This is because the first-order condition implied by Theorem 4.1 becomes coarse when r is large. However, due to bounded β as we will show, the training approach motivated by the theorem still empirically works well under large r.

Diversified Gradients. The core of first term in ERI is the magnitude of the vector sum of gradients: $\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \|_2$. According to the law of cosines: $\|\boldsymbol{a} + \boldsymbol{b}\|_2 = \sqrt{\|\boldsymbol{a}\|_2^2 + \|\boldsymbol{b}\|_2^2 + 2\|\boldsymbol{a}\|_2\|\boldsymbol{b}\|_2 \cos\langle \boldsymbol{a}, \boldsymbol{b} \rangle}$, to reduce this term, we could either reduce the base models' gradient magnitude or diversify their gradients (in terms of cosine similarity). Since simply reducing base models' gradient magnitude would hurt model expressivity [159], during regularization the main functionality of this term would be promoting diversified gradients.

Large Confidence Margins. The core of second term in ERI is the confidence margin: $\sum_{j=1}^{N} w_j f_j^{y_0/y_i}(\boldsymbol{x}_0)$. Due to the negative sign of second term in ERI, we need to increase this term, i.e., we need to increase confidence margins to achieve higher ensemble robustness.

In summary, the diversified gradients and large confidence margins are the sufficient and necessary conditions for high certified robustness of ensembles. In Section 4.2, we will directly regularize these two key factors to promote certified robustness of ensembles.

Impact of Model-Smoothness Bound β. From Theorem 4.1, we observe that: (1) if $\min_{y_i \neq y_0} I_{y_i} \leqslant -\beta r$, $\mathcal{M}_{\text{WE}}$ is guaranteed to be r-robust (sufficient condition); and (2) if $\min_{y_i \neq y_0} I_{y_i} > \beta r$, $\mathcal{M}_{\text{WE}}$ cannot be r-robust (necessary condition). However, if $\min_{y_i \neq y_0} I_{y_i} \in (-\beta r, \beta r]$, we only know $\mathcal{M}_{\text{WE}}$ is possibly r-robust. As a result, the model-smoothness bound β decides the correlation strength between $\min_{y_i \neq y_0} I_{y_i}$ and the robustness of $\mathcal{M}_{\text{WE}}$: if β becomes larger, $\min_{y_i \neq y_0} I_{y_i}$ is more likely to fall in $(-\beta r, \beta r]$, inducing an undetermined robustness status from Theorem 4.1, vice versa. Specifically, when $\beta = 0$, i.e., all base models are linear, the gap is closed and we can always certify the robustness of $\mathcal{M}_{\text{WE}}$ via comparing $\min_{y_i \neq y_0}$ with 0. Similar observations can be drawn for MME. Therefore, to strengthen the correlation between I_{y_i} and ensemble robustness, we would need model-smoothness bound β to be small.

4.1.3 Robustness Conditions for Smoothed Ensemble Models

Typically neural networks are nonsmooth or admit only coarse smoothness bounds [347], i.e., β is large. Therefore, applying Theorem 4.1 for normal nonsmooth models would lead to near-zero certified radius. Therefore, we propose soft smoothing to enforce the smoothness of base models. However, with the soft smoothed base models, directly applying Theorem 4.1 to certify robustness is still practically challenging, since the LHS of Equations (4.4) and (4.5) involves gradient of the soft smoothed confidence. A precise computation of such gradient requires high-confidence estimation of high-dimensional vectors via sampling, which requires linear number of samples with respect to input dimension [267, 317] and is thus too expensive in practice. To solve this issue, we then propose *Ensemble-before-Smoothing* as the practical smoothing protocol, which serves as an approximation of soft smoothing, so as to leverage the randomized smoothing based techniques for certification.

Soft Smoothing. To impose base models' smoothness, we now introduce soft smoothing [193], which applies randomized smoothing over *the confidence scores*. Given base model's confidence function $f : \mathbb{R}^d \to \mathbf{\Delta}^C$ (see Section 4.1.1), we define *soft smoothed confidence* by

$\bar{g}_f^\epsilon : \boldsymbol{x} \mapsto \mathbb{E}_\epsilon f(\boldsymbol{x}+\epsilon)$. Note that *soft* smoothed confidence is different from smoothed confidence g_F^ϵ defined in (4.3). We consider soft smoothing instead of classical smoothing in (4.3) since soft smoothing reveals differentiable and thus practically regularizable training objectives. The following theorem shows the smoothness bound for $\bar{g}_f^\epsilon$.

Theorem 4.2 (Model-Smoothness Upper Bound for $\bar{g}_f^\epsilon$)**.** Let $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ be a Gaussian random variable, then the soft smoothed confidence function $\bar{g}_f^\epsilon$ is $(2/\sigma^2)$-smooth.

We defer the proof to Appendix D.1.4. The proof views the Gaussian smoothing as the Weierstrass transform [401] of a function from $\mathbb{R}^d$ to $[0,\,1]^C$, leverages the symmetry property, and bounds the absolute value of diagonal elements of the Hessian matrix. Note that a Lipschitz constant $\sqrt{2/(\pi\sigma^2)}$ is derived for smoothed confidence in previous work [317, Lemma 1], which characterizes only the first-order smoothness property; while our bound in addition shows the second-order smoothness property. In Appendix D.1.4, we further show that our smoothness bound in Theorem 4.2 is tight up to a constant factor.

Now, we apply WE and MME protocols with these soft smoothness confidence $\{\bar{g}_i^\epsilon(\boldsymbol{x}_0)\}_{i=1}^N$ as base models' confidence scores, and obtain soft ensemble $\bar{G}^\epsilon_{\mathcal{M}_{\text{WE}}}$ and $\bar{G}^\epsilon_{\mathcal{M}_{\text{MME}}}$ respectively. Since each $\bar{g}_i^\epsilon$ is $(2/\sigma^2)$-smooth, take WE as an example, we can study the ensemble robustness with Theorem 4.1. We state the full statement in Corollary D.2 (and in Corollary D.3 for MME) in Appendix D.1.1. From the corollary, we observe that the corresponding ERI for the soft smoothed WE can be written as

$$\bar{I}_{y_i} := \Big\| \mathop{\mathbb{E}}_{\epsilon} \nabla_{\boldsymbol{x}} \sum_{j=1}^{N} w_j f_j^{y_0/y_i}(\boldsymbol{x}_0+\epsilon) \Big\|_2 \Big/ \|\boldsymbol{w}\|_1 - \frac{1}{r\|\boldsymbol{w}\|_1} \mathop{\mathbb{E}}_{\epsilon} \sum_{j=1}^{N} w_j f_j^{y_0/y_i}(\boldsymbol{x}_0+\epsilon). \tag{4.7}$$

We have following observations: (1) unlike for standard models with unbounded β, for the smoothed ensemble models, this ERI (Equation (4.7)) would have *guaranteed* correlation with the model robustness since $\beta = \Theta(1/\sigma^2)$ is bounded and can be controlled by tuning σ for smoothing. (2) we can still control ERI by diversifying gradients and ensuring large confidence margins as discussed in Section 4.1.2, but need to compute on the noise augmented input $\boldsymbol{x}_0 + \epsilon$ instead of original input $\boldsymbol{x}_0$.

Towards Practical Certification. As outlined at the beginning of this subsection, even with smoothed base models, certifying robustness using Theorem 4.1 is practically difficult. Therefore, we introduce *Ensemble-before-Smoothing* strategy as below to construct $G^\epsilon_{\mathcal{M}_{\text{WE}}}$ and $G^\epsilon_{\mathcal{M}_{\text{MME}}}$ as approximations of soft ensemble $\bar{G}^\epsilon_{\mathcal{M}_{\text{WE}}}$ and $\bar{G}^\epsilon_{\mathcal{M}_{\text{MME}}}$ respectively.

Definition 4.5 (***Ensemble-before-Smoothing*** **(EBS)).** Let $\mathcal{M}$ be an ensemble model over base models $\{F_i\}_{i=1}^N$ and ϵ be a random variable. The EBS strategy construct smoothed classifier $G_{\mathcal{M}}^{\epsilon} : \mathbb{R}^d \to [C]$ that picks the class with highest smoothed confidence of $\mathcal{M}$: $G_{\mathcal{M}}^{\epsilon}(\boldsymbol{x}) := \arg\max_{j\in[C]} g_{\mathcal{M}}^{\epsilon}(\boldsymbol{x})_j$.

Here $\mathcal{M}$ could be either $\mathcal{M}_{\text{WE}}$ or $\mathcal{M}_{\text{MME}}$. EBS aims to approximate the soft smoothed ensemble. Formally, use WE as an example, we let $f_{\mathcal{M}_{\text{WE}}} := \frac{\sum_{j=1}^N w_j f_j}{\|\boldsymbol{w}\|_1}$ to be WE ensemble's confidence, then

$$g_{\mathcal{M}_{\text{WE}}}^{\epsilon}(\boldsymbol{x})_i = \mathop{\mathbb{E}}_{\epsilon} \mathbb{I}[\mathcal{M}_{\text{WE}}(\boldsymbol{x}+\epsilon) = i] \approx \mathop{\mathbb{E}}_{\epsilon} f_{\mathcal{M}_{\text{WE}}}(\boldsymbol{x}+\epsilon)_i = \frac{\sum_{j=1}^N w_j (\bar{g}_{f_j}^{\epsilon})_i}{\sum_{j=1}^N w_j} = \bar{g}_{\mathcal{M}_{\text{WE}}}^{\epsilon}(\boldsymbol{x})_i \quad (4.8)$$

where LHS is the smoothed confidence of EBS ensemble and RHS is the soft smoothed ensemble's confidence. Such approximation is also adopted in existing work [193, 317, 437] and shown effective and useful. Therefore, our robustness analysis of soft smoothed ensemble still applies with EBS and we can control ERI in Equation (4.7) to improve the certified robustness of EBS ensemble. For EBS ensemble, we can leverage randomized smoothing based techniques to compute the robustness certification (see Proposition D.5 in Appendix D.2).

Remark 4.2. As will present, we will control ERI of the ensemble to boost the robustness in DRT. Then, we will compute the robustness certification leveraging the standard certification of randomized smoothing. Hence, it could be possible that standard certification of randomized smoothing can not fully reflect the improved robustness due to the certification looseness. Even with this negative effect, our DRT approach shows superior certified robustness, and we conjecture advanced certification approaches (e.g., DSRS in Chapter 2) can certify further visibly larger radii for our DRT approach.

4.1.4 Additional Properties of ML Ensembles

Comparison between Ensemble and Single-Model Robustness. In Appendix D.1.1, we show Corollary D.1, a corollary of Theorem 4.1, which indicates that when the base models are smooth enough, both WE and MME ensemble models are more certifiably robust than the base models. This aligns with our empirical observations (see Table 4.1 and Table 4.2), though *without* advanced training approaches such as DRT, the improvement of robustness brought by ensemble itself is marginal. In Appendix D.1.1, we also show larger number of base models N can lead to better certified robustness.

Comparison between WE and MME Robustness. Since in actual computing, the certified radius of a smoothed model in standard (Neyman-Pearson-based) randomized smoothing is directly correlated with the probability of correct prediction under smoothed input (see Proposition 2.1), we study the robustness of both WE and MME along with single models from the statistical robustness perspective in the full published version of this chapter [431]. From the study, we have the following theoretical observations verified by numerical experiments: (1) MME is more robust when the adversarial transferability is high; while WE is more robust when the adversarial transferability is low. (2) If we further assume that $f_i(x_0 + \epsilon)_{y_0}$ follows marginally uniform distribution, when the number of base models N is sufficiently large, MME is always more certifiably robust.

4.2 DIVERSITY-REGULARIZED TRAINING

Inspired by the above key factors in the sufficient and necessary conditions for the certifiably robust ensembles, we propose Diversity-Regularized Training (DRT). In particular, let $\boldsymbol{x}_0$ be a training sample, DRT contains the following two regularization terms in the objective function to minimize:

- Gradient Diversity Loss (GD Loss):

$$\mathcal{L}_{\mathrm{GD}}(\boldsymbol{x}_0)_{ij} = \left\| \nabla_{\boldsymbol{x}} f_i^{y_0/y_i^{(2)}}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_j^{y_0/y_j^{(2)}}(\boldsymbol{x}_0) \right\|_2. \tag{4.9}$$

- Confidence Margin Loss (CM Loss):

$$\mathcal{L}_{\mathrm{CM}}(\boldsymbol{x}_0)_{ij} = f_i^{y_i^{(2)}/y_0}(\boldsymbol{x}_0) + f_j^{y_j^{(2)}/y_0}(\boldsymbol{x}_0). \tag{4.10}$$

In Equations (4.9) and (4.10), y_0 is the ground-truth label of $\boldsymbol{x}_0$, and $y_i^{(2)}$ (or $y_j^{(2)}$) is the runner-up class of base model F_i (or F_j). Intuitively, for each model pair (F_i, F_j) where $i, j \in [N]$ and $i \neq j$, the GD loss promotes the *diversity of gradients* between the base model F_i and F_j. Note that the gradient computed here is actually the gradient difference between different labels. As our theorem reveals, it is the gradient difference between different labels instead of pure gradient itself that matters, which improves existing understanding of gradient diversity [93, 287]. Specifically, the GD loss encourages both large gradient diversity and small base models' gradient magnitude in a naturally balanced way, and encodes the interplay between gradient magnitude and direction diversity. In contrast, solely regularizing the base models' gradient would hurt the model's benign accuracy, and solely regularizing

gradient diversity is hard to realize due to the boundedness of cosine similarity. The CM loss encourages the *large margin* between the true and runner-up classes for base models. Both regularization terms are directly motivated by theoretical analysis in Section 4.1.

For each input $\boldsymbol{x}_0$ with ground truth y_0, we use $\boldsymbol{x}_0+\epsilon$ with $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ as training input for each base model (i.e., Gaussian augmentation). We call two base models $\big(F_i, F_j\big)$ a *valid* model pair at $(\boldsymbol{x}_0, y_0)$ if both $F_i(\boldsymbol{x}_0+\epsilon)$ and $F_j(\boldsymbol{x}_0+\epsilon)$ equal to y_0. For every valid model pair, we apply DRT: GD Loss and CM Loss with ρ_1 and ρ_2 as the weight hyperparameters as below.

$$\mathcal{L}_{\text{train}} = \sum_{i\in[N]} \mathcal{L}_{\text{std}}(\boldsymbol{x}_0+\epsilon, y_0)_i + \rho_1 \sum_{\substack{i,j\in[N], i\neq j \\ (F_i, F_j) \text{ is valid}}} \mathcal{L}_{\text{GD}}(\boldsymbol{x}_0+\epsilon)_{ij} + \rho_2 \sum_{\substack{i,j\in[N], i\neq j \\ (F_i, F_j) \text{ is valid}}} \mathcal{L}_{\text{CM}}(\boldsymbol{x}_0+\epsilon)_{ij}. \tag{4.11}$$

The standard training loss $\mathcal{L}_{\text{std}}(\boldsymbol{x}_0+\epsilon, y_0)_i$ of each base model F_i is either cross-entropy loss [77], or adversarial training loss [317]. This standard training loss will help to produce sufficient valid model pairs with high benign accuracy for robustness regularization. Specifically, as discussed in Section 4.1.3, we compute $\mathcal{L}_{\text{GD}}$ and $\mathcal{L}_{\text{CM}}$ on the noise augmented inputs $(\boldsymbol{x}_0+\epsilon)$ instead of $\boldsymbol{x}_0$ to improve the certified robustness for the *smoothed* ensemble.

Discussion. To our best knowledge, this is the *first* training approach that is able to promote the certified robustness of ML ensembles, while existing work either only provide empirical robustness without guarantees [178, 287, 428, 429], or tries to only optimize the weights of Weighted Ensemble [234, 444]. We should notice that, though concepts similar with the gradient diversity have been explored in empirically robust ensemble training (e.g., ADP [287], GAL [178]), directly applying these regularizers cannot train models with high *certified robustness* due to the lack of theoretical guarantees in their design. We indicate this through ablation studies in [431, Appendix G.4]. Our approach is generalizable for other ℓ_p-bounded perturbations such as ℓ_1 and ℓ_∞ leveraging existing work [204, 214, 218, 427].

4.3 EXPERIMENTAL EVALUATION

To make a thorough comparison with existing certified robustness approaches, we evaluate DRT on different datasets including MNIST [203], CIFAR-10 [191], and ImageNet [94], based on both MME and WE protocols. Overall, we show that the DRT-enabled ensemble outperforms all baselines in terms of certified robustness under different settings.

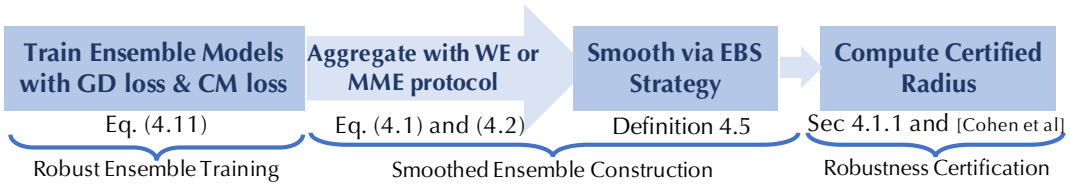


Figure 4.2: Pipeline for DRT-based ensemble.

4.3.1 Experimental Setup

Baselines. We consider the following state-of-the-art baselines for certified robustness: Gaussian smoothing [77], SmoothAdv [317], MACER [437], Stability [218], and SWEEN [234]. Detail description of these baselines can be found in Appendix D.4. We follow the configurations of baselines, and compare DRT-based ensemble with Gaussian Smoothing, SmoothAdv, and MACER on all datasets, and in addition compare it with other baselines on MNIST and CIFAR-10 considering the training efficiency. There are other baselines, e.g., [165]. However, SmoothAdv performs consistently better across different datasets, so we mainly consider SmoothAdv as our strong baseline.

Models. For base models in our ensemble, we follow the configurations used in baselines: LeNet [202], ResNet-110, and ResNet-50 [142] for MNIST, CIFAR-10, and ImageNet datasets respectively. Throughout the experiments, we use $N = 3$ base models to construct the ensemble for demonstration. We expect more base models would yield higher ensemble robustness.

Training Details. We follow Section 4.2 to train the base models. We combine DRT with Gaussian smoothing and SmoothAdv (i.e., instantiating $\mathcal{L}_{\text{std}}$ by either cross-entropy loss [77, 427] or adversarial training loss [317]). We leave training details along with hyperparameters in Appendix D.4.

Pipeline. After the base models are trained with DRT, we aggregate them to form the ensemble $\mathcal{M}$, using either WE or MME protocol (see Definitions 4.2 and 4.3). If we use WE, to filter out the effect of different weights, we adopt the average ensemble where all weights are equal. We also studied how optimizing weights can further improve the certified robustness in [431, Appendix G.3]. Then, we leverage *Ensemble-before-Smoothing* strategy to form a smoothed ensemble (see Definition 4.5). Finally, we compute the certified robustness for the smoothed ensemble based on Monte-Carlo sampling with high-confidence (99.9%). The training pipeline is shown in Figure 4.2.

Table 4.1: Certified accuracy under different radii on MNIST dataset. The grey rows present the performance of the proposed DRT approach. The brackets show the base models we use.

Radius r	0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50
Gaussian [77]	99.1	97.9	96.6	94.7	90.0	83.0	68.2	46.6	33.0	20.5	11.5
SmoothAdv [317]	99.1	98.4	97.0	96.3	93.0	87.7	80.2	66.3	43.2	34.3	24.0
MACER [437]	99.2	98.5	97.4	94.6	90.2	83.5	72.4	54.4	36.6	26.4	16.5
Stability [218]	99.3	**98.6**	97.1	93.8	90.7	83.2	69.2	46.8	33.1	20.0	11.2
SWEEN (Gaussian) [234]	99.2	98.4	96.9	94.9	90.5	84.4	71.1	48.9	35.3	23.7	12.8
SWEEN (SmoothAdv) [234]	99.2	98.2	97.4	96.3	93.4	88.1	81.0	67.2	44.5	34.9	25.0
MME (Gaussian)	99.2	98.4	96.8	94.9	90.5	84.3	69.8	48.8	34.7	23.4	12.7
DRT + MME (Gaussian)	**99.5**	**98.6**	97.5	95.5	92.6	86.8	76.5	60.2	43.9	36.0	29.1
MME (SmoothAdv)	99.2	98.2	97.3	96.4	93.2	88.1	80.6	67.9	44.8	35.0	25.2
DRT + MME (SmoothAdv)	99.2	98.4	**97.6**	**96.7**	93.1	**88.5**	83.2	68.9	48.2	**40.3**	34.7
WE (Gaussian)	99.2	98.4	96.9	94.9	90.6	84.5	70.4	49.0	35.2	23.7	12.9
DRT + WE (Gaussian)	**99.5**	**98.6**	97.4	95.6	92.6	86.7	76.7	60.2	43.9	35.8	29.0
WE (SmoothAdv)	99.1	98.2	97.4	96.4	**93.4**	88.2	81.1	67.9	44.7	35.2	24.9
DRT + WE (SmoothAdv)	99.1	98.4	**97.6**	**96.7**	**93.4**	**88.5**	**83.3**	**69.6**	**48.3**	40.2	**34.8**

Evaluation Metric. We report the standard *certified accuracy* under different ℓ_2 radii r's as our evaluation metric following existing work [77, 165, 428, 437]. More evaluation details are in Appendix D.4.

4.3.2 Experimental Results

Here we consider ensemble models consisting of three base models. We show that 1) DRT-based ensembles outperform the SOTA baselines significantly especially under large perturbation radii; 2) smoothed ensembles are always more certifiably robust than each base model (Corollary D.1 in Appendix D.1.1); 3) applying DRT for either MME or WE ensemble protocols achieves similar and consistent improvements on certified robustness.

Certified Robustness of DRT with Different Ensemble Protocols. The evaluation results on MNIST, CIFAR-10, ImageNet are shown in Tables 4.1, 4.2, 4.3 respectively. It is clear that though the certified accuracy of a single model can be improved by directly applying either MME or WE ensemble training (proved in Corollary D.1), such improvements are usually negligible (usually less than 2%). In contrast, in all tables we find DRT provides significant gains on certified robustness for both MME and WE (up to over 16% as Table 4.1 shows). From Tables 4.1 and 4.2 on MNIST and CIFAR-10, we find that compared with all baselines, DRT-based ensemble achieves the highest robust accuracy, and the performance gap is more pronounced on large radii (over 8% for $r = 2.50$ on MNIST and 6% for $r = 1.50$ on CIFAR-10). We also demonstrate the scalability of DRT by training on ImageNet, and Table 4.3 shows that DRT achieves the highest certified robustness under large radii. It is

Table 4.2: Certified accuracy under different radii on CIFAR-10 dataset. The grey rows present the performance of the proposed DRT approach. The brackets show the base models we use.

Radius r	0.00	0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00
Gaussian [77]	78.9	64.4	47.4	33.7	23.1	18.3	13.6	10.5	7.3
SmoothAdv [317]	68.9	61.0	54.4	45.7	34.8	28.5	21.9	18.2	15.7
MACER [437]	79.5	68.8	55.6	42.3	35.0	27.5	23.4	20.4	17.5
Stability [218]	72.4	58.2	43.4	27.5	23.9	16.0	15.6	11.4	7.8
SWEEN (Gaussian) [234]	81.2	68.7	54.4	38.1	28.3	19.6	15.2	11.5	8.6
SWEEN (SmoothAdv) [234]	69.5	62.3	55.0	46.2	35.2	29.5	22.4	19.3	16.6
MME (Gaussian)	80.8	68.2	53.4	38.4	29.0	19.6	15.6	11.6	8.8
DRT + MME (Gaussian)	81.4	**70.4**	57.8	43.8	34.4	29.6	24.9	20.9	16.6
MME (SmoothAdv)	71.4	64.5	57.6	48.4	36.2	29.8	23.9	19.5	16.2
DRT + MME (SmoothAdv)	72.6	67.2	**60.2**	50.4	39.4	35.8	**30.4**	24.0	20.1
WE (Gaussian)	80.8	68.4	53.6	38.4	29.2	19.7	15.9	11.8	8.9
DRT + WE (Gaussian)	**81.5**	**70.4**	57.9	44.0	34.2	29.6	24.9	20.8	16.4
WE (SmoothAdv)	71.8	64.6	57.8	48.5	36.2	29.6	24.2	19.6	16.0
DRT + WE (SmoothAdv)	72.6	67.0	**60.2**	**50.5**	**39.5**	**36.0**	30.3	**24.1**	**20.3**

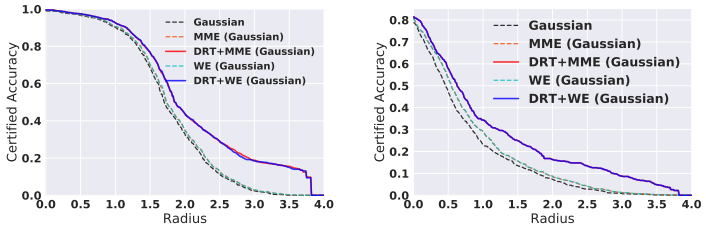
clear that DRT can be easily combined with existing training approaches (e.g. Gaussian smoothing or SmoothAdv), boost their certified robustness, and set the state-of-the-art results to the best of our knowledge.

To evaluate the computational cost of DRT, we analyze the theoretical complexity in Appendix D.3 and compare the efficiency of different methods in practice in Appendices D.4.1 and D.4.2. In particular, we show that DRT with Gaussian Smoothing base models even achieves around two times speedup compared with SmoothAdv with comparable or even higher certified robustness, since DRT does not require adversarial training. More discussions about hyper-parameters settings for DRT can be found in Appendix D.4. In the full version [431, Appendix G.4], we also show that our proposed DRT approach could achieve $6\% \sim 10\%$ higher certified accuracy compared to adapted ADP [287] and GAL [178] training on large radii for both MNIST and CIFAR-10 datasets.

Certified Accuracy with Different Perturbation Radius. We visualize the trend of certified accuracy along with different perturbation radii in Figure 4.3. For each radius r, we present the best certified accuracy among different smoothing parameters $\sigma \in \{0.25, 0.50, 1.00\}$. We notice that while simply applying MME or WE protocol could slightly improve the certified accuracy, DRT could significantly boost the certified accuracy under different radii. We also present the trends of different smoothing parameters separately in the full version [431, Appendix F] which lead to similar conclusions.

Table 4.3: Certified accuracy under different radii on ImageNet dataset. The grey rows present the performance of the proposed DRT approach. The brackets show the base models we use.

Radius r	0.00	0.50	1.00	1.50	2.00	2.50	3.00
Gaussian [77]	57.2	46.2	37.0	29.2	19.6	15.2	12.4
SmoothAdv [317]	54.6	49.0	43.8	37.2	27.0	25.2	20.4
MACER [437]	**68.0**	**57.0**	43.0	31.0	25.0	18.0	14.0
SWEEN (Gaussian) [234]	58.4	47.0	37.4	29.8	20.2	15.8	12.8
SWEEN (SmoothAdv) [234]	55.2	50.0	44.2	37.8	27.6	26.6	21.6
MME (Gaussian)	58.0	47.2	38.8	31.2	21.4	16.4	14.2
DRT + MME (Gaussian)	52.2	46.8	42.4	34.2	24.0	19.6	18.0
MME (SmoothAdv)	55.0	50.2	44.2	38.6	27.4	26.4	21.6
DRT + MME (SmoothAdv)	49.8	46.8	**44.4**	**39.8**	30.2	28.2	**23.4**
WE (Gaussian)	58.2	47.2	38.6	31.2	21.6	17.0	14.4
DRT + WE (Gaussian)	52.2	46.8	41.8	33.6	24.2	19.8	18.4
WE (SmoothAdv)	55.2	50.2	**44.4**	38.6	28.2	26.2	22.0
DRT + WE (SmoothAdv)	49.8	46.6	**44.4**	38.8	**30.4**	**29.0**	23.2



(a) MNIST (b) CIFAR-10

Figure 4.3: Certified accuracy for ML ensembles with Gaussian smoothed base models, under smoothing parameter $\sigma \in \{0.25, 0.50, 1.00\}$ on (Left) MNIST; (Right) CIFAR-10.

Effects of GD and CM Losses in DRT. To explore the effects of individual Gradient Diversity and Confidence Margin Losses in DRT, we set ρ_1 or ρ_2 to 0 separately and tune the other for evaluation on MNIST and CIFAR-10. The full results are shown in the full version [431, Appendix G.1]. We observe that both GD and CM losses have positive effects on improving the certified accuracy, and GD plays a major role on larger radii. By combining these two regularization losses as DRT does, the ensemble model achieves the highest certified accuracy under all radii.

4.4 SUMMARY

In this chapter, we explored and characterized the robustness conditions for certifiably robust ensemble ML models theoretically, and proposed DRT for training a robust ensemble. Our analysis provided the justification of the regularization-based training approach DRT. Extensive experiments showed that DRT-enhanced ensembles achieve the highest certified robustness compared with existing baselines.

Ethics Statement. In this chapter, we characterized the robustness conditions for certifying ML ensemble robustness. Based on the analysis, we propose DRT to train a certifiably robust ensemble. On the one hand, the training approach boosts the certified robustness of ML ensemble, thus significantly reducing the security vulnerabilities of ML ensemble. On the other hand, the trained ML ensemble can only guarantee its robustness under specific conditions of the attack. Specifically, we evaluate the trained ML ensemble on the held-out test set and constrain the attack to be within predefined ℓ_2 distance from the original input. We cannot provide robustness guarantee for all possible real-world inputs. Therefore, users should be aware of such limitations of DRT-trained ensembles, and should not blindly rely on the ensembles when the attack can cause large deviations measured by ℓ_2 distance. As a result, we encourage researchers to understand the potential risks, and evaluate whether our attack constraints align with their usage scenarios when applying our DRT approach to real-world applications. We do not expect any ethics issues raised by our work.

Reproducibility Statement. Our evaluation is conducted on commonly accessible MNIST, CIFAR-10, and ImageNet datasets. We upload the source code as the supplementary material for reproducibility purpose in `https://openreview.net/attachment?id=tUa4REjGjTf&name=supplementary_material`.

CHAPTER 5: TRAINING FOR ENHANCING WHITE-BOX CERTIFIED ROBUSTNESS: ROBUSTRA

Though certification approaches enable robustness guarantees for deep neural networks (DNNs), the robustness guarantee for normally-trained DNNs is usually vacuous, due to their lack of robustness and adaptation for certification approaches. In Chapter 4, we propose training approach DRT for randomized smoothing, a white-box smoothing-inference-based certification. How to train suitable DNNs for white-box certification such as GCP-CROWN? In this chapter, we propose an approach named Robustra for effectively improving the certified robustness for ReLU-based DNNs.

In Robustra, we formalize the target task as a min-max game between the attackers and defenders. Different from previous work [406] that solves the min-max optimization over the whole norm-bounded space which is challenging, we use the adversarial space of a reference model as the feasible region. Specifically, given a reference model, our formalized problem seeks to train a model that is robust in the adversarial space of the reference model, i.e., the adversarial examples cannot transfer to the model that we train. In this way, the model is adapted to "most vulnerable perturbations" at first, then gradually expands its robust region to the whole perturbation space, which eases the burden of optimization compared to adapting to the whole norm-bounded space at once.

Solving such a non-convex problem with constrained adversarial space is non-trivial. We use linear approximation for ReLU activations in the DNNs, and solve its dual problem. The constraint related to the reference adversarial space introduces a family of slack variables in the dual problem, causing no existing approach to effectively solve it. To address the problem, by leveraging the monotonicity and boundedness of the slack variables' gradients, only querying the gradients once, our approach directly calculates out the solution for slack variables. Thus, we solve the problem in the same order of time as the previous work [407]. Furthermore, by using similar techniques introduced in the previous work [407], Robustra is scalable to be applied to large models such as ResNet. Moreover, we train a pair of models mutually, i.e., iteratively using one as training model and the other as reference, and thus we obtain a pair of robust models at the same time, without requiring external reference.

We evaluate Robustra on the MNIST and CIFAR10 datasets. The evaluation results show that our approach can provide significantly better certified robust accuracy on both datasets, compared to the state-of-the-art results. Specifically, we improve the certified accuracy from 75.81% to 83.09% with ℓ_∞ norm $r = 0.3$ on MNIST, and from 53.89% to 56.32% with ℓ_∞ norm $r = 2/255$ on CIFAR10.

In summary, this chapter makes the following main contributions:

- We propose Robustra, a novel approach for training robust ReLU-based DNNs. Robustra formalizes the problem of robust training as a min-max optimization over the adversarial space of a reference model.
- We propose an algorithm that leverages the monotonicity and boundedness of the slack variables in the dual problem to efficiently solve the optimization problem.
- We evaluate Robustra on both MNIST and CIFAR10 datasets. The evaluation results show that Robustra produces DNNs with significantly better certified adversarial error bounds compared to the state-of-the-art results.

Our code and model weights are available at `https://github.com/llylly/Robustra`.

5.1 PRELIMINARIES

In this work, we consider a pair of ReLU-based DNNs f and g, where f is the trainable model, and g is the reference model. We refer to f as $f_{\boldsymbol{\theta}}$ in some places to explicitly emphasize the trainable parameters $\boldsymbol{\theta}$.

Let d denote the dimension of input vector, and C denote the number of classes. f and g are formally defined as follows respectively:

$$\begin{cases} \boldsymbol{z}_0 = \boldsymbol{x} \in \mathbb{R}^d, \\ \boldsymbol{z}_i = \mathrm{ReLU}(\boldsymbol{W}_i \boldsymbol{z}_{i-1} + \boldsymbol{b}_i), 1 \leqslant i \leqslant k_1 \\ f(\boldsymbol{x}) = \boldsymbol{z}_{k_1} \in \mathbb{R}^C. \end{cases} \qquad \begin{cases} \boldsymbol{w}_0 = \boldsymbol{x} \in \mathbb{R}^d, \\ \boldsymbol{w}_i = \mathrm{ReLU}(\boldsymbol{U}_i \boldsymbol{w}_{i-1} + \boldsymbol{d}_i), 1 \leqslant i \leqslant k_2 \\ g(\boldsymbol{x}) = \boldsymbol{w}_{k_2} \in \mathbb{R}^C. \end{cases} \tag{5.1}$$

Namely, f (resp. g): $\mathbb{R}^d \to \mathbb{R}^C$ is a k_1 (resp. k_2)-layer neural network with parameters $\boldsymbol{W}_i, \boldsymbol{d}_i, 1 \leqslant i \leqslant k_1$ (resp. $\boldsymbol{U}_i, \boldsymbol{b}_i, 1 \leqslant i \leqslant k_2$). All activations functions are ReLU functions in f and g.

Given an input $\boldsymbol{x}$, we have $f(\boldsymbol{x}), g(\boldsymbol{x}) \in \mathbb{R}^C$, representing the predicted confidence for each class. Let $\big(f(\boldsymbol{x})\big)_i$ (resp. $\big(g(\boldsymbol{x})\big)_i$) represent the i^{th} dimension of $f(\boldsymbol{x})$ (resp. $g(\boldsymbol{x})$), i.e., the predicted probability of class i.

In this chapter, we use ℓ_∞ norm to measure the adversarial perturbation since white-box approaches excel in certified robustness under ℓ_∞ norm (for black-box approaches, smoothing-based approaches like DRT in Chapter 4 excel). The approach can be extended to ℓ_1 or ℓ_2 norm by applying the techniques introduced in [407].

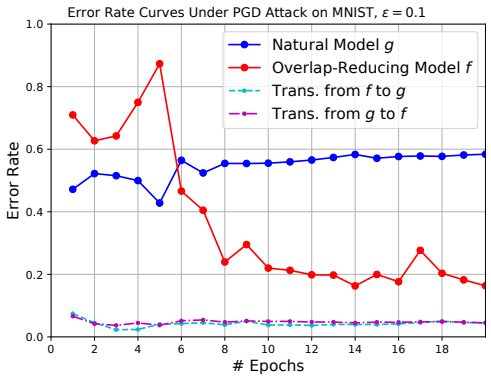


Figure 5.1: A heuristic training approach that aims at reducing the overlap of adversarial space to a reference model can make the model internally robust (see red curve).

5.2 RELATED WORK AND BACKGROUND

Attacks and Empirical Defenses. Szegedy et al. [359] discover the broad existence of adversarial examples in neural networks. From then on, various attack approaches have been proposed, such as FGSM [132], CW [53, 54], and PGD [251]. At the same time, heuristic defenses have been proposed, such as Distillation [288] and Defense-GAN [321]. Although these heuristic defenses have empirically shown effectiveness against existing attacks, they can be adaptively attacked again after being proposed [15, 52].

Certified Training Approaches for White-box Certification. Certified training approaches (or certified/provable defenses) train DNNs with high certified robustness accuracy, i.e., they can provably guarantee robustness for a fraction of test set inputs against any bounded attacks for the trained DNNs. Existing certified defenses for white-box approaches mainly use the upper bound of loss function as (part or all of) the training objective. These upper-bound inspired approaches include SDP [307], LP-dual [406, 407], and LP [262, 340]. Besides, concurrent work by Xiao et al. [419] combines PGD adversarial training with regularization for weight sparsity and ReLU stability. However, certified robust accuracies of DNNs obtained from these existing approaches are far from satisfaction.

5.3 MOTIVATION: AN EMPIRICAL OBSERVATION

Our approach is inspired by an empirical observation: during training, merely limiting overlap of the adversarial space with a reference model can make the model internally robust. This observation indicates that the adversarial space of one model should be defended against with high priority.

Specifically, given an input $\boldsymbol{x}$ and its true label y, a naturally trained reference model $g(\cdot)$, we train a model $f(\cdot)$ by modifying the training objective from original loss function $\mathcal{L}_f := \mathcal{L}\left(f(\boldsymbol{x}), y\right)$ to $\mathcal{L}_f + \lambda|\cos\langle\nabla_{\boldsymbol{x}}\ell_f, \nabla_{\boldsymbol{x}}\ell_g\rangle|$, where $\lambda = 0.1$ is a tunable hyperparameter and g, the reference model, is a naturally trained model. Our intuitive approach aims at guiding the local gradients between two models to be diverse (orthogonal) in terms of their cosine angle, hence reducing the overlap of their adversarial spaces.

Figure 5.1 shows the error (i.e., 100% - accuracy) curves by training epochs. The error rate is measured by the success rate of untargeted PGD attack bounded by ℓ_∞ with perturbation radius $r = 0.1$. The blue curve represents the natural trained model g, where as the natural training goes, the error rate slightly increases. The two dashed curves represent the transfer attack's success rate, where adversarial examples are generated from f/g model and used to attack the other model. The curves show that the transfer error rate is reduced as expected. The red curve represents the error rate of model f. We find that, after a few epochs, the error rate of model f decreases below 20% although we never deliberately train the model to be self-robust.

This observation inspires us to train a robust model constrained on the adversarial space of another model.

5.4 ROBUSTRA

In this section, we introduce Robustra in detail.

5.4.1 Min-Max Problem Formulation

We formalize our target task as a min-max game.

Given an input sample $\boldsymbol{x}^*$, the true label y^*, and some target label $y^{\text{targ}} \neq y^*$, we define $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$, the adversarial space of g bounded by ℓ_∞ norm with radius ϵ, as follows:

$$\mathcal{A}_\epsilon(\boldsymbol{x}^*) = \left\{\boldsymbol{x} : \left(\|\boldsymbol{x} - \boldsymbol{x}^*\|_\infty \leqslant \epsilon\right) \wedge \left(\left(g(\boldsymbol{x})\right)_{y^{\text{targ}}} - \left(g(\boldsymbol{x})\right)_{y^*} \geqslant t\right)\right\}, \tag{5.2}$$

where t measures the margin between the decision boundary of adversarial examples in $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$. Particularly, when $t = 0$, $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$ is the exact adversarial space of g; when t decreases, $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$ becomes larger, vice versa.

We expect the model f to be robust to adversarial examples of the reference model g. Specifically, for each input in the adversarial space $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$, the output of f has a larger

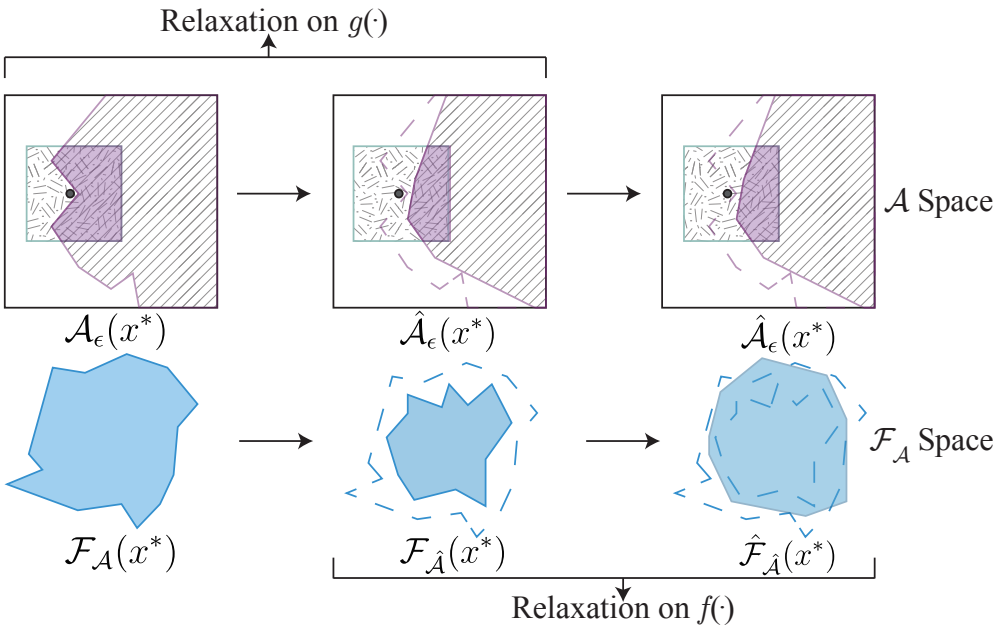


Figure 5.2: Visual illustration of approximation by linear convex relaxation. From the exact space $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$, the relaxation of model g brings convex space $\hat{\mathcal{A}}_\epsilon(\boldsymbol{x}^*)$. Then, the relaxation of model f brings convex space $\hat{\mathcal{F}}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*)$.

value at y^* than at y^{targ}. We formalize the objective as the following optimization problem:

$$\begin{aligned} \forall y^{\text{targ}} \neq y^*, \quad & \min_{\boldsymbol{\theta}} \left(\max_{\boldsymbol{x}} \left(f_{\boldsymbol{\theta}}(\boldsymbol{x})\right)_{y^{\text{targ}}} - \left(f_{\boldsymbol{\theta}}(\boldsymbol{x})\right)_{y^*} \right), \\ \text{s.t.} \quad & \boldsymbol{x} \in \mathcal{A}_\epsilon(\boldsymbol{x}^*). \end{aligned} \tag{5.3}$$

Let $c = e_{y^{\text{targ}}} - e_{y^*}$, and let $\mathcal{F}_\mathcal{A}$ denote the value space of $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$, i.e.,

$$\mathcal{F}_\mathcal{A}(x^*) = \left\{ f_{\boldsymbol{\theta}}(\boldsymbol{x}) : \boldsymbol{x} \in \mathcal{A}_\epsilon(\boldsymbol{x}^*) \right\}. \tag{5.4}$$

Then, the optimization (5.3) can be rewritten as follows:

$$\begin{aligned} \forall y^{\text{targ}} \neq y^*, \quad & \min_{\boldsymbol{\theta}} \max_{\boldsymbol{v}} \boldsymbol{c}^\top \boldsymbol{v}, \\ \text{s.t.} \quad & \boldsymbol{v} \in \mathcal{F}_\mathcal{A}(\boldsymbol{x}^*). \end{aligned} \tag{5.5}$$

Since the set $\mathcal{F}_\mathcal{A}(\boldsymbol{x}^*)$ is consecutive and highly non-convex, solving such optimization problem is computationally intractable.

5.4.2 Linear Convex Relaxation

To solve the optimization problem, we approximate the optimization space $\mathcal{F}_{\mathcal{A}}(\boldsymbol{x}^*)$ of the problem by using relaxation of the ReLU function. A convex space $\hat{\mathcal{F}}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*)$ is obtained from the approximation.

Since both $f(\cdot)$ and $g(\cdot)$ are ReLU-based DNNs, we use the linear relaxation for ReLU function. Let $o = \text{ReLU}(i)$ denote the ReLU function and $[l, u]$ denote the range of input i, i.e., $l \leqslant i \leqslant u$. The approximation of ReLU function corresponds to its convex outer space:

1. When $l \leqslant u \leqslant 0$, $o = 0$;
2. When $0 \leqslant l \leqslant u$, $o = i$;
3. When $l < 0 < u$, $(u - l)o \leqslant u(i - l) \wedge o \geqslant 0 \wedge o \geqslant i$.

We replace each ReLU function in the problem with corresponding linear constraints according to the range of the input.

We apply the relaxation to both $f(\cdot)$ and $g(\cdot)$ models, as shown in Figure 5.2.

First, we apply the relaxation to the ReLU neurons in model $g(\cdot)$. Let $\hat{G}(\cdot)$ denote the output space after applying the relaxation. Note that $\hat{G}(\boldsymbol{x}) \subseteq \mathbb{R}^C$, while $g(\boldsymbol{x}) \in \mathbb{R}^C$. Accordingly, the adversarial space $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$ is replaced with its approximation subspace $\hat{\mathcal{A}}_\epsilon(\boldsymbol{x}^*)$:

$$\hat{\mathcal{A}}_\epsilon(\boldsymbol{x}^*) = \left\{ \boldsymbol{x} : \left(\|\boldsymbol{x} - \boldsymbol{x}^*\|_\infty \leqslant \epsilon \right) \wedge \left(\forall \boldsymbol{v}' \in \hat{G}(\boldsymbol{x}) : \boldsymbol{c}^\top \boldsymbol{v}' \geqslant t \right) \right\}. \tag{5.6}$$

Note that $\hat{\mathcal{A}}_\epsilon(\boldsymbol{x}^*)$ is a convex subspace of $\mathcal{A}_\epsilon(\boldsymbol{x}^*)$, because $\hat{G}(\boldsymbol{x})$ is a convex outerspace of $g(\boldsymbol{x})$. Then, we obtain an approximated optimization subspace $\mathcal{F}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*)$ (shown in Figure 5.2).

Second, we similarly apply the relaxation to the ReLU neurons in model $f(\cdot)$. Let $\hat{F}(\cdot)$ denote the output space after applying the relaxation. Then, we obtain $\hat{\mathcal{F}}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*) = \bigcup_{\boldsymbol{x} \in \hat{\mathcal{A}}_\epsilon(\boldsymbol{x}^*)} \hat{F}(\boldsymbol{x})$, which is a convex approximation space of $\mathcal{F}_{\mathcal{A}}(\boldsymbol{x}^*)$.

After the two-step relaxation, the optimization problem that we would like to address is as follows:

$$\begin{aligned} \forall y^{\text{targ}} \neq y^*, \quad & \min_{\boldsymbol{\theta}} \max_{\boldsymbol{v}} \boldsymbol{c}^\top \boldsymbol{v}, \\ \text{s.t.} \quad & \boldsymbol{v} \in \hat{\mathcal{F}}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*). \end{aligned} \tag{5.7}$$

It is an approximation of the original optimization problem (5.5).

Figure 5.2 illustrates the procedure of the relaxation. At the beginning, the optimization space $\mathcal{F}_{\mathcal{A}}(\boldsymbol{x}^*)$ is non-convex; after relaxation of model g, we obtain a subspace $\mathcal{F}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*)$; after

relaxation of model f, we obtain $\hat{\mathcal{F}}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*)$, which is a convex approximation of the original space. We address the optimization problem over this approximation convex space.

5.4.3 Duality

Training on the outer minimization in Equation (5.7) requires to calculate the gradient of model parameter $\boldsymbol{\theta}$ w.r.t. the inner maximization problem: $\nabla_{\boldsymbol{\theta}} \max_{\boldsymbol{v}} \boldsymbol{c}^\top \boldsymbol{v} = \nabla_{\boldsymbol{\theta}} \boldsymbol{c}^\top \boldsymbol{v}^*$, where $\boldsymbol{v}^*$ is the solution to the inner problem. To calculate the gradient, efficiently solving the inner problem, i.e., finding $\boldsymbol{v}^*$, is desirable.

Since the inner maximization is an optimization over linear convex space $\hat{\mathcal{F}}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*)$, the minimum of its dual problem corresponds to the solution due to strong duality. Inspired by previous work [406, 407], we formulate the dual problem by a feedforward network as follows:

$$\max_{\boldsymbol{v}} \ \boldsymbol{c}^\top \boldsymbol{v} \leqslant \min_{\eta \geqslant 0} \ h_{\boldsymbol{c}}(\eta), \tag{5.8}$$

$$\begin{aligned} &\text{where } h_{\boldsymbol{c}}(\eta) \\ &= \ \boldsymbol{x}^\top (\hat{\boldsymbol{\nu}}_1 + \eta \hat{\boldsymbol{q}}_1) + \epsilon \|\hat{\boldsymbol{\nu}}_1 + \eta \hat{\boldsymbol{q}}_1\|_1 - \eta t \\ &\quad + \sum_{i=1}^{k_1-1} \boldsymbol{\nu}_{i+1}^\top \boldsymbol{b}_i + \eta \sum_{i=1}^{k_2-1} \boldsymbol{q}_{i+1}^\top \boldsymbol{d}_i - \sum_{i=2}^{k_1-1} \sum_{j \in \mathcal{I}_i} l_{i,j} [\nu_{i,j}]_+ - \eta \sum_{i=2}^{k_2-1} \sum_{j \in \mathcal{Q}_i} l'_{i,j} [q_{i,j}]_+. \end{aligned} \tag{5.9}$$

$\boldsymbol{\nu}$ and $\boldsymbol{q}$ are defined as follows:

$$\begin{aligned} \boldsymbol{\nu}_{k_1} &= -\boldsymbol{c} & \boldsymbol{q}_{k_2} &= -\boldsymbol{c} \\ \hat{\boldsymbol{\nu}}_i &= \boldsymbol{W}_i^\top \boldsymbol{\nu}_{i+1}, \quad \text{for } i = k_1 - 1, \ldots, 1 & \hat{\boldsymbol{q}}_i &= \boldsymbol{U}_i^\top \boldsymbol{q}_{i+1}, \quad \text{for } i = k_2 - 1, \ldots, 1 \\ \nu_{i,j} &= \begin{cases} 0 & j \in \mathcal{I}_i^- \\ \hat{\nu}_{i,j} & j \in \mathcal{I}_i^+ \\ (u_{i,j}/(u_{i,j} - l_{i,j}))\hat{\nu}_{i,j} & j \in \mathcal{I}_i, \end{cases} & q_{i,j} &= \begin{cases} 0 & j \in \mathcal{Q}_i^- \\ \hat{q}_{i,j} & j \in \mathcal{Q}_i^+ \\ (u'_{i,j}/(u'_{i,j} - l'_{i,j}))\hat{q}_{i,j} & j \in \mathcal{Q}_i, \end{cases} \\ & \text{for } i = k_1 - 1, \ldots, 2 & & \text{for } i = k_2 - 1, \ldots, 2 \end{aligned} \tag{5.10}$$

In (5.9) and (5.10), $u_{i,j}$ and $l_{i,j}$ represent the upper and lower bound of the ReLU function input for the j^{th} neuron in layer i of network $f(\cdot)$. Similarly, $u'_{i,j}$ and $l'_{i,j}$ represent the upper and lower bounds of the ReLU function input for the j^{th} neuron in layer i of network $g(\cdot)$. They are called intermediate layer bounds in Chapter 3 and literature [443, 448].

For layer i in model f and g, $\mathcal{I}_i^+$ and $\mathcal{Q}_i^+$ represent the sets of neurons where the inputs are always positive, respectively ($0 \leqslant l \leqslant u$); similarly, $\mathcal{I}_i^-$ and $\mathcal{Q}_i^-$ are the sets of neurons where the inputs are always negative ($l \leqslant u \leqslant 0$); and $\mathcal{I}_i$ and $\mathcal{Q}_i$ are the sets of neurons

spanning zero ($l < 0 < u$).

$\boldsymbol{\nu}, \hat{\boldsymbol{\nu}}, \boldsymbol{q}, \hat{\boldsymbol{q}}$, and η are newly introduced dual variables. We transform the original problem to an optimization problem over the new variable η.

5.4.4 Gradient Monotonicity and t Selection

To solve the dual problem $\min_{\eta \geqslant 0}\ h_{\boldsymbol{c}}(\eta)$, we turn to the gradient $\nabla_\eta h_{\boldsymbol{c}}(\eta)$, which can be written as follows:

$$\begin{aligned} & \nabla_\eta h_{\boldsymbol{c}}(\eta) \\ = \ & \epsilon \nabla_\eta \|\hat{\boldsymbol{\nu}}_1 + \eta \hat{\boldsymbol{q}}_1\|_1 - t + \boldsymbol{x}^\top \hat{\boldsymbol{q}}_1 + \sum_{i=1}^{k_2-1} \boldsymbol{q}_{i+1}^\top \boldsymbol{d}_i - \sum_{i=2}^{k_2-1} \sum_{j \in \mathcal{Q}_i} l'_{i,j} [q_{i,j}]_+ \qquad (5.11) \\ =: \ & \epsilon \gamma(\eta) - t + C, \end{aligned}$$

where $\gamma(\eta) = \sum_{j=1}^{n} \operatorname{sgn}(\hat{v}_{1,j} + \eta \hat{q}_{1,j}) \cdot \hat{q}_{1,j}$ and C is a constant independent of η and t. Note that $\gamma(\eta)$ is the analytic form of $\nabla_\eta \|\hat{\boldsymbol{\nu}}_1 + \eta \hat{\boldsymbol{q}}_1\|_1$.

Theorem 5.1 (Property of $\gamma(\eta)$)**.** $\gamma(\eta)$ is a bounded non-decreasing step-function, and the transition points are $\{-\hat{\nu}_{1,j}/\hat{q}_{1,j} : 1 \leqslant j \leqslant n\}$.

The theorem can be proved by case discussion over sign of $\hat{q}_{1,j}$. Since $\gamma(\eta)$ is the only η-dependent term in (5.11), $\nabla_\eta h_{\boldsymbol{c}}(\eta)$ has the same property, i.e., $\nabla_\eta h_{\boldsymbol{c}}(\eta)$ is a bounded non-decreasing step-function. Thus, we can obtain the value range of the gradient of $h_{\boldsymbol{c}}(\eta)$. Specifically, we have

Theorem 5.2 (Property of $h_{\boldsymbol{c}}(\eta)$)**.** When the domain of $h_{\boldsymbol{c}}(\eta)$ is $[0, \infty)$, let $\eta_l = 0, \eta_r = \max_j(-\hat{\nu}_{1,j}/\hat{q}_{1,j})$, the gradient value $\bigtriangledown_\eta h_{\boldsymbol{c}}(\eta) \in [l, r] := [\nabla_\eta h_{\boldsymbol{c}}(\eta)|_{\eta=\eta_l}, \nabla_\eta h_{\boldsymbol{c}}(\eta)|_{\eta=\eta_r}]$.

Let us define *adversarial space constraints* to be constraints imposed by the reference model, i.e., those corresponding to $\Big(\big(g(\boldsymbol{x})\big)_{y^{\text{targ}}} - \big(g(\boldsymbol{x})\big)_{y^*} \geqslant t\Big)$ in (5.2). We can relate the solution for $\min_{\eta \geqslant 0} h_{\boldsymbol{c}}(\eta)$ to sign of l and r:

- If $l \leqslant r \leqslant 0$, when $\eta \to +\infty$, $h_{\boldsymbol{c}}(\eta) \to -\infty$. According to the duality theorem, there is no solution to the original problem $\max_{\boldsymbol{v}}\ \boldsymbol{c}^\top \boldsymbol{v}$, indicating that the space $\hat{\mathcal{F}}_{\hat{\mathcal{A}}}(\boldsymbol{x}^*)$ is empty. In this situation, we regard adversarial space constraints to be *infeasible*. We need to decrease t, i.e., enlarge adversarial space $\hat{\mathcal{A}}_\epsilon(\boldsymbol{x}^*)$.

- If $l < 0 < r$, a positive η solution exists. In this situation, we regard adversarial space constraints to be *tight*.

- If $0 \leqslant l \leqslant r$, the solution is $\eta = 0$. As seen from (5.9), reference-model-related variables are all eliminated from objective function $h_{\boldsymbol{c}}(\eta)$, indicating that the adversarial space constraints are *loose*. In this situation, we would like to increase t, i.e., narrow adversarial space $\hat{\mathcal{A}}_\epsilon(\boldsymbol{x}^*)$.

The ideal situation is that for all optimization objectives $h_{\boldsymbol{c}}(\eta)$, the adversarial space constraints are tight and the solution exists, i.e., $l < 0 < r$. To guarantee so, we fix a reasonable η as the solution and inversely obtain the corresponding t.

Specifically, we set η to be the median number of $\gamma(\eta)$ transition points. Combining Theorem 5.1, Theorem 5.2, and constraint $\eta \geqslant 0$, we pick the median number in $\left\{-\hat{\nu}_{1,j}/\hat{q}_{1,j} : 1 \leqslant j \leqslant n, -\hat{\nu}_{1,j}/\hat{q}_{1,j} > 0\right\}$ as the solution for η, denoted as η_{med}.

η_{med} being the solution indicates $\nabla_\eta h_{\boldsymbol{c}}(\eta)|_{\eta=\eta_{\text{med}}} = 0$, which yields $t = t_0 = \epsilon\gamma(\eta_{\text{med}}) + C$. Thus, the hyperparameter t is set to t_0 according to η_{med}.

As result, we effectively solve the inner maximization problem $\min_{\eta\geqslant 0}\ h_{\boldsymbol{c}}(\eta)$ by forward passing c to the network defined in (5.10) and extracting the median number using $\hat{\boldsymbol{v}}_1$ and $\hat{\boldsymbol{q}}_1$. After that, we use t_0 and η_{med} to directly calculate $h_{\boldsymbol{c}}(\eta)$. Formally,

$$\max_{v} \boldsymbol{c}^\top \boldsymbol{v} \leqslant \min_{\eta\geqslant 0} h_{\boldsymbol{c}}(\eta) = h_{\boldsymbol{c}}(\eta)|_{\eta=\eta_{\text{med}}, t=t_0}. \tag{5.12}$$

5.4.5 Scaling

To scale up the training process, we find that Cauchy random projection estimation of ℓ_1 norm [407] can be directly applied in our approach. When a vector multiplies a standard Cauchy random matrix, the median number of the result vector is an estimation of ℓ_1 norm of the vector. Using this property, we can approximately calculate $\sum_{i=2}^{k_1-1}\sum_{j\in\mathcal{I}_i} l_{i,j}[\nu_{i,j}]_+$ and $\sum_{i=2}^{k_2-1}\sum_{j\in\mathcal{Q}_i} l'_{i,j}[q_{i,j}]_+$ of (5.9) in linear complexity w.r.t neuron numbers [407].

With this calculation, our approach has the same order of complexity with [407]. This calculation enables our approach to scale up to large datasets such as CIFAR10 and large DNN models such as ResNet.

5.4.6 Training

During the training process, we use the solution for each y^{targ} as the probability of that class. Then we use cross-entropy with softmax as the training loss to jointly minimize errors for all $y^{\text{targ}} \neq y^*$.

Table 5.1: Statistics of the models used in our experiments.

Dataset	Model	# Layers	# Hidden Units	# Parameters
MNIST	Small	4	4,804	166,406
	Large	7	28,064	1,974,762
CIFAR10	Small	4	6,244	214,918
	Large	7	62,464	2,466,858
	ResNet	11	107,496	4,214,850

We adopt a manual training scheme to optimize a pair of models. Initially, $f(\cdot)$ and $g(\cdot)$ are randomly initialized independently. When the model size is small, in each epoch, we train model $f(\cdot)$ where model $g(\cdot)$ is the reference, and then train model $g(\cdot)$ where model $f(\cdot)$ is the reference. When the model size is large, we accelerate the process by distributing model $f(\cdot)$ and model $g(\cdot)$ to two GPUs. By loading the last epoch weights of the other model as the reference, two models are trained parallelly, both using the other model as the reference at the same time. This training scheme optimizes both models efficiently.

5.5 EXPERIMENTS

We evaluate Robustra on image classification tasks with two datasets: MNIST [203] and CIFAR10 [191].

We use the same DNNs as used in [407] for comparison. All DNNs are convolutional neural networks with multiple layers, using only ReLU activations. Table 5.1 shows the statistics of DNNs that we use.

For each model and dataset, we run 100 epochs on the training set. Adam optimizer is used for MNIST models, and SGD optimizer (0.9 momentum, 5×10^{-4} weight decay) is used for CIFAR10 models. The perturbation radius ϵ under ℓ_∞ norm is initialized by 0.01, and then it linearly increases to the configured ϵ (0.1 or 0.3 for MNIST, 2/255 or 8/255 for CIFAR10) in the first 20 epochs. In the first 20 epochs, the learning rate is set to be 0.001; then, it decades by half every 10 epochs. The batch size is set to 50. The scaling approximation (Section 5.4.5) is applied for training 'Large' and 'ResNet' models, where the projection dimension is 50. Note that no hyperparameter selection or tuning is applied to guarantee the soundness of the results. All experiments are run on Geforce GTX 1080 Ti GPUs.

Table 5.2: Certified error bound of Robustra and the state of the art (**SOA**) [406, 407]. Two bounds are used for evaluation: *LP bound* [406] and *MIP bound* [368]. For SOA models, we use the error bounds reported in their chapters. We mark a '-' on an entry if there are no reported results in the corresponding chapter. For our models that are too large for *MIP bound* to be calculated in feasible time (i.e., 24 hours), we mark '/' on the entry. In addition, we compare clean example test error rate (Clean Err.) of SOA models and our models.

Dataset	**Perturbation Radius ϵ**	**Model**	*LP bound*		*MIP bound*		Clean Error	
			SOA	**Robustra**	**SOA**	**Robustra**	**SOA**	**Robustra**
MNIST	0.1	Small	**4.48**%	4.84%	4.38%	**2.55**%	1.26%	**1.01**%
		Large	**3.67**%	3.93%	2.74%	**2.09**%	1.08%	**0.83**%
	0.3	Small	43.10%	**31.42**%	25.79%	**16.91**%	11.40%	**7.37**%
		Large	45.66%	**34.59**%	**24.19**%	24.43%	**11.16**%	11.61%
CIFAR10	$\frac{2}{255}$	Small	52.75%	**51.47**%	50.20%	**47.93**%	38.91%	**36.03**%
		Large	46.59%	**43.68**%	-	/	31.28%	**28.48**%
		ResNet	46.11%	**44.43**%	-	/	31.72%	**29.51**%
	$\frac{8}{255}$	Small	79.25%	**76.29**%	-	**74.87**%	72.24%	**66.34**%
		Large	83.43%	**78.32**%	-	**77.56**%	80.56%	**71.79**%
		ResNet	**78.22**%	79.57%	**77.60**%	78.10%	**71.33**%	72.10%

5.5.1 Certified Error Bound

Under the given perturbation radius ϵ, we evaluate trained models in terms of the *certified error bound*, which stands for the ratio of test points whose robustness cannot be certified to be robust. Certified error bound is an upper bound of maximum error rate that can be achieved by ϵ-radius ℓ_∞-bounded adversarial attack, and it equals to (100% - certified robust accuracy) where certified robust accuracy is as defined in Chapter 4. We use two certification approaches to evaluate the models trained by Robustra: LP relaxation (denoted as *LP bound*) [406] and MIP solver (denoted as *MIP bound*) [368]. We remark that with the latest certification approaches like GCP-CROWN we can expect lower absolute values of error bounds, but the relative tendency should be similar. The full test set is used in our evaluation. We compare the error bounds of Robustra to the state of the art on the same model structures [406, 407].

The results in Table 5.2 show that on both the MNIST dataset and CIFAR10 dataset, Robustra outperforms the state of the art in most of the settings. Specifically, on the MNIST dataset, with $\epsilon = 0.1$, Robustra significantly improves the *MIP bound* on 'Large' model from 2.74% to 2.09%; with $\epsilon = 0.3$, Robustra achieves 16.91% certified test error using 'Small' model and improves *LP bound*, from previous 43.10%/45.66% to 31.42%/34.59% for 'Small' and 'Large' models, respectively. On the CIFAR10 dataset, with $\epsilon = 2/255$, Robustra exceeds the state of the art on all models. It reduces the previous bound from 46.11% to 43.68%. With $\epsilon = 8/255$, Robustra reduces the error bound from 77.60% (on 'ResNet', *MIP*

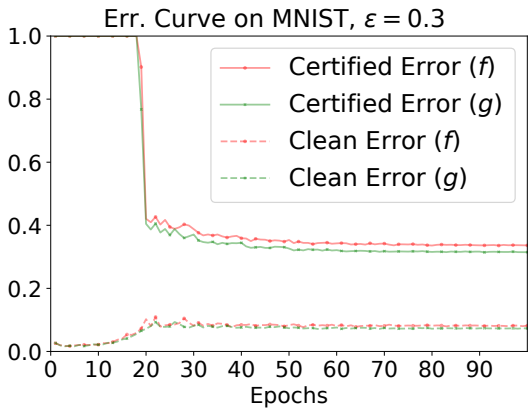


Figure 5.3: Certified error and clean error curves of each epoch, evaluated on the MNIST test set, with $\epsilon = 0.3$. Certified error stands for *LP bound*. Clean error stands for the standard error rate on clean examples. $f(\cdot)$ and $g(\cdot)$ stand for the two mutually trained models.

bound) to 74.87% (on 'Small', *MIP bound*).

Clean error represents the error rate of a model on a clean (i.e., unperturbed) test set. The model trained by Robustra has similar or smaller clean error rate compared to state-of-the-art models. Furthermore, we find that a model with a smaller clean error usually has a tighter certified error bound.

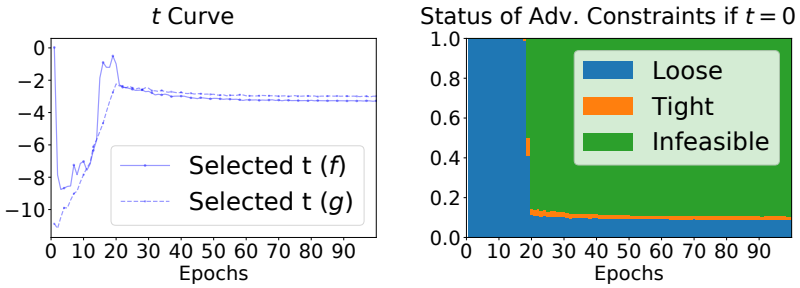
5.5.2 Training Progress

The training curves are shown in Figure 5.3. Certified error is measured by *LP bound*. Clean error is measured by error rate on clean (i.e., unperturbed) examples. Both are evaluated on the test set.

In the first 20 epochs, ϵ used for training is linearly increased to desired ϵ (0.3). The certified robust error keeps being 100% for these first 20 epochs, and then a significant drop occurs as the training ϵ reaches 0.3. The result reveals that the model is not robust to larger perturbations unless it is trained at that or larger perturbation level. During these first 20 epochs, the clean error increases while the certified robust error bound decreases. Then both certified errors and clean errors decrease.

At each epoch, both models are trained mutually: we first regard $g(\cdot)$ as the reference model and train the model $f(\cdot)$; then, we set the trained $f(\cdot)$ as the reference model and train the model $g(\cdot)$. Thus, at each epoch, the test error of $g(\cdot)$ is slightly smaller than the test error of $f(\cdot)$, as shown in Figure 5.3.

5.5.3 t Selection Effectiveness



(a) (b)

Figure 5.4: On MNIST with $\epsilon = 0.3$: (a) curve of selected t; and (b) tightness status of adversarial space constraints of each epoch when $t = 0$.

In our approach, we select some t to solve the inner maximization problem, as discussed in Section 5.4.4. Figure 5.4(a) plots the curve of selected t. Larger t corresponds tighter reference space, as shown in (5.2). In the first 20 epochs, t is small and sharply increases to around -3. This result indicates that our approach dynamically tightens these constraints along with increasing training ϵ. After 20 epochs, as the reference model is trained more robust, its adversarial space becomes smaller. The approach gradually decreases t to loosen the constraints.

In Figure 5.4(b), on MNIST with $\epsilon = 0.3$, we study the solution status on test dataset if fixing $t = 0$. In the first 20 epochs, non-robust models make almost all constraints loose. After 20 epochs, the robust models make almost all constraints infeasible. Thus, when $t = 0$, we are not able to effectively solve the problem. Instead, we need to dynamically select t as we propose in Section 5.4.4.

5.6 SUMMARY

We have presented Robustra, an approach for training certified robust neural networks. In particular, we formalize the training procedure as a min-max game over the adversarial space of the reference model, and effectively solve the min-max game. The experimental results have shown that Robustra significantly outperforms existing approaches.

Our approach exploits the adversarial space of the reference model. Instead of optimizing over the whole perturbation space, we instead consider only the overlap of the perturbation space and transferable space. By leveraging linear approximation and monotonicity of slack variables, we solve the dual problem efficiently.

However, theoretical analysis of the reasons why training over reference space produces more robust models is desirable. Currently, the performance of existing models with certified bounds on CIFAR10 still has a relatively large gap compared with vanilla training. Further improvement on CIFAR10 remains the future work. In future work, we plan to derive theoretical evidence for why training over transferable spaces could get better certified robustness. Our approach is still relatively slow, and some certified bounds are still unsatisfactory such as that of CIFAR10 when $\epsilon = 8/255$. Further improving our approach remains our future work.

CHAPTER 6: DISCUSSION FOR PART II

In this part, we first proposed two certification approaches DSRS and DRT. The former is a black-box certification approach and requires smoothing as the post-processing inference protocol. The latter is a white-box certification approach based on branch-and-bound and cutting plane for tightening the relaxation. Then, we proposed two certified training approaches DRT and Robustra. The former is suitable for black-box certification, and the latter is suitable for white-box certification. Both training approaches share the idea of reducing transferability among models. The former reduces the first-order transferability by encouraging gradient diversity and leverages smoothing to bound higher-order uncertainties. The latter constrains the model training within the adversarial space of a reference model to ease the optimization burden. As we can see, reducing transferability is the principle but leveraged in different ways to cater to certification approaches.

In following sections, we will briefly discuss the whole research field of certified approaches against ℓ_p-bounded perturbations.

6.1 PRACTICAL IMPLICATIONS

In practice, what are the most suitable certified approaches for users to deploy? Based on the above results and our leaderboard in `https://sokcertifiedrobustness.github.io/`, we present practical implications in Figure 6.1, where we envision two scenarios: 1) users want to improve certified robustness for their tasks at hand; 2) users want to evaluate or certify the robustness of given models.

When users want to improve certified robustness for their tasks, they need to achieve this by choosing a certified training approach and certifying the robustness with the corresponding certification approach as discussed in Section 1.1. The upper part of Figure 6.1 shows the recommended combinations of certification and certified training approaches in light gray boxes. Depending on the dataset size and the type of ℓ_p adversary to defend against, we recommend the corresponding approach combinations which achieve high certified accuracy in practice based on our leaderboard. When multiple choices are available, we show the top-2 choices and label them with "①" and "②" respectively. In summary, linear-relaxation-based verification is suitable only on small datasets against ℓ_∞ adversary, Lipschitz-based (including smooth layers, Lipschitz, and general Lipschitz) verification is suitable on small and medium datasets, and smoothed DNNs (including Neyman-Pearson-based and differential-privacy-based) are suitable on medium and large datasets. In particular, on large datasets

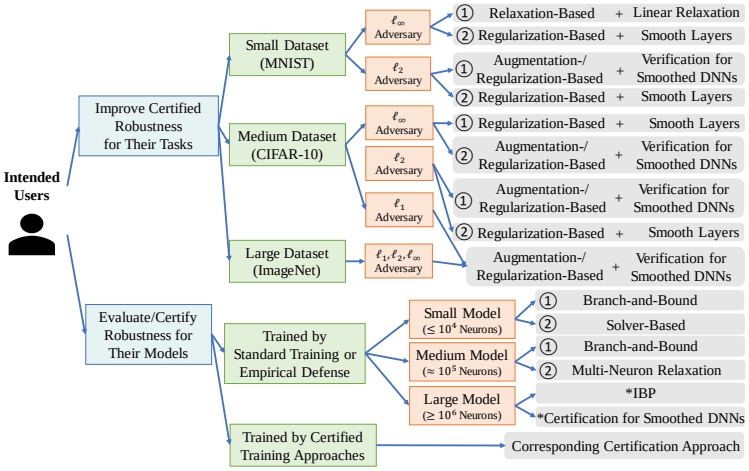


Figure 6.1: Practical implications for users to select certified approaches. Gray boxes depict suitable "(certification) + (certification training)" approach combinations or certified approaches for given scenarios. If exist, "①" and "②" label the most and runner-up suitable ones. Details in Section 6.1.

like ImageNet, only approaches for smoothed DNNs, like [77] and DSRS, can provide robustness certification at the current stage.

When users want to evaluate or certify the robustness of certain models, they need to choose a suitable certificaiton approach. Inspired by our benchmark, we present the implications in the lower part of Figure 6.1. For small and medium models trained by standard training or empirical defenses, the branch-and-bound based complete verification ([115, 389], GCP-CROWN) and multi-neuron relaxation verification [286] can certify the robustness efficiently. Specifically, for small models, the solver-based (concretely, MIP-based) certification approaches can certify good robustness. But for large models, none of these methods can finish in a feasible time (one day per input). Therefore, we must use more efficient but loose certification such as interval bound propagation, which usually yields trivial certified robustness radius and it is an active research area to make tighter certification approaches scalable for these large models. In addition, we find that the ranking of empirical defenses for small/medium models based on certified robustness is consistent with that evaluated by strong empirical attacks such as PGD [89, 389]. If models are trained by robust training approaches, using the corresponding certification approaches targeted by the training approach would be the best choice.

6.2 CHARACTERISTICS, STRENGTHS, LIMITATIONS, AND CONNECTIONS

To reveal the fundamental connections, we adopt a unified view of robustness certification: all existing certification approaches provide an abstraction of given DL models to verify the robustness. For example, the branch-and-bound verification views the model as the union of several sub-domains where the model output in each domain can be bounded, e.g., by linear inequalities. The branching process is essentially refining the abstraction by splitting sub-domains whose current abstractions are not precise enough. The linear relaxation-based certification uses some linear constraints to abstract the possible behavior of the model in the whole perturbation region. The smoothing-based certification uses queried information, such as zeroth-order information, to abstract the model behavior. This view is closely related to the concept of abstract interpretation in traditional program analysis [80]. Therefore, the scalability and tightness trade-off of certification mentioned in Section 1.1 is essentially the inherent trade-off between preciseness and efficiency of abstraction: more precise abstraction enables tighter robustness certification, whereas has higher time and space complexity. Thus, for a model that is not specifically trained, the most suitable certification approach is the most precise one that can be computed for this model size. We note that, under this unified view, the favored properties of each certification are tight conditions of the corresponding abstraction domain. Thus, robust training approaches that promote these properties can boost certification tightness for the model to improve certified robustness.

6.3 CHALLENGES AND BARRIERS

Although there has been remarkable progress towards certifiably robust DL systems, scalability and tightness challenges persist. For example: (1) Complete verification is NP-complete [180, 402]. (2) Multi-neuron based linear relaxation needs exponential number of constraints [367]. (3) Smoothing-based certification based on zeroth-order information cannot certify high robustness against ℓ_∞ adversary for real-world high-dimensional inputs [38, 195, 269, 427], where DSRS may be the key but further improvement room exists. These theoretical barriers are intrinsic challenges for further improvements in these certification approaches. There are also practical issues to solve, such as guaranteeing certification soundness under floating-point arithmetic [169, 381, 468] and safeguarding robust training against training-time attacks [260].

6.4 FUTURE DIRECTIONS

Despite the challenges and barriers, there are also several potential future directions. Here we name a few promising directions.

(1) **Scalable and tight verification:** There are still hopes for more scalable and tighter verification for DNNs *in practice* despite theoretical barriers. For example, good heuristics have boosted complete verification to handle DNNs with over 10^5 neurons [389]. It is promising to explore other better heuristics. For instance, a recent work [115] improves the complete verification by proposing better bounding heuristics based on multi-neuron relaxation. GCP-CROWN also indicates a very promising research thread by migrating research results from long-studied MIP solvers. For SDP verification, better formulation and solvers can lead to better certification [27, 89]. For smoothed DNNs, although only using zeroth-order information cannot certify high robustness against ℓ_∞ adversary, this barrier may be circumvented by leveraging more information as DSRS.

(2) **Effective certified training with theoretical understanding:** Unlike certification where theoretical barriers exist, certified training can empirically boost the certified robustness without known theoretical limitations. Indeed, even the empirically loose interval relaxations are universal approximators [17, 396] and achieve training convergence (under some assumptions) [394], which implies that with effective and generalizable robust training the certified accuracy could be on par with benign accuracy. However, theoretical understanding of robust training, such as why robust training generalizes, is still lacking [176]. Recent work shows that when an efficient complete certification approach exists, generalizable robust training is achievable [13]. Extending this result to broader scenarios, e.g., the generalization of robust training with *incomplete* certification, would significantly advance our understanding of ML robustness.

(3) **Design certifiably robust DNN architectures**: Based on the model properties required for different verification approaches, it is promising to leverage novel DNN architectures, e.g., Lipschitz layers, binarized layers, and attention modules, to further improve the certified robustness. In addition, it is also possible to design sparse DNNs following the model compression literature [326] to achieve efficient and certifiably robust models.

(4) **Integrate domain knowledge and logic reasoning ability into ML to improve**

certified robustness: It has been shown that joint inference with knowledge rules can improve model benign accuracy [305, 422, 458], and therefore it would be promising to integrate domain knowledge, causal analysis, and security rules into ML pipeline to further improve and tighten its end-to-end certified robustness [450].

Part III

Robustness Certification beyond ℓ_p-bounded Perturbations

In Part II, we propose certified approaches for certifying and improving the robustness of DL systems against ℓ_p-bounded perturbations. However, the threat model of ℓ_p-bounded perturbations is too restrictive to cover real-world attacks. Existing studies (as will introduce in the following chapters) have shown that physically realizable input perturbations can either incur large ℓ_p difference or persist in multiple time frames, but remain imperceptible to human and maintain high attack success rate. Hence, it is of practical demand to generalize the certified approaches to cover these real-world threats. In this part, we propose two types of generalization in following chapters respectively:

(1) For certified robustness against semantic transformations (threat model defined in Section 1.2.2), we propose TSS for image models in Chapter 7 and briefly discuss its extension to point cloud models in Appendix A; and

(2) For certified robustness in reinforcement learning (threat models defined in Sections 1.2.3 and 1.2.4), we propose CROP against observation perturbations in Chapter 8 and briefly discuss its extension to data poisoning attacks in Chapter 9.

All these approaches are generalized from certified approaches in Part II but involve non-trivial techniques to handle the threat-specific challenges. Though these approaches cannot cover the full set of existing robustness threats, they are ideal examples for extending certification beyond ℓ_p-bounded perturbation robustness, and research status for other threat models in the literature is briefly visited in Chapter 10.

CHAPTER 7: ROBUSTNESS AGAINST SEMANTIC TRANSFORMATIONS: TSS

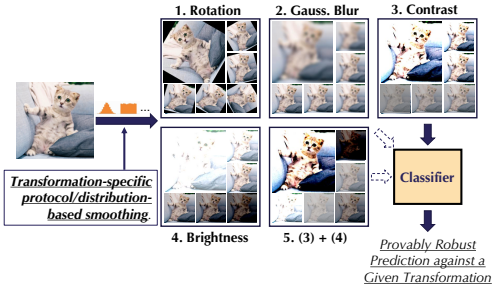


Figure 7.1: TSS: ***T****ransformation-****S****pecific* ***S****moothing-based robustness certification*, a general robustness certification framework against various semantic transformations. We develop a range of different transformation-specific smoothing protocols and various techniques to provide substantially better certified robustness bounds than state-of-the-art approaches on large-scale datasets.

The existing practice of security in ML has fallen into the cycle where new empirical defense techniques are proposed [249, 370], followed by new adaptive attacks breaking these defenses [15, 112, 132, 416]. In response, recent research has attempted to provide *certifiable robustness guarantees* for an ML model. Such certification usually follows the form that the ML model is provably robust against arbitrary adversarial attacks, as long as the perturbation magnitude is below a certain threshold. Different certifiable defenses and robustness verification approaches have provided nontrivial robustness guarantees against ℓ_p perturbations where the perturbation is bounded by small ℓ_p norm [77, 221, 368, 406, 424].

However, certifying robustness only against ℓ_p perturbations is not sufficient for attacks based on semantic transformation. For instance, image rotation, scaling, and other semantic transformations are able to mislead ML models effectively [110, 124, 130, 416]. These transformations are common and practical [46, 144, 291]. For example, it has been shown [153] that brightness/contrast attacks can achieve 91.6% attack success on CIFAR-10, and 71%-100% attack success rate on ImageNet [144]. In practice, brightness- and contrast-based attacks have been demonstrated to be successful in autonomous driving [291, 366]. These attacks incur large ℓ_p-norm differences and are thus beyond the reach of existing certifiable defenses [38, 141, 195, 318]. *Can we provide provable robustness guarantees against these semantic transformations?*

In this paper, we propose theoretical and empirical analyses to certify the ML robustness against a wide range of semantic transformations beyond ℓ_p bounded perturbations. The theoretical analysis is nontrivial given different properties of the transformations, and our

empirical results set the new state-of-the-art robustness certification for a range of semantic transformations, exceeding existing work by a large margin. In particular, we propose TSS: **T**ransformation-**S**pecific **S**moothing-based robustness certification — a *general framework* based on function smoothing providing certified robustness for ML models against a range of adversarial transformations (Figure 7.1). To this end, we first categorize semantic transformations as either *resolvable* or *differentially resolvable*. We then provide certified robustness against *resolvable* transformations, which include brightness, contrast, translation, Gaussian blur, and their composition. Second, we develop novel certification techniques for *differentially resolvable* transformations (e.g., rotation and scaling), based on the *building block* that we have developed for resolvable transformations.

For resolvable transformations, we leverage the framework to *jointly* reason about (1) function smoothing under different smoothing distributions and (2) the properties inherent to each specific transformation. To our best knowledge, this is the first time that the interplay between smoothing distribution and semantic transformation has been analyzed as existing work [77, 218, 427] that studies different smoothing distributions considers only ℓ_p perturbations. Based on this analysis, we find that against certain transformations such as Gaussian blur, exponential distribution is better than Gaussian smoothing, which is commonly used in the ℓ_p-case.

For differentially resolvable transformations, such as rotation, scaling, and their composition with other transformations, the common **challenge** is that they naturally induce *interpolation error*. Existing work [22, 118] can provide robustness guarantees but it cannot rigorously certify robustness for ImageNet-scale data. We develop a collection of novel techniques, including stratified sampling and Lipschitz bound computation to provide a tighter and sound upper bound for the interpolation error. We integrate these novel techniques into our TSS framework and further propose a progressive-sampling-based strategy to accelerate the robustness certification. We show that these techniques comprise a scalable and general framework for certifying robustness against differentially resolvable transformations.

We conduct extensive experiments to evaluate the proposed certification framework and show that our framework significantly outperforms the state-of-the-art on different datasets including the large-scale ImageNet against a series of practical semantic transformations. In summary, this chapter makes the following *contributions*:

(*1*) We propose a *general* function smoothing framework, TSS, to certify ML robustness against general semantic transformations.

(*2*) We categorize common adversarial semantic transformations in the literature into *resolvable* and *differentially resolvable* transformations and show that our framework is

general enough to certify both types of transformations.

(*3*) We theoretically explore different smoothing strategies by sampling from different distributions including non-isotropic Gaussian, uniform, and Laplace distributions. We show that for specific transformations, such as Gaussian blur, smoothing with exponential distribution is better.

(*4*) We propose a pipeline, **TSS-DR**, including a stratified sampling approach, an effective Lipschitz-based bounding technique, and a progressive sampling strategy to provide rigorous, tight, and scalable robustness certification against differentially resolvable transformations such as rotation and scaling.

(*5*) We conduct extensive experiments and show that our framework TSS can provide significantly higher certified robustness compared with the state-of-the-art approaches, against a range of semantic transformations and their composition on MNIST, CIFAR-10, and ImageNet.

(*6*) We show that TSS also provides much higher empirical robustness against adaptive attacks and unforeseen corruptions such as CIFAR-10-C and ImageNet-C.

The code implementation and all trained models are publicly available at `https://github.com/AI-secure/semantic-randomized-smoothing`.

7.1 BACKGROUND

We next provide an overview of different semantic transformations and explain the intuition behind the randomized smoothing [77] that has been proposed to certify the robustness against ℓ_p perturbations.

Semantic Transformation Based Attacks. Beyond adversarial ℓ_p perturbations, a realistic threat model is given by image transformations that preserve the underlying semantics. Examples for these types of transformations include changes to contrast or brightness levels, or rotation of the entire image. These attacks share three common characteristics: (1) The perturbation stemming from a successful semantic attack typically has higher ℓ_p norm compared to ℓ_p-bounded attacks. However, these attacks still preserve the underlying semantics (a car image rotated by 10° still contains a car). (2) These attacks are governed by a low-dimensional parameter space. For example, the rotation attack chooses a single-dimensional rotation angle. (3) Some of such adversarial transformations would lead to high

interpolation error (e.g., rotation), which makes it challenging to certify. Nevertheless, these types of attacks can also cause significant damage [144, 153] and pose realistic threats for practical ML applications such as autonomous driving [291]. We remark that our proposed framework can be extended to certify robustness against other attacks sharing these characteristics even beyond the image domain, such as GAN-based attacks against ML based malware detection [154, 390], where a limited dimension of features of the malware can be manipulated in order to preserve the malicious functionalities and such perturbation usually incurs large ℓ_p differences for the generated instances.

Randomized Smoothing. On a high level, randomized smoothing [77, 204, 218, 224] provides a way to certify robustness based on randomly perturbing inputs to an ML model. The intuition behind such randomized classifier is that noise smoothens the decision boundaries and suppresses regions with high curvature. Since adversarial examples aim to exploit precisely these high curvature regions, the vulnerability to this type of attack is reduced. Formally, a base classifier h is smoothed by adding noise ε to a test instance. The prediction of the smoothed classifier is then given by the most likely prediction under this smoothing distribution. Subsequently, a tight robustness guarantee can be obtained, based on the noise variance and the class probabilities of the smoothed classifier. It is guaranteed that, as long as the ℓ_2 norm of the perturbation is bounded by a certain amount, the prediction on an adversarial vs. benign input will stay the same. This technique provides a powerful framework to study the robustness of classification models against adversarial attacks for which the primary figure of merit is a low ℓ_p norm with a simultaneously high success rate of fooling the classifier [106, 427]. However, semantic transformations incur large ℓ_p perturbations, which renders classical randomized smoothing infeasible [38, 141, 195], making it of great importance to generalize randomized smoothing to this kind of threat model.

7.2 THREAT MODEL & TSS OVERVIEW

In this section, we first introduce the notations used throughout this chapter. We then define our **threat model**, the **defense goal** and outline the **challenges** for certifying the robustness against semantic transformations. Finally, we will provide a brief **overview** of our TSS certification framework.

We denote the space of inputs as $\mathcal{X} \subseteq \mathbb{R}^d$ and the set of labels as $\mathcal{Y} = \{1, \ldots, C\} = [C]$ (where $C \geqslant 2$ is the number of classes). The set of transformation parameters is given by $\mathcal{Z} \subseteq \mathbb{R}^m$ (e.g., rotation angles). We use the notation $\mathcal{P}_X$ to denote the probability measure induced by the random variable X and write f_X for its probability density function. For

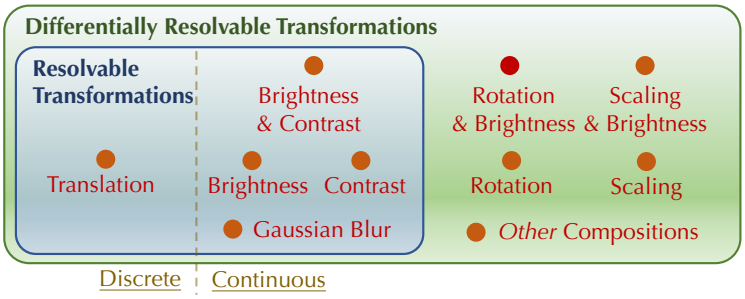


Figure 7.2: We provide strong robustness certification for both resolvable transformations and differentially revolvable transformations. These two categories cover common adversarial semantic transformations.

a set S, we denote its probability by $\Pr_X(S)$. A classifier is defined to be a deterministic function h mapping inputs $x \in \mathcal{X}$ to classes $y \in \mathcal{Y}$. Formally, a classifier learns a conditional probability distribution $p(y|\, x)$ over labels and outputs the class that maximizes p, i.e., $h(x) = \arg\max_{y \in \mathcal{Y}} p(y|\, x)$.

7.2.1 Threat Model and Certification Goal

Semantic Transformations. We model semantic transformations as deterministic functions $\phi\colon \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$, transforming an image $\boldsymbol{x} \in \mathcal{X}$ with a $\mathcal{Z}$-valued parameter α. For example, we use $\phi_R(\boldsymbol{x}, \alpha)$ to model a rotation of the image $\boldsymbol{x}$ by α degrees counter-clockwise with bilinear interpolation. We further partition semantic transformations into two different categories, namely resolvable and differentially resolvable transformations. We will show that these two categories could cover commonly known semantic attacks. This categorization depends on whether or not it is possible to write the composition of the transformation ϕ with itself as applying the same transformation just once, but with a different parameter, i.e., whether for any $\alpha, \beta \in \mathcal{Z}$ there exists γ such that $\phi(\phi(\boldsymbol{x}, \alpha), \beta) = \phi(\boldsymbol{x}, \gamma)$. Precise definitions are given in Sections 7.4 and 7.5. Figure 7.2 presents an overview of the transformations considered in this work.

Threat Model. As briefly defined in Section 1.2.2, We consider an adversary that launches a semantic attack, a type of data evasion attack, against a given classification model h by applying a semantic transformation ϕ with parameter α to an input image $\boldsymbol{x} \to \phi(\boldsymbol{x}, \alpha)$. We allow the attacker to choose an *arbitrary* parameter α within a predefined (attack) parameter space $\mathcal{S}$. For instance, a naïve adversary who randomly changes brightness from within $\pm 40\%$ is able to reduce the accuracy of a state-of-the-art ImageNet classifier from

74.4% to 21.8% (Table 7.3). While this attack is an example random adversarial attack, our threat model also covers other types of semantic attacks and we provide the first taxonomy for semantic attacks (i.e., resolvable and differentially resolvable) in detail in Sections 7.4 and 7.5.

Certification Goal. Since the only degree of freedom that a semantic adversary has is the parameter, **our goal** is to characterize a set of parameters for which the model under attack is guaranteed to be robust. Formally, we wish to find a set $\mathcal{S}_{\text{adv}} \subseteq \mathcal{Z}$ such that, for a classifier h and adversarial transformation ϕ, we have

$$h(\boldsymbol{x}) = h(\phi(\boldsymbol{x}, \alpha)) \quad \forall\, \alpha \in \mathcal{S}_{\text{adv}}. \tag{7.1}$$

Challenges for Certifying Semantic Transformations. Certifying ML robustness against semantic transformations is nontrivial and requires careful analysis. We identify the following two main challenges that we aim to address in this chapter:

(C1) The absolute difference between semantically transformed images in terms of ℓ_p-norms is typically high. This factor causes existing certifiable defenses against ℓ_p bounded perturbations to be inapplicable [38, 141, 195, 318].

(C2) Certain semantic transformations incur additional *interpolation errors*. To derive a robustness certificate, it is required to bound these errors, an endeavour that has been proven to be hard both analytically and computationally. This challenge applies to transformations that involve interpolation, such as rotation and scaling.

We remark that it is in general not feasible to use brute-force approaches such as grid search to enumerate all possible transformation parameters (e.g., rotation angles) since the parameter spaces are typically continuous. Given that different transformations have their own unique properties, it is crucial to provide a *unified* framework that takes into account transformation-specific properties in a general way.

To address these challenges, we generalize randomized smoothing via our proposed *function smoothing framework* to certify arbitrary input transformations via different smoothing distributions, paving the way to robustness certifications that go beyond ℓ_p perturbations. This result addresses challenge **(C1)** in a unified way. Based on this generalization and depending on specific transformation properties, we address challenge **(C2)** and propose a series of smoothing strategies and computing techniques that provide robustness certifications for a diverse range of transformations.

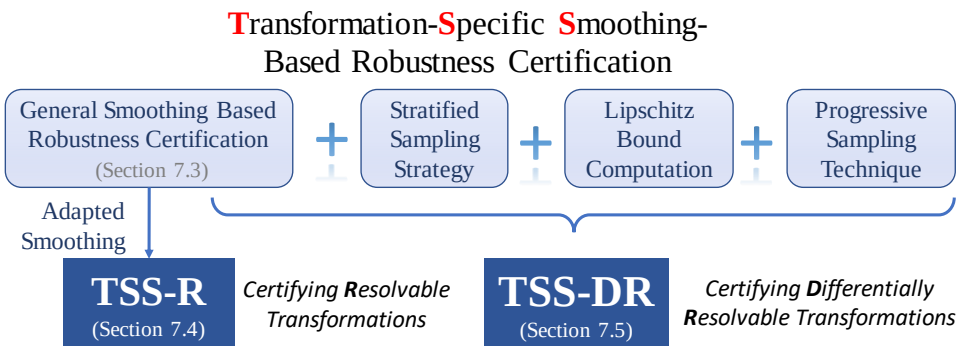


Figure 7.3: An overview of TSS.

We next introduce our generalized function smoothing framework and show how it can be leveraged to certify semantic transformations. We then categorize transformations as either *resolvable* transformations (Section 7.4) such as Gaussian blur, or *differentially resolvable* transformations (Section 7.5) such as rotations.

7.2.2 Framework Overview

An overview of our proposed framework TSS is given in Figure 7.3. We propose the function smoothing framework, a generalization of randomized smoothing, to provide robustness certifications under general smoothing distributions (Section 7.3). This generalization enables us to smooth the model on specific transformation dimensions. We then consider two different types of transformation attacks. For *resolvable* transformations, using function smoothing framework, we adapt different smoothing strategies for specific transformations and propose **TSS-R** (Section 7.4). We show that some smoothing distributions are more suitable for certain transformations. For *differentially resolvable* transformations, to address the interpolation error, we combine function smoothing with the proposed stratified sampling approach and a novel technique for Lipschitz bound computation to compute a rigorous upper bound of the error. We then develop a progressive sampling strategy to accelerate the certification. This pipeline is termed **TSS-DR**, and we provide details and the theoretical groundwork in Section 7.5.

7.3 TSS: TRANSFORMATION SPECIFIC SMOOTHING BASED CERTIFICATION

In this section, we extend randomized smoothing and propose a function smoothing framework TSS (**T**ransformation-**S**pecific **S**moothing-based robustness certification) for certifying

robustness against semantic transformations. This framework constitutes the main building block for **TSS-R** and **TSS-DR** against specific types of adversarial transformations.

Given an arbitrary base classifier h, we construct a smoothed classifier g by randomly transforming inputs with parameters sampled from a smoothing distribution. Specifically, given an input $\boldsymbol{x}$, the smoothed classifier g predicts the class that h is most likely to return when the input is perturbed by some random transformation. We formalize this intuition in the following definition.

Definition 7.1 (ε-Smoothed Classifier)**.** Let $\phi\colon \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ be a transformation, $\varepsilon \sim \mathcal{P}_\varepsilon$ a random variable taking values in $\mathcal{Z}$ and let $h\colon \mathcal{X} \to \mathcal{Y}$ be a base classifier. We define the ε-smoothed classifier $g\colon \mathcal{X} \to \mathcal{Y}$ as $g(\boldsymbol{x}; \varepsilon) = \arg\max_{y \in \mathcal{Y}} q(y|\, \boldsymbol{x};\, \varepsilon)$ where q is given by the expectation with respect to the smoothing distribution ε, i.e.,

$$q(y|\, \boldsymbol{x};\, \varepsilon) := \mathbb{E}(p(y|\, \phi(\boldsymbol{x},\, \varepsilon))). \tag{7.2}$$

A key to certifying robustness against a specific transformation is the choice of transformation ϕ in the definition of the smoothed classifier (7.2). For example, if the goal is to certify the Gaussian blur transformation, a reasonable choice is to use the same transformation in the smoothed classifier. However, for other types of transformations this choice does not lead to the desired robustness certificate, and a different approach is required. In Sections 7.4 and 7.5, we derive approaches to overcome this challenge and certify robustness against a broader family of semantic transformations.

General Robustness Certification. Given an input $\boldsymbol{x} \in \mathcal{X}$ and a random variable ε taking values in $\mathcal{Z}$, suppose that the base classifier h predicts $\phi(\boldsymbol{x}, \varepsilon)$ to be of class y_A with probability at least p_A and the second most likely class with probability at most p_B (i.e., (7.4)). Our goal is to derive a robustness certificate for the ε-smoothed classifier g, i.e., we aim to find a set of perturbation parameters $\mathcal{S}_{\text{adv}}$ depending on p_A, p_B, and smoothing parameter ε such that, for all possible perturbation $\alpha \in \mathcal{S}_{\text{adv}}$, it is guaranteed that

$$g(\phi(\boldsymbol{x},\, \alpha);\, \varepsilon) = g(\boldsymbol{x};\, \varepsilon) \tag{7.3}$$

In other words, the prediction of the smoothed classifier can never be changed by applying the transformation ϕ with parameters α that are in the robust set $\mathcal{S}_{\text{adv}}$. The following theorem provides a generic robustness condition that we will subsequently leverage to obtain conditions on transformation parameters. In particular, this result addresses the first challenge **(C1)** for certifying semantic transformations since this result allows to certify

robustness beyond additive perturbations.

Theorem 7.1. Let $\varepsilon_0 \sim \mathcal{P}_0$ and $\varepsilon_1 \sim \mathcal{P}_1$ be $\mathcal{Z}$-valued random variables with probability density functions f_0 and f_1 with respect to a measure μ on $\mathcal{Z}$ and let $\phi\colon \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ be a semantic transformation. Suppose that $y_A = g(\boldsymbol{x};\, \varepsilon_0)$ and let $p_A,\, p_B \in [0,\, 1]$ be bounds to the class probabilities, i.e.,

$$q(y_A|\, \boldsymbol{x},\, \epsilon_0) \geqslant p_A > p_B \geqslant \max_{y \neq y_A} q(y|\, \boldsymbol{x},\, \epsilon_0). \tag{7.4}$$

For $t \geqslant 0$, let $\underline{S}_t,\, \overline{S}_t \subseteq \mathcal{Z}$ be the sets defined as $\underline{S}_t := \{f_1/f_0 < t\}$ and $\overline{S}_t := \{f_1/f_0 \leqslant t\}$ and define the function $\xi\colon [0,\, 1] \to [0,\, 1]$ by

$$\begin{aligned} \xi(p) &:= \sup\{\mathcal{P}_1(S)\colon\ \underline{S}_{\tau_p} \subseteq S \subseteq \overline{S}_{\tau_p}\} \\ \text{where} \quad \tau_p &:= \inf\{t \geqslant 0\colon\ \mathcal{P}_0(\overline{S}_t) \geqslant p\}. \end{aligned} \tag{7.5}$$

Then, if the condition

$$\xi(p_A) + \xi(1 - p_B) > 1 \tag{7.6}$$

is satisfied, then it is guaranteed that $g(\boldsymbol{x};\, \varepsilon_1) = g(\boldsymbol{x};\, \varepsilon_0)$.

A detailed proof for this statement is provided in Appendix E.1. At a high level, the condition (7.4) defines a family of classifiers based on class probabilities obtained from smoothing the input $\boldsymbol{x}$ with the distribution ε_0. Based on the Neyman Pearson Lemma from statistical hypothesis testing, shifting $\varepsilon_0 \to \varepsilon_1$ results in bounds to the class probabilities associated with smoothing x with ε_1. For class y_A, the lower bound is given by $\xi(p_A)$, while for any other class the upper bound is given by $1 - \xi(1 - p_B)$, leading to the the robustness condition $\xi(p_A) > 1 - \xi(1 - p_B)$. It is a more general version of what is proved by Cohen et al. [77], and its generality allows us to analyze a larger family of threat models. Notice that it is not immediately clear how one can obtain the robustness guarantee (7.3) and deriving such a guarantee from Theorem 7.1 is nontrivial. We will therefore explain in detail how this result can be instantiated to certify semantic transformations in Sections 7.4 and 7.5.

7.4 TSS-R: RESOLVEABLE TRANSFORMATIONS

In this section, we define resolvable transformations and then show how Theorem 7.1 is used to certify this class of semantic transformations. We then proceed to providing a robustness verification strategy for each specific transformation. In addition, we show how the generality of our framework allows us to reason about the best smoothing strategy for

a given transformation, which is beyond the reach of related randomized smoothing based approaches [118, 427].

Intuitively, we call a semantic transformation resolvable if we can separate transformation parameters from inputs with a function that acts on parameters and satisfies certain regularity conditions.

Definition 7.2 (Resolvable Transform)**.** A transformation $\phi\colon \mathcal{X}\times\mathcal{Z} \to \mathcal{X}$ is called resolvable if for any $\alpha \in \mathcal{Z}$ there exists a resolving function $\gamma_\alpha\colon \mathcal{Z} \to \mathcal{Z}$ that is injective, continuously differentiable, has non-vanishing Jacobian and for which

$$\phi(\phi(\boldsymbol{x}, \alpha), \beta) = \phi(\boldsymbol{x}, \gamma_\alpha(\beta)) \quad \boldsymbol{x} \in \mathcal{X}, \beta \in \mathcal{Z}. \tag{7.7}$$

Furthermore, we say that ϕ is additive, if $\gamma_\alpha(\beta) = \alpha + \beta$.

The following result provides a more intuitive view on Theorem 7.1, expressing the condition on probability distributions as a condition on the transformation parameters.

Corollary 7.1. Suppose that the transformation ϕ in Theorem 7.1 is resolvable with resolving function γ_α. Let $\alpha \in \mathcal{Z}$ and set $\varepsilon_1 := \gamma_\alpha(\varepsilon_0)$ in the definition of the function ξ. Then, if α satisfies condition (7.6), it is guaranteed that $g(\phi(\boldsymbol{x}, \alpha); \varepsilon_0) = g(\boldsymbol{x}; \varepsilon_0)$.

This corollary implies that for resolvable transformations, after we choose the smoothing distribution for the random variable ε_0, we can infer the distribution of $\varepsilon_1 = \gamma_\alpha(\varepsilon_0)$. Then, by plugging in ε_0 and ε_1 into Theorem 7.1, we can derive an explicit robustness condition from (7.6) such that for any α satisfying this condition, we can certify the robustness. In particular, for additive transformations we have $\epsilon_1 = \gamma_\alpha(\epsilon_0) = \alpha + \epsilon_0$. For common smoothing distributions ϵ_0 along with additive transformation, we derive robustness conditions in Appendix E.2.

In the next subsection, we focus on specific resolvable transformations. For certain transformations, this result can be applied directly. However, for some transformations, e.g., the composition of brightness and contrast, more careful analysis is required. We remark that this corollary also serves a stepping stone to certifying more complex transformations that are in general not resolvable, such as rotations as we will present in Section 7.5.

7.4.1 Certifying Specific Transformations

Here we build on our theoretical results from the previous section and provide approaches to certifying a range of different semantic transformations that are resolvable. We state all results here and provide proofs in appendices.

Gaussian Blur

This transformation is widely used in image processing to reduce noise and image detail. Mathematically, applying Gaussian blur amounts to convolving an image with a Gaussian function

$$G_\alpha(k) = \frac{1}{\sqrt{2\pi\alpha}} \exp\left(-k^2/(2\alpha)\right) \tag{7.8}$$

where $\alpha > 0$ is the *squared* kernel radius. For $\boldsymbol{x} \in \mathcal{X}$, we define Gaussian blur as the transformation $\phi_B \colon \mathcal{X} \times \mathbb{R}_{\geqslant 0} \to \mathcal{X}$ where

$$\phi_B(\boldsymbol{x}, \alpha) = \boldsymbol{x} * G_\alpha \tag{7.9}$$

and $*$ denotes the convolution operator. The following lemma shows that Gaussian blur is an *additive transform*. Thus, existing robustness conditions for additive transformations shown in Appendix E.2 are directly applicable.

Lemma 7.1. The Gaussian blur transformation is additive, i.e., for any $\alpha, \beta \geqslant 0$, we have $\phi_B(\phi_B(\boldsymbol{x}, \alpha), \beta) = \phi_B(\boldsymbol{x}, \alpha + \beta)$.

We notice that the Gaussian blur transformation uses only positive parameters. We therefore consider uniform noise on $[0, a]$ for $a > 0$, folded Gaussians and exponential distribution for smoothing.

Brightness and Contrast

This transformation first changes the brightness of an image by adding a constant value $b \in \mathbb{R}$ to every pixel, and then alters the contrast by multiplying each pixel with a positive factor e^k, for some $k \in \mathbb{R}$. We define the brightness and contrast transformation $\phi_{BC} \colon \mathcal{X} \times \mathbb{R}^2 \to \mathcal{X}$ as

$$(\boldsymbol{x}, \alpha) \mapsto \phi_{BC}(\boldsymbol{x}, \alpha) := e^k \cdot (\boldsymbol{x} + b \cdot \mathbb{1}_d), \quad \alpha = (k, b)^T \tag{7.10}$$

where $k, b \in \mathbb{R}$ are contrast and brightness parameters, respectively. We remark that ϕ_{BC} is resolvable; however, it is not additive and applying Corollary 7.1 directly using the resolving function γ_α leads to analytically intractable expressions. On the other hand, if the parameters k and b follow independent Gaussian distributions, we can circumvent this difficulty as follows. Given $\varepsilon_0 \sim \mathcal{N}(0, \operatorname{diag}(\sigma^2, \tau^2))$, we compute the bounds p_A and p_B to the class probabilities associated with the classifier $g(x; \varepsilon_0)$, i.e., smoothed with ε_0. In the next step, we identify a distribution ε_1 with the property that we can map any lower bound p of $q(y|\, x; \varepsilon_0)$

to a lower bound on $q(y|\, x;\, \varepsilon_1)$. Using ϵ_1 as a bridge, we then derive a robustness condition, which is based on Theorem 7.1, and obtain the guarantee that $g(\phi_{BC}(\boldsymbol{x},\, \alpha);\, \varepsilon_0) = g(\boldsymbol{x};\, \varepsilon_0)$ whenever the transformation parameters satisfy this condition. The next lemma shows that the distribution ε_1 with the desired property (lower bound to the classifier smoothed with ε_1) is given by a Gaussian with transformed covariance matrix.

Lemma 7.2. Let $\boldsymbol{x} \in \mathcal{X}$, $k \in \mathbb{R}$, $\varepsilon_0 \sim \mathcal{N}(0,\, \mathrm{diag}(\sigma^2,\, \tau^2))$ and $\varepsilon_1 \sim \mathcal{N}(0,\, \mathrm{diag}(\sigma^2,\, e^{-2k}\tau^2))$. Suppose that $q(y|\, \boldsymbol{x};\, \varepsilon_0) \geqslant p$ for some $p \in [0,\, 1]$ and $y \in \mathcal{Y}$. Let Φ be the cumulative density function of the standard Gaussian. Then

$$q(y|\, \boldsymbol{x};\, \varepsilon_1) \geqslant \begin{cases} 2\Phi\left(e^k \Phi^{-1}\left(\frac{1+p}{2}\right)\right) - 1 & k \leqslant 0 \\ 2\left(1 - \Phi\left(e^k \Phi^{-1}(1 - \frac{p}{2})\right)\right) & k > 0. \end{cases} \tag{7.11}$$

Now suppose that $g(\cdot;\, \varepsilon_0)$ makes the prediction y_A at $\boldsymbol{x}$ with probability at least p_A. Then, the preceding lemma tells us that the prediction confidence of $g(\cdot;\, \varepsilon_1)$ satisfies the lower bound (7.11) for the same class. Based on these confidence levels, we instantiate Theorem 7.1 with the random variables ε_1 and $\alpha + \varepsilon_1$ to get a robustness condition.

Lemma 7.3. Let ε_0 and ε_1 be as in Lemma 7.2 and suppose that

$$q(y_A|\, \boldsymbol{x};\, \varepsilon_1) \geqslant \tilde{p}_A > \tilde{p}_B \geqslant \max_{y \neq y_A} q(y|\, \boldsymbol{x};\, \varepsilon_1). \tag{7.12}$$

Then it is guaranteed that $y_A = g(\phi_{BC}(\boldsymbol{x},\, \alpha);\, \varepsilon_0)$ as long as $\alpha = (k,\, b)^T$ satisfies

$$\sqrt{\left(k/\sigma\right)^2 + \left(b/(e^{-k}\tau)\right)^2} < \frac{1}{2}\left(\Phi^{-1}\left(\tilde{p}_A\right) - \Phi^{-1}\left(\tilde{p}_B\right)\right). \tag{7.13}$$

In practice, we apply this lemma by replacing $\tilde{p}_A$ and $\tilde{p}_B$ in (7.13) with the bound computed from (7.11) based on the class probability bounds p_A and p_B associated with the classifier $g(x;\, \varepsilon_0)$.

In the actual computation, instead of certifying a single pair $(k,\, b)$, we usually certify the robustness against a set of transformation parameters

$$\mathcal{S}_{\mathrm{adv}} = \{(k,\, b)|\, k \in [-k_0,\, k_0], b \in [-b_0,\, b_0]\}, \tag{7.14}$$

which stands for any contrast change within e^{k_0} and brightness change within b_0. It is infeasible to check every $(k,\, b) \in \mathcal{S}_{\mathrm{adv}}$. To mitigate this, we relax the robustness condition

in Lemma 7.3 from

$$\sqrt{\left(k/\sigma\right)^2 + \left(b/(e^{-k}\tau)\right)^2} < \frac{1}{2}\left(\Phi^{-1}\left(\tilde{p}_A\right) - \Phi^{-1}\left(\tilde{p}_B\right)\right) \tag{7.15}$$

to

$$\sqrt{\left(k/\sigma\right)^2 + \left(b/(\min\{e^{-k},\, 1\}\tau)\right)^2} < \frac{1}{2}\left(\Phi^{-1}\left(\tilde{p}_A\right) - \Phi^{-1}\left(\tilde{p}_B\right)\right). \tag{7.16}$$

Thus, we only need to check the condition (7.16) for $(k_0,\, b_0)$ and $(-k_0,\, b_0)$ to certify the robustness for any $(k,\, b)$ in (7.14). This is because the LHS of (7.16) is monotonically increasing w.r.t. $|k|$ and $|b|$, and the RHS of (7.16) is equal to $\Phi^{-1}(\tilde{p}_A)$ that is monotonically decreasing w.r.t. $|k|$. Throughout the experiments, we use this strategy for certification of brightness and contrast.

Translation

Let $\bar{\phi}_T \colon \mathcal{X} \times \mathbb{Z}^2 \to \mathcal{X}$ be the transformation moving an image k_1 pixels to the right and k_2 pixels to the bottom with reflection padding. In order to handle continuous noise distributions, we define the translation transformation $\phi_T \colon \mathcal{X} \times \mathbb{R}^2 \to \mathcal{X}$ as $\phi_T(\boldsymbol{x},\, \alpha) = \bar{\phi}_T(\boldsymbol{x},\, [\alpha])$ where $[\cdot]$ denotes rounding to the nearest integer, applied element-wise. We note that ϕ_T is an *additive transform*, allowing us to directly apply Corollary 7.1 and derive robustness conditions. We note that if we use black padding instead of reflection padding, the transformation is not additive. However, since the number of possible translations is finite, another possibility is to use a simple brute-force approach that can handle black padding, which has already been studied extensively [268, 292].

Composition of Gaussian Blur, Brightness, Contrast, and Translation

Interestingly, the composition of all these four transformations is still resolvable. Thus, we are able to derive the explicit robustness condition for this composition based on Corollary 7.1, as shown in details in Section 7.6. Based on this robustness condition, we compute practically non-trivial robustness certificates as we will present in experiments in Section 7.7.

Robustness Certification Strategies

With these robustness conditions, for a given clean input $\boldsymbol{x}$, a transformation ϕ, and a set of parameters $\mathcal{S}_{\text{adv}}$, we certify the robustness of the smoothed classifier g with two steps: 1) estimate p_A and p_B (Equation (7.4)) with Monte-Carlo sampling and high-confidence

Table 7.1: Summary of the Robustness Certification Strategies for Resolvable Transformations.

Transformation	Step 1	Step 2
Gaussian Blur	Compute p_A and p_B with Monte-Carlo Sampling	Check via Corollary E.2 (in Appendix E.2)
Brightness		Check via Corollary E.1 (in Appendix E.2)
Translation		Check via Corollary E.1 (in Appendix E.2)
Brightness and Contrast		Compute $\tilde{p}_A$ via Lemma 7.2, then check via Lemma 7.3 (detail in Section 7.4.1)
Gaussian Blur, Brightness, Contrast and Translation		Compute $\tilde{p}_A$ via Corollary 7.3, then check via Lemma 7.6 (detail in Section 7.6)

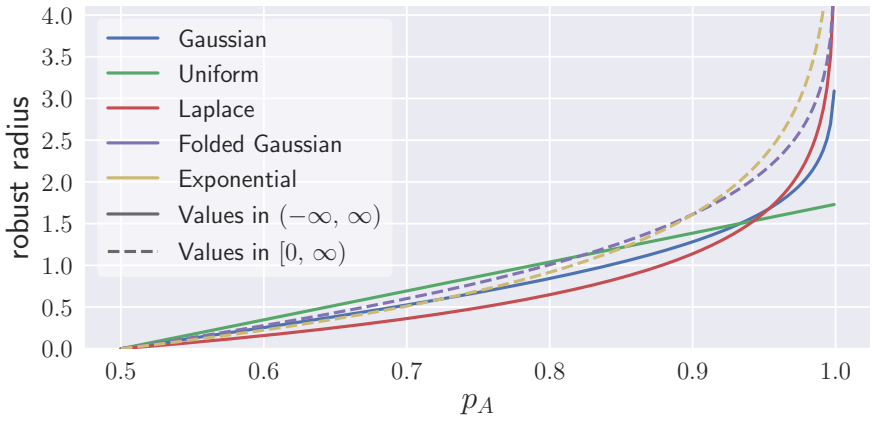


Figure 7.4: Robust radius comparison for different noise distributions, each with unit variance and dimension.

Table 7.2: Comparison of certification radii with $p_A + p_B = 1$, the variance and noise dimension are set to 1 for each distribution.

Distribution	Value Space	Robust Radius
Gaussian(0, 1)	$(-\infty, \infty)$	$\Phi^{-1}(p_A)$
Laplace(0, $1/\sqrt{2}$)	$(-\infty, \infty)$	$-\log(2-2p_A)/\sqrt{2}$
Uniform[$-\sqrt{3}$, $-\sqrt{3}$]	$(-\infty, \infty)$	$2\sqrt{3}\cdot(p_A - 1/2)$
Exponential(1)	$[0, \infty)$	$-\log(2-2p_A)$
FoldedGaussian(0, $\sqrt{\frac{\pi}{\pi-2}}$)	$[0, \infty)$	$\sqrt{\frac{\pi}{\pi-2}}\cdot\left(\Phi^{-1}\left(\frac{1+p_A}{2}\right) - \Phi^{-1}\left(\frac{3}{4}\right)\right)$

bound following Cohen et al. [77]; and 2) leverage the robustness conditions to obtain the certificate. A summary for each transformation including the used robustness conditions are shown in Table 7.1.

7.4.2 Properties of Smoothing Distributions

The robustness condition in Theorem 7.1 is generic and leaves a degree of freedom with regards to which smoothing distribution should be used. Previous work mainly provides results for cases in which this distribution is Gaussian [77, 437], while it is nontrivial to extend it to other distributions. Here, we aim to answer this question and provide results for a range of distributions, and discuss their differences. As we will see, *for different scenarios, different distributions behave differently and can certify different radii.* We instantiate Theorem 7.1 with an arbitrary transformation ϕ and with $\varepsilon_1 := \alpha + \varepsilon_0$ where ε_0 is the smoothing distribution and α is the transformation parameter. The robust radius is then derived by solving condition (7.6) for α.

Figure 7.4 illustrates robustness radii associated with different smoothing distributions, each scaled to have unit variance. The bounds are derived in Appendix E.2 and summarized in Table 7.2. We emphasize that the contribution of this work is not merely these results on different smoothing distributions but, more importantly, the *joint study between different smoothing mechanisms and different semantic transformations.* To compare the different radii for a fixed base classifier, we assume that *the smoothed classifier $g(\cdot; \varepsilon)$ always has the same confidence p_A for noise distributions with equal variance.* Finally, we provide the following conclusions and we will verify them empirically in Section 7.7.3.

1. *Exponential noise can provide larger robust radius.* We notice that smoothing with exponential noise generally allows for larger adversarial perturbations than other distributions. We also observe that, while all distributions behave similarly for low confidence levels, it is only non-uniform noise distributions that converge toward $+\infty$ when $p_A \to 1$ and exponential noise converges quickest.

2. *Additional knowledge can lead to larger robust radius.* When we have additional information on the transformation, e.g., all perturbations in Gaussian blur are positive, we can take advantage of this additional information and certify larger radii. For example, under this assumption, we can use folded Gaussian noise for smoothing instead of a standard Gaussian, resulting in a larger radius.

7.5 TSS-DR: Differentially Resolvable Transformations

As we have seen, our proposed function smoothing framework can directly deal with resolvable transformations. However, due to their use of interpolation, some important transformations do not fall into this category, including rotation, scaling, and their composition

with resolvable transformations. In this section, we show that they belong to the more general class termed *differentially resolvable transformations* and to address challenge **(C2)**, we propose a novel pipeline **TSS-DR** to provide rigorous robustness certification using our function smoothing framework as a central building block.

Common semantic transformations such as rotations and scaling do not fall into the category of resolvable transformations due to their use of interpolation. To see this issue, consider for example the rotation transformation denoted by ϕ_R. As shown in Figure 7.5(b), despite very similar, the image rotated by 30° is different from the image rotated separately by 15° and then again by 15°. The reason is the bilinear interpolation occurring during the rotation. Therefore, if the attacker inputs $\phi_R(\boldsymbol{x}, 15)$, the smoothed classifier in Section 7.4 outputs

$$g(\phi_R(\boldsymbol{x}, 15);\, \varepsilon) = \arg\max_{y \in \mathcal{Y}} \mathbb{E}\left(p(y|\, \phi_R(\phi_R(\boldsymbol{x}, 15), \varepsilon))\right), \tag{7.17}$$

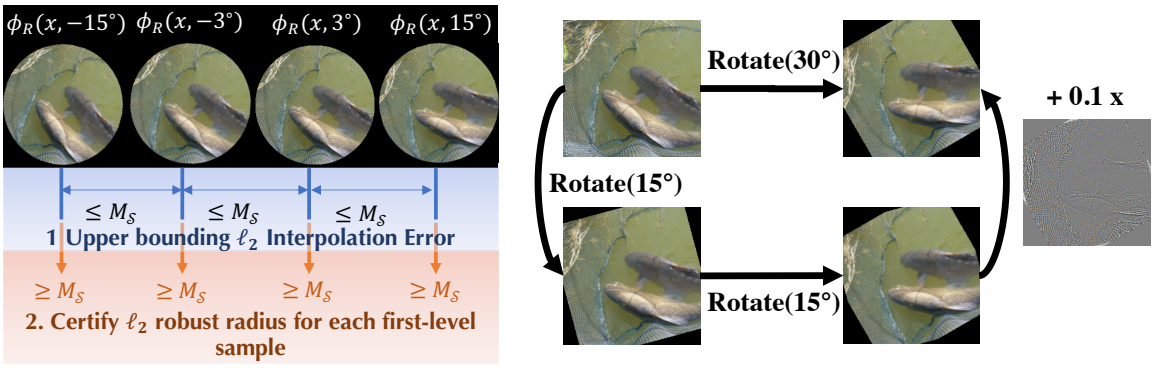
which is a weighted average over the predictions of the base classifier on the randomly perturbed set $\{\phi_R(\phi_R(\boldsymbol{x}, 15), \alpha) \colon \alpha \in \mathcal{Z}\}$. However, in order to use Corollary 7.1 and to reason about whether this prediction agrees with the prediction on the clean input (i.e., the average prediction on $\{\phi_R(\boldsymbol{x}, \alpha) \colon \alpha \in \mathcal{Z}\}$), we need ϕ_R to be resolvable. As it turns out, this is not the case for transformations that involve interpolation such as rotation and scaling.

To address these issues, we define a transformation ϕ to be *differentially resolvable*, if it can be written in terms of a resolvable transformation ψ and a parameter mapping δ.

Definition 7.3 (Differentially Resolvable Transform)**.** Let $\phi \colon \mathcal{X} \times \mathcal{Z}_\phi \to \mathcal{X}$ be a transformation with noise space $\mathcal{Z}_\phi$ and let $\psi \colon \mathcal{X} \times \mathcal{Z}_\psi \to \mathcal{X}$ be a resolvable transformation with noise space $\mathcal{Z}_\psi$. We say that ϕ can be resolved by ψ if for any $x \in \mathcal{X}$ there exists function $\delta_x \colon \mathcal{Z}_\phi \times \mathcal{Z}_\phi \to \mathcal{Z}_\psi$ such that for any $\alpha \in \mathcal{Z}_\phi$ and any $\beta \in \mathcal{Z}_\phi$,

$$\phi(\boldsymbol{x},\, \alpha) = \psi(\phi(\boldsymbol{x},\, \beta),\, \delta_x(\alpha,\, \beta)). \tag{7.18}$$

This definition leaves open a certain degree of freedom with regard to the choice of resolvable transformation ψ. For example, we can choose the resolvable transformation corresponding to additive noise $(\boldsymbol{x},\, \delta) \mapsto \psi(\boldsymbol{x},\, \delta) := \boldsymbol{x} + \delta$, which lets us write any transformation ϕ as $\phi(\boldsymbol{x},\, \alpha) = \phi(\boldsymbol{x},\, \beta) + (\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \beta)) = \psi(\phi(\boldsymbol{x},\, \beta),\, \delta)$ with $\delta = (\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \beta))$. In other words, $\phi(\boldsymbol{x}, \alpha)$ can be viewed as first being transformed to $\phi(\boldsymbol{x}, \beta)$ and then to $\phi(\boldsymbol{x}, \beta) + \delta$.



(a) (b)

Figure 7.5: (a) High-level illustration of our robustness certification pipeline **TSS-DR** for differentially resolvable transformations; (b) interpolation error.

7.5.1 Overview of TSS-DR

Here, we derive a general robustness certification strategy for differentially resolvable transformations. Suppose that our goal is to certify the robustness against a transformation ϕ that can be resolved by ψ and for transformation parameters from the set $\mathcal{S} \subseteq \mathcal{Z}_\phi$. To that end, we first sample a set of parameters $\{\alpha_i\}_{i=1}^N \subseteq \mathcal{S}$, and transform the input (with those sampled parameters) that yields $\{\phi(\boldsymbol{x}, \alpha_i)\}_{i=1}^N$. In the second step, we compute the class probabilities for each transformed input $\phi(\boldsymbol{x}, \alpha_i)$ with the classifier smoothed with the resolvable transformation ψ. Finally, the intuition is that, if every $\alpha \in \mathcal{S}$ is close enough to one of the sampled parameters, then the classifier is guaranteed to be robust against parameters from the set $\mathcal{S}$. In the next theorem, we show the existence of such a "proximity set" for general δ_x.

Theorem 7.2. Let $\phi\colon \mathcal{X} \times \mathcal{Z}_\phi \to \mathcal{X}$ be a transformation that is resolved by $\psi\colon \mathcal{X} \times \mathcal{Z}_\psi \to \mathcal{X}$. Let $\varepsilon \sim \mathcal{P}_\varepsilon$ be a $\mathcal{Z}_\psi$-valued random variable and suppose that the smoothed classifier $g : \mathcal{X} \to \mathcal{Y}$ given by $q(y|\,\boldsymbol{x}; \varepsilon) = \mathbb{E}(p(y|\,\psi(\boldsymbol{x}, \varepsilon)))$ predicts $g(\boldsymbol{x}; \varepsilon) = y_A = \arg\max_y q(y|\,\boldsymbol{x}; \varepsilon)$. Let $\mathcal{S} \subseteq \mathcal{Z}_\phi$ and $\{\alpha_i\}_{i=1}^N \subseteq \mathcal{S}$ be a set of transformation parameters such that for any i, the class probabilities satisfy

$$q(y_A|\,\phi(x, \alpha_i); \varepsilon) \geqslant p_A^{(i)} \geqslant p_B^{(i)} \geqslant \max_{y \neq y_A} q(y|\,\phi(x, \alpha_i); \varepsilon). \tag{7.19}$$

Then there exists a set $\Delta^* \subseteq \mathcal{Z}_\psi$ with the property that, if for any $\alpha \in \mathcal{S}$, $\exists \alpha_i$ with $\delta_{\boldsymbol{x}}(\alpha, \alpha_i) \in \Delta^*$, then it is guaranteed that

$$q(y_A|\,\phi(\boldsymbol{x}, \alpha); \varepsilon) > \max_{y \neq y_A} q(y|\,\phi(\boldsymbol{x}, \alpha); \varepsilon). \tag{7.20}$$

In the theorem, the smoothed classifier $g(\cdot;\,\varepsilon)$ is based on the resolvable transformation ψ that serves as a starting point to certify the target transformation ϕ. To certify ϕ over its parameter space $\mathcal{S}$, we input N transformed samples $\phi(\boldsymbol{x}, \alpha_i)$ to the smoothed classifier $g(\cdot;\,\varepsilon)$. Then, we get Δ^*, the certified robust parameter set for resolvable transformation ψ. This Δ^* means that for any $\phi(\boldsymbol{x}, \alpha_i)$, if we apply the transformation ψ with any parameter $\delta \in \Delta^*$, the resulting instance $\psi(\phi(\boldsymbol{x}, \alpha_i), \delta)$ is robust for $g(\cdot;\varepsilon)$. Since ϕ is resolvable by ψ, i.e., for any $\alpha \in \mathcal{S}$, there exists an α_i and $\delta \in \Delta^*$ such that $\phi(\boldsymbol{x}, \alpha) = \psi(\phi(\boldsymbol{x}, \alpha_i), \delta)$, we can assert that for any $\alpha \in \mathcal{S}$, the output of $g(\cdot;\varepsilon)$ on $\phi(\boldsymbol{x}, \alpha)$ is robust.

The key of using this theorem for a specific transformation is to choose the resolvable transformation ψ that can enable a tight calculation of Δ^* under a specific way of sampling $\{\alpha_i\}_{i=1}^N$. First, we observe that a large family of transformations including rotation and scaling can be resolved by the additive transformation $\psi\colon \mathcal{X} \times \mathcal{X} \to \mathcal{X}$ defined by $(\boldsymbol{x},\, \delta) \mapsto \boldsymbol{x} + \delta$. Indeed, any transformation whose pixel value changes are continuous (or with finite discontinuities) with respect to the parameter changes are differentially resolvable—they all can be resolved by the additive transformation. Choosing isotropic Gaussian noise $\varepsilon \sim \mathcal{N}(0,\, \sigma^2 \boldsymbol{I}_d)$ as smoothing noise then leads to the condition that the maximum ℓ_2-interpolation error between the interval $\mathcal{S} = [a,\, b]$ (which is to be certified) and the sampled parameters α_i must be bounded by a radius R. This result is shown in the next corollary, which is derived from Theorem 7.2.

Corollary 7.2. Let $\psi(\boldsymbol{x},\, \delta) = \boldsymbol{x} + \delta$ and let $\varepsilon \sim \mathcal{N}(0,\, \sigma^2 \boldsymbol{I}_d)$. Furthermore, let ϕ be a transformation with parameters in $\mathcal{Z}_\phi \subseteq \mathbb{R}^m$ and let $\mathcal{S} \subseteq \mathcal{Z}_\phi$ and $\{\alpha_i\}_{i=1}^N \subseteq \mathcal{S}$. Let $y_A \in \mathcal{Y}$ and suppose that for any i, the ε-smoothed classifier defined by $q(y|\, x; \varepsilon) := \mathbb{E}(p(y|\, \boldsymbol{x} + \varepsilon))$ has class probabilities that satisfy

$$q(y_A|\, \phi(\boldsymbol{x},\, \alpha_i);\, \varepsilon) \geqslant p_A^{(i)} \geqslant p_B^{(i)} \geqslant \max_{y \neq y_A} q(y|\, \phi(\boldsymbol{x},\, \alpha_i);\, \varepsilon). \tag{7.21}$$

Then it is guaranteed that $\forall \alpha \in \mathcal{S}\colon\ y_A = \arg\max_y q(y|\, \phi(\boldsymbol{x},\, \alpha); \varepsilon)$ if the maximum interpolation error

$$M_{\mathcal{S}} := \max_{\alpha \in \mathcal{S}} \min_{1 \leqslant i \leqslant N} \left\| \phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i) \right\|_2 \tag{7.22}$$

$$\text{satisfies} \quad M_{\mathcal{S}} < R := \frac{\sigma}{2} \min_{1 \leqslant i \leqslant N} \left(\Phi^{-1}\left(p_A^{(i)}\right) - \Phi^{-1}\left(p_B^{(i)}\right) \right). \tag{7.23}$$

In a nutshell, this corollary shows that if the smoothed classifier classifies all samples of transformed inputs $\{\phi(\boldsymbol{x}, \alpha_i)\}_{i=1}^N$ consistent with the original input and the smallest gap between confidence levels $p_A^{(i)}$ and $p_B^{(i)}$ is large enough, then it is guaranteed to make the same prediction on transformed inputs $\phi(\boldsymbol{x}, \alpha)$ for *any* $\alpha \in \mathcal{S}$.

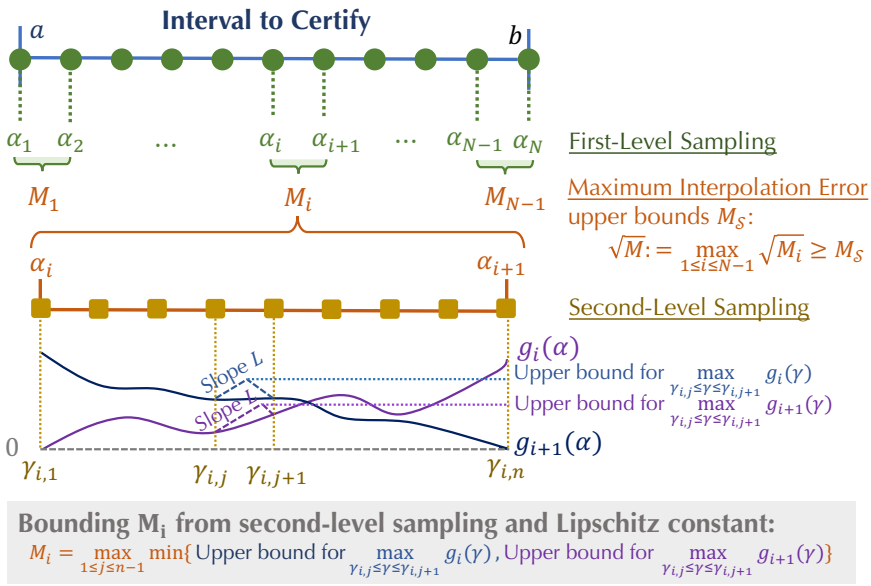


Figure 7.6: An overview of our interpolation error bounding technique based on stratified sampling and Lipschitz computation.

The main challenge now lies in computing a tight and scalable upper bound $M \geqslant M_{\mathcal{S}}$. Given this bound, a set of transformation parameters $\mathcal{S}$ can then be certified by computing R in (7.23) and checking that $R > M_{\mathcal{S}}$. With this methodology, we address challenge **(C2)** and provide means to certify transformations that incur interpolation errors. Figure 7.5(a) illustrates this methodology on a high level for the rotation transformation as an example. In the following, we present the general methodology that provides an upper bound of the interpolation error $M_{\mathcal{S}}$ and provide closed-form expressions for rotation and scaling. In Section 7.6, we further extend this methodology to certify transformation compositions such as rotation + brightness change + ℓ_2 perturbations.

We remark that dealing with the interpolation error has already been tried before [22, 118]. However, these approaches either leverage explicit linear or interval bound propagation – techniques that are either not scalable or not tight enough. Therefore, on large datasets such as ImageNet, they can provide only limited certification (e.g., against certain random attack instead of any attack).

7.5.2 Upper Bounding Interpolation Error

Here, we present the general methodology to compute a rigorous upper bound of the interpolation error introduced in Corollary 7.2. The methodology presented here is based on stratified sampling and is of a general nature; an explicit computation is shown for the

case of rotation and scaling toward the end of this subsection.

Let $\mathcal{S} = [a, b]$ be an interval of transformation parameters that we wish to certify and let $\{\alpha_i\}_{i=1}^N$ be parameters dividing $\mathcal{S}$ uniformly, i.e.,

$$\alpha_i = a + (b-a) \cdot \frac{i-1}{N-1}, \quad i = 1, \ldots, N. \tag{7.24}$$

The set of these parameters corresponds to the first-level samples in stratified sampling. With respect to these first-level samples, we define the functions $g_i \colon [a, b] \to \mathbb{R}_{\geqslant 0}$ as

$$\alpha \mapsto g_i(\alpha) := \|\phi(x, \alpha) - \phi(x, \alpha_i)\|_2^2 \tag{7.25}$$

corresponding to the squared ℓ_2 interpolation error between the image x transformed with α and α_i, respectively. For each first-level interval $[\alpha_i, \alpha_{i+1}]$ we look for an upper bound M_i such that

$$M_i \geqslant \max_{\alpha_i \leqslant \alpha \leqslant \alpha_{i+1}} \min\{g_i(\alpha), g_{i+1}(\alpha)\}. \tag{7.26}$$

It is easy to see that $\max_{1 \leqslant i \leqslant N-1} M_i \geqslant M_{\mathcal{S}}^2$ and hence setting

$$\sqrt{M} := \max_{1 \leqslant i \leqslant N-1} \sqrt{M_i} \tag{7.27}$$

is a valid upper bound to $M_{\mathcal{S}}$. The problem has thus reduced to computing the upper bounds M_i associated with each first-level interval $[\alpha_i, \alpha_{i+1}]$. To that end, we now continue with a second-level sampling within the interval $[\alpha_i, \alpha_{i+1}]$ for each i. Namely, let $\{\gamma_{i,j}\}_{j=1}^n$ be parameters dividing $[\alpha_i, \alpha_{i+1}]$ uniformly, i.e.,

$$\gamma_{i,j} = \alpha_i + (\alpha_{i+1} - \alpha_i) \cdot \frac{j-1}{n-1}, \quad j = 1, \ldots, n. \tag{7.28}$$

Now, suppose that L is a global Lipschitz constant for all functions $\{g_i\}_{i=1}^N$. By definition, for any $1 \leqslant i \leqslant N-1$, L satisfies

$$L \geqslant \max \left\{ \max_{c,d \in [\alpha_i, \alpha_{i+1}]} \left| \frac{g_i(c) - g_i(d)}{c-d} \right|, \max_{c,d \in [\alpha_i, \alpha_{i+1}]} \left| \frac{g_{i+1}(c) - g_{i+1}(d)}{c-d} \right| \right\}. \tag{7.29}$$

In the following, we will derive explicit expressions for L for rotation and scaling. Given the

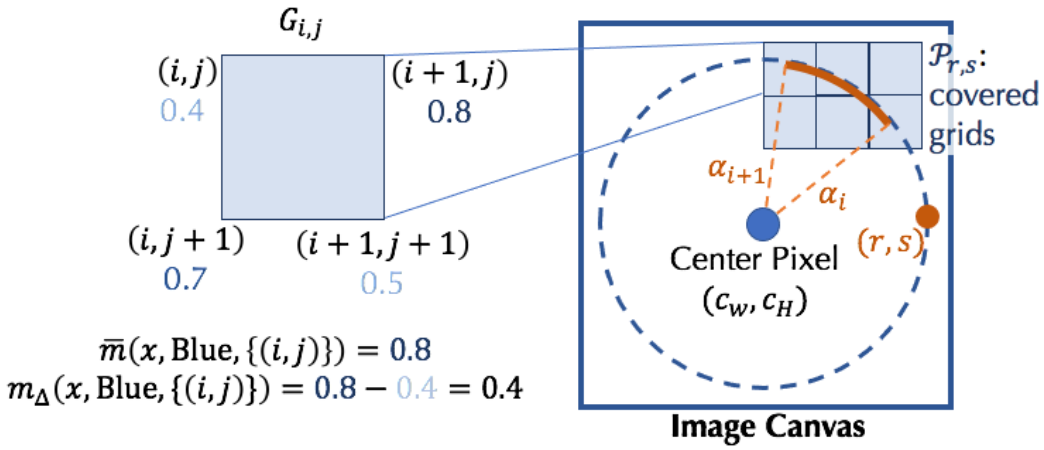


Figure 7.7: An illustration of Grid Pixel Generator $G_{i,j}$, Color Extractors $\bar{m}$ and m_Δ (take blue channel as example), and the set $\mathcal{P}_{r,s}$.

Lipschitz constant L, one can show the following closed-form expression for M_i:

$$M_i = \frac{1}{2} \max_{1 \leqslant j \leqslant n-1} \Big(\min \big\{ g_i(\gamma_{i,j}) + g_i(\gamma_{i,j+1}), \\ g_{i+1}(\gamma_{i,j}) + g_{i+1}(\gamma_{i,j+1}) \big\} \Big) + L \cdot \frac{b-a}{(N-1)(n-1)}. \tag{7.30}$$

An illustration of this bounding technique using stratified sampling is shown in Figure 7.6. We notice that, as the number N of first-level samples is increased, the interpolation error M_i becomes smaller by shrinking the sampling interval $[\alpha_i, \alpha_{i+1}]$; similarly, increasing the number of second-level samples n makes the upper bound of the interpolation error M_i tighter since the term $L(b-a)/\big((N-1)(n-1)\big)$ decreases. Furthermore, it is easy to see that as $N \to \infty$ or $n \to \infty$ we have $M \to M_{\mathcal{S}}^2$, i.e., our interpolation error estimation is *asymptotically tight*. Finally, this tendency also highlights an important advantage of our two-level sampling approach: without stratified sampling, it is required to sample $N \times n$ α_i's in order to achieve the same level of approximation accuracy. As a consequence, these $N \times n$ α_i's in turn require to evaluate the smoothed classifier in Corollary 7.2 $N \times n$ times, compared to just N times in our case.

It thus remains to find a way to efficiently compute the Lipschitz constant L for different transformations. In the following, we derive closed form expressions for rotation and scaling transformations.

7.5.3 Computing the Lipschitz Constant

Here, we derive a global Lipschitz constant L for the functions $\{g_i\}_{i=1}^N$ defined in (7.25), for rotation and scaling transformations. In the following, we define K-channel images of width W and height H to be tensors $\boldsymbol{x} \in \mathbb{R}^{K \times W \times H}$ and define the region of valid pixel indices as $\Omega := [0,\, W-1] \times [0,\, H-1] \cap \mathbb{N}^2$. Furthermore, for $(r,\, s) \in \Omega$, we define $d_{r,s}$ to be the ℓ_2-distance to the center of an image, i.e.,

$$d_{r,s} = \sqrt{\big(r - (W-1)/2\big)^2 + \big(s - (H-1)/2\big)^2}. \tag{7.31}$$

For ease of notation we make the following definitions that are illustrated in Figure 7.7.

Definition 7.4 (Grid Pixel Generator)**.** For pixels $(i,\, j) \in \Omega$, we define the grid pixel generator G_{ij} as

$$G_{ij} := \{(i,\, j),\, (i+1,\, j),\, (i,\, j+1),\, (i+1,\, j+1)\}. \tag{7.32}$$

Definition 7.5 (Max-Color Extractor)**.** We define the operator that extracts the channel-wise maximum pixel wise on a grid $S \subseteq \Omega$ as the map $\overline{m}\colon \mathbb{R}^{K \times W \times H} \times \{0, \ldots, K-1\} \times 2^{\Omega} \to \mathbb{R}$ with

$$\overline{m}(\boldsymbol{x},\, k,\, S) := \max_{(i,j) \in S} \big(\max_{(r,s) \in G_{ij}} x_{k,r,s} \big). \tag{7.33}$$

Definition 7.6 (Max-Color Difference Extractor)**.** We define the operator that extracts the channel-wise maximum change in color on a grid $S \subseteq \Omega$ as the map $m_\Delta\colon \mathbb{R}^{K \times W \times H} \times \{0, \ldots, K-1\} \times 2^{\Omega} \to \mathbb{R}$ with

$$m_\Delta(\boldsymbol{x},\, k,\, S) := \max_{(i,j) \in S} \big(\max_{(r,s) \in G_{ij}} x_{k,r,s} - \min_{(r,s) \in G_{ij}} x_{k,r,s} \big). \tag{7.34}$$

7.5.4 Rotation

The rotation transformation is defined as rotating an image by an angle α counter-clock wise, followed by bilinear interpolation I. Clearly, when rotating an image, some pixels may be padded that results in a sudden change of pixel colors. To mitigate this issue, we apply black padding to all pixels that are outside the largest centered circle in a given image (see Figure 7.5(a) for an illustration). We define the rotation transformation ϕ_R as the (raw) rotation $\tilde{\phi}_R$ followed by interpolation and the aforementioned preprocessing step P so that $\phi_R = P \circ I \circ \tilde{\phi}_R$ and refer the reader to Appendix E.5 for details. We remark that our certification is independent of different rotation padding mechanisms, since these

padded pixels are all refilled by black padding during preprocessing. The following lemma provides a closed form expression for L in (7.30) for rotation. A detailed proof is given in Appendix E.6.

Lemma 7.4. Let $x \in \mathbb{R}^{K\times W\times H}$ be a K-channel image and let $\phi_R = P \circ I \circ \tilde{\phi}_R$ be the rotation transformation. Then, a global Lipschitz constant L for the functions $\{g_i\}_{i=1}^N$ is given by

$$L_r = \max_{1\leqslant i\leqslant N-1} \sum_{k=0}^{K-1} \sum_{r,s\in V} 2d_{r,s} \cdot m_\Delta(x, k, \mathcal{P}_{r,s}^{(i)}) \cdot \overline{m}(x,\, k,\, \mathcal{P}_{r,s}^{(i)}) \tag{7.35}$$

where $V = \big\{(r,s) \in \mathbb{N}^2 |\, d_{r,s} < \frac{1}{2}(\min\{W,H\} - 1)\big\}$. The set $\mathcal{P}_{r,s}^{(i)}$ is given by all integer grid pixels that are covered by the trajectory of source pixels of (r,s) when rotating from angle α_i to α_{i+1}.

7.5.5 Scaling

The scaling transformation ϕ_S first stretches height and width of the input image by a factor $\alpha \in \mathbb{R}_+$ where values $\alpha < 1$ (> 1) correspond to shrinking (enlarging) an image. Then, bilinear interpolation is applied, followed by black padding to determine pixel values. We refer the reader to Appendix E.5 for a formal definition. Due to black padding, the functions g_i may contain discontinuities. To circumvent this issue, we enumerate all these discontinuities as $\mathcal{D}$. It can be shown that $\mathcal{D}$ contains at most $H + W$ elements. Hence, for large enough N, the interval $[\alpha_i,\, \alpha_{i+1}]$ contains at most one discontinuity. We thus modify the upper bounds M_i in (7.30) as

$$M_i := \begin{cases} \max\limits_{\alpha_i\leqslant\alpha\leqslant\alpha_{i+1}} \min\{g_i(\alpha),\, g_{i+1}(\alpha)\} & [\alpha_i,\, \alpha_{i+1}] \cap \mathcal{D} = \varnothing \\ \max\left\{ \max\limits_{\alpha_i\leqslant\alpha\leqslant t_i} g_{i+1}(\alpha),\, \max\limits_{t_i\leqslant\alpha\leqslant\alpha_{i+1}} g_i(\alpha) \right\} & [\alpha_i,\, \alpha_{i+1}] \cap \mathcal{D} = \{t_i\} \end{cases} \tag{7.36}$$

In either case, the quantity M_i can again be bounded by a Lipschitz constant. With this definition, the following lemma provides a closed form expression for the Lipschitz constant L in (7.30) for scaling. A detailed proof is given in Appendix E.6.

Lemma 7.5. Let $x \in \mathbb{R}^{K\times W\times H}$ be a K-channel image and let ϕ_S be the scaling transformation. Then, a global Lipschitz constant L for the functions $\{g_i\}_{i=1}^N$ is given by

$$L_\mathrm{s} = \max_{1\leqslant i\leqslant N-1} \sum_{k=0}^{K-1} \sum_{r,s\in\Omega\cap\mathbb{N}^2} \frac{\sqrt{2}d_{r,s}}{a^2} \cdot m_\Delta(x, k, \mathcal{P}_{r,s}^{(i)}) \cdot \overline{m}(x,\, k,\, \mathcal{P}_{r,s}^{(i)}) \tag{7.37}$$

where $\Omega = [0,\, W-1] \times [0,\, H-1]$ and a is the lower boundary value in $\mathcal{S} = [a,\, b]$. The set $\mathcal{P}_{r,s}^{(i)}$ is given by all integer grid pixels that are covered by the trajectory of source pixels of (r, s) when scaling with factors from α_{i+1} to α_i.

7.5.6 Computational Complexity

We provide pseudo-code for computing bound M in Appendix E.7. The algorithm is composed of two main parts, namely the computation of the Lipschitz constant L, and the computation of the interpolation error bound M based on L. The former is of computational complexity $\mathcal{O}(N \cdot KWH)$, and the latter is of $\mathcal{O}(NR \cdot KWH)$, for both scaling and rotation. We note that $\mathcal{P}_{r,s}$ contains only a constant number of pixels since each interval $[\alpha_i, \alpha_{i+1}]$ is small. Thus, the bulk of costs come from the transformation operation. We improve the speed by implementing a fast and fully-parallelized `C` kernel for rotation and scaling of images. As a result, on CIFAR-10, the algorithm takes less than 2 s on average with 10 processes for rotation with $N = 556$ and $n = 200$ and the time for scaling is faster. We refer readers to Section 7.7 for detailed experimental evaluation. Also, we remark that the algorithm is model-independent. Thus, we can precompute M for test set and reuse for any models that need a certification.

7.5.7 Acceleration with Progressive Sampling

With the methodology mentioned above, for differentially resolvable transformations such as rotation and scaling, computing robustness certification follows two steps: (1) computing the interpolation error bound M; (2) generate transformed samples $\{\phi(x, \alpha_i)\}_{i=1}^{N}$, compute $p_A^{(i)}$ and $p_B^{(i)}$ for each sample, and check whether $M_{\mathcal{S}} < R$ holds for each sample according to Corollary 7.2.

In step (2) above, we need to estimate $p_A^{(i)}$ and $p_B^{(i)}$ for each sample $\phi(x, \alpha_i)$ to check whether $M_{\mathcal{S}} < R$. In the brute-force approach, to obtain a high-confidence bound on $p_A^{(i)}$ and $p_B^{(i)}$, we typically sample $n_s = 10,000$ or more [77] then apply the binomial statistical test. In total, we thus need to sample the classifier's prediction $N \times n_s$ times, which is costly.

To accelerate the computation, we design a *progressive sampling strategy* from the following two insights: (1) we only need to check whether $R > M_{\mathcal{S}}$, but are not required to compute R precisely; (2) for any sample $\phi(x, \alpha_i)$ if the check fails, the model is not certifiably robust and there is no need to proceed. Based on (1), for the current $\phi(x,\, \alpha_i)$, we sample n_s samples in batches and maintain high-confidence lower bound of R based on existing estimation. Once the lower bound exceeds $M_{\mathcal{S}}$ we proceed to the next $\phi(x,\, \alpha_{i+1})$. Based on (2), we

terminate early if the check $R > M_{\mathcal{S}}$ for the current $\phi(x, \alpha_i)$ fails. More details are provided in Appendix E.7.

7.6 EXTENSION TO MORE TRANSFORMATIONS

For other transformations that involve interpolation, we can similarly compute the interpolation error bound using intermediate results in our above lemmas. For transformation compositions, we extend our certification pipeline for the composition of (1) Gaussian blur, brightness change, contrast change, and translation happening together, (2) rotation/scaling with brightness, and (3) rotation/scaling with brightness and ℓ_p-bounded additive perturbations. These compositions simulate an attacker who does not precisely perform the specified transformation. At the end of this section, we discuss how to analyze possible new transformations and then extend TSS to provide certification.

Gaussian Blur, Brightness, Contrast, and Translation

The certification generally follows the same procedure as in certifying brightness and contrast. In the following, we first provide a formal definition of this transformation composition. Specifically, the transformation ϕ_{BTBC} is defined as:

$$\phi_{BTBC}(\boldsymbol{x}, \alpha) := \phi_B(\phi_T(\phi_{BC}(\boldsymbol{x}, \alpha_k, \alpha_b), \alpha_{Tx}, \alpha_{Ty}), \alpha_B), \tag{7.38}$$

where ϕ_B, ϕ_T and ϕ_{BC} are Gaussian blur, translation, and brightness and contrast transformations respectively as defined before; $\alpha := (\alpha_k, \alpha_b, \alpha_{Tx}, \alpha_{Ty}, \alpha_B)^T \in \mathbb{R}^4 \times \mathbb{R}_{\geqslant 0}$ is the transformation parameter.

Our certification relies on the following corollary (extended from Lemma 7.2) and lemma, which are proved in Appendix E.3.

Corollary 7.3. Let $x \in \mathcal{X}$, $k \in \mathbb{R}$ and let $\epsilon_0 := (\epsilon_0^a, \epsilon_0^b)^T$ be a random variable defined as

$$\epsilon_0^a \sim \mathcal{N}(0, \mathrm{diag}(\sigma_k^2, \sigma_b^2, \sigma_T^2, \sigma_T^2)) \,\mathrm{and}\, \epsilon_0^b \sim \mathrm{Exp}(\lambda_B). \tag{7.39}$$

Similarly, let $\epsilon_1 := (\epsilon_1^a, \epsilon_1^b)$ be a random variable with

$$\epsilon_1^a \sim \mathcal{N}(0, \mathrm{diag}(\sigma_k^2, e^{-2k}\sigma_b^2, \sigma_T^2, \sigma_T^2)) \,\mathrm{and}\, \epsilon_1^b \sim \mathrm{Exp}(\lambda_B). \tag{7.40}$$

For either random variable (denoted as ϵ), recall that $q(y|\boldsymbol{x}; \epsilon) := \mathbb{E}(p(y|\phi_{BTBC}(\boldsymbol{x}, \epsilon)))$.

Suppose that $q(y|\boldsymbol{x};\, \epsilon_0) \geqslant p$ for some $p \in [0,\, 1]$ and $y \in \mathcal{Y}$. Then $q(y|\boldsymbol{x};\, \epsilon_1)$ satisfies Eq. (11).

Lemma 7.6. Let ϵ_0 and ϵ_1 be as in Corollary 7.3 and suppose that

$$q(y_A|\boldsymbol{x};\, \varepsilon_1) \geqslant \tilde{p}_A > \tilde{p}_B \geqslant \max_{y \neq y_A} q(y|\boldsymbol{x};\, \varepsilon_1). \tag{7.41}$$

Then it is guaranteed that $y_A = g(\phi_{BTBC}(x,\, \alpha); \varepsilon_0)$ as long as $p'_A > p'_B$, where

$$p'_A = \begin{cases} 0, & \text{if } \tilde{p}_A \leqslant 1 - \exp(-\lambda_B \alpha_B), \\ \Phi\left(\Phi^{-1}\left(1 - (1 - \tilde{p}_A)\exp(\lambda_B \alpha_B)\right) - \sqrt{\alpha_k^2/\sigma_k^2 + \alpha_b^2/(e^{-2\alpha_k}\sigma_b^2) + (\alpha_{Tx}^2 + \alpha_{Ty}^2)/\sigma_T^2}\right) & \text{otherwise} \end{cases} \tag{7.42}$$

and

$$p'_B = \begin{cases} 1, & \text{if } \tilde{p}_B \geqslant \exp(-\lambda_B \alpha_B), \\ 1 - \Phi\left(\Phi^{-1}\left(1 - \tilde{p}_B \exp(\lambda_B \alpha_B)\right) - \sqrt{\alpha_k^2/\sigma_k^2 + \alpha_b^2/(e^{-2\alpha_k}\sigma_b^2) + (\alpha_{Tx}^2 + \alpha_{Ty}^2)/\sigma_T^2}\right). & \text{otherwise} \end{cases} \tag{7.43}$$

The ϵ_0 specified by (7.39) is the smoothing distribution. Similar as in brightness and contrast certification, we first obtain p_A, a lower bound of $q(y_A|\, \boldsymbol{x},\, \varepsilon_0)$ by Monte-Carlo sampling. For a given transformation parameter $\alpha := (\alpha_k,\, \alpha_b,\, \alpha_{Tx},\, \alpha_{Ty},\, \alpha_B)^T$, we then trigger Corollary 7.3 to get $\tilde{p}_A$, a lower bound of $q(y_A|\, \boldsymbol{x},\, \varepsilon_1)$ and set $\tilde{p}_B = 1 - \tilde{p}_A$. Finally, we use the explicit condition in Lemma 7.6 to obtain the certification. Indeed, with $\tilde{p}_B = 1 - \tilde{p}_A$, Lemma 7.6 can be simplified to the following corollary.

Corollary 7.4. Let ϵ_0 and ϵ_1 be as in Corollary 7.3 and suppose that

$$q(y_A|\boldsymbol{x};\, \varepsilon_1) \geqslant \tilde{p}_A. \tag{7.44}$$

Then it is guaranteed that $y_A = g(\phi_{BTBC}(x,\, \alpha); \varepsilon_0)$ as long as

$$\tilde{p}_A > 1 - \exp(-\lambda_B \alpha_B)\left(1 - \Phi\left(\sqrt{\frac{\alpha_k^2}{\sigma_k^2} + \frac{\alpha_b^2}{e^{-2\alpha_k}\sigma_b^2} + \frac{\alpha_{Tx}^2 + \alpha_{Ty}^2}{\sigma_T^2}}\right)\right). \tag{7.45}$$

To certify against a set of transformation parameters

$$\begin{aligned}\mathcal{S}_{\text{adv}} = \{(\alpha_k,\, \alpha_b,\, \alpha_{Tx},\, \alpha_{Ty},\, \alpha_B)^T | \alpha_k \in [-k_0,\, k_0], \alpha_b \in [-b_0,\, b_0], \\ \|(\alpha_{Tx}, \alpha_{Ty})\|_2 \leqslant T, \alpha_B \leqslant B_0\},\end{aligned} \tag{7.46}$$

we relax the robust condition in (7.45) to

$$\tilde{p}_A > 1 - \exp(-\lambda_B \alpha_B)\left(1 - \Phi\left(\sqrt{\frac{\alpha_k^2}{\sigma_k^2} + \frac{\alpha_b^2}{\min\{e^{-2\alpha_k},\, 1\}\sigma_b^2} + \frac{\alpha_{Tx}^2 + \alpha_{Ty}^2}{\sigma_T^2}}\right)\right). \tag{7.47}$$

The LHS of Equation (7.47) is monotonically decreasing w.r.t. $|\alpha_k|$ and the RHS is monotonically increasing w.r.t. $|\alpha_k|, |\alpha_b|, \|(\alpha_{Tx}, \alpha_{Ty}\|_2$, and $|\alpha_B|$, and the RHS is symmetric w.r.t. α_b and $\|(\alpha_{Tx}, \alpha_{Ty})\|_2$. As a result, we only need to check the condition for $(-k_0,\, b_0,\, T,\, 0,\, B_0)$ and $(k_0,\, b_0,\, T,\, 0,\, B_0)$ to certify the entire set $\mathcal{S}_{\text{adv}}$. Throughout the experiments, we use this strategy for certification.

Scaling/Rotation and Brightness

To certify the composition of scaling and brightness or rotation and brightness, we follow the same methodology as certifying scaling or rotation alone and reuse the computed interpolatation error $M_{\mathcal{S}}$. We only make the following two changes: (1) alter the smoothing distribution from additive Gaussian noise $\psi(\boldsymbol{x}, \delta) = \boldsymbol{x} + \delta$ where $\delta \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ to additive Gaussian noise and Gaussian brightness change $\psi(\boldsymbol{x}, \delta, \delta_b) = \boldsymbol{x} + \delta + b \cdot \boldsymbol{I}_d$ where $\delta \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d), b \sim \mathcal{N}(0, \sigma_b^2)$; (2) change the robustness condition from $R > M_{\mathcal{S}}$ in Corollary 7.2 to $R > \sqrt{M_{\mathcal{S}}^2 + (\sigma^2/\sigma_b^2) b_0^2}$. We formalize this robustness condition in the following corollary, and the proof is entailed in Appendix E.4.

Corollary 7.5. Let $\psi_B(\boldsymbol{x},\, \delta,\, b) = \boldsymbol{x} + \delta + b \cdot \mathbb{1}_d$ and let $\varepsilon \sim \mathcal{N}(0,\, \sigma^2 \boldsymbol{I}_d)$, $\varepsilon_b \sim \mathcal{N}(0,\, \sigma_b^2)$. Furthermore, let ϕ be a transformation with parameters in $\mathcal{Z}_\phi \subseteq \mathbb{R}^m$ and let $\mathcal{S} \subseteq \mathcal{Z}_\phi$ and $\{\alpha_i\}_{i=1}^N \subseteq \mathcal{S}$. Let $y_A \in \mathcal{Y}$ and suppose that for any i, the $(\varepsilon, \varepsilon_b)$-smoothed classifier $q(y|\, x;\, \varepsilon, \varepsilon_b) := \mathbb{E}(p(y|\, \psi_B(x,\, \varepsilon,\, \varepsilon_b))$ satisfies

$$q(y_A|\, \boldsymbol{x};\, \varepsilon, \varepsilon_b) \geqslant p_A^{(i)} > p_B^{(i)} \geqslant \max_{y \neq y_A} q(y|\, \boldsymbol{x};\, \varepsilon, \varepsilon_b). \tag{7.48}$$

for each i. Let

$$R := \frac{\sigma}{2} \min_{1 \leqslant i \leqslant N} \left(\Phi^{-1}\left(p_A^{(i)}\right) - \Phi^{-1}\left(p_B^{(i)}\right)\right) \tag{7.49}$$

Then, $\forall \alpha \in \mathcal{S}$ and $\forall b \in [-b_0, b_0]$ it is guaranteed that $y_A = \arg\max_y q(y|\, \phi(\boldsymbol{x}, \alpha) + b \cdot \mathbb{1}_d;\, \varepsilon, \varepsilon_b)$

as long as

$$R > \sqrt{M_{\mathcal{S}}^2 + \frac{\sigma^2}{\sigma_b^2} b_0^2}, \tag{7.50}$$

where $M_{\mathcal{S}}$ is defined as in Corollary 7.2.

Scaling/Rotation, Brightness, and ℓ_2 Perturbations

We use the same smoothing distribution as above, and the following corollary directly allows us to certify the robustness against the composition of scaling/rotation, brightness, and an additional ℓ_2-bounded perturbations—we only need to change the robustness condition from $R > \sqrt{M_{\mathcal{S}}^2 + (\sigma^2/\sigma_b^2) b_0^2}$ to $R > \sqrt{(M_{\mathcal{S}} + r)^2 + (\sigma^2/\sigma_b^2) b_0^2}$. The proof is given in Appendix E.4.

Corollary 7.6. Under the same setting as in Corollary 7.5, for $\forall \alpha \in \mathcal{S}$, $\forall b \in [-b_0, b_0]$ and $\forall \delta \in \mathbb{R}^d$ such that $\|\delta\|_2 \leqslant r$, it is guaranteed that $y_A = \arg\max_k q(y|\, \phi(\boldsymbol{x}, \alpha) + b \cdot \mathbb{1}_d + \delta;\, \varepsilon, \varepsilon_d)$ as long as

$$R > \sqrt{(M_{\mathcal{S}} + r)^2 + \frac{\sigma^2}{\sigma_b^2} b_0^2}, \tag{7.51}$$

where $M_{\mathcal{S}}$ is defined as in Corollary 7.2.

On More Transformations and Compositions

Our TSS is not limited to specific transformations. For a new transformation, we first identify the parameter space $\mathcal{Z}$, where the identified parameter should completely and deterministically decide the output after transformation for any given input. Then, we use Definition 7.2 to check whether the transformation is resolvable. If so, we can write down the function γ_α. Next, we choose a smoothing distribution, i.e., the distribution of the random variable ε_0, and identify the distribution of $\varepsilon_1 = \gamma_\alpha(\varepsilon_0)$. Finally, we use Theorem 7.1 to derive the robustness certificates and follow the two-step template (Section 7.4.1) to compute the robustness certificate.

If the transformation is not resolvable, we identify a dimension in $\mathcal{Z}$ for which the transformation is *resolvable*. For example, the composition of rotation and brightness has a rotation and a brightness axis, where the brightness axis is itself resolvable. As a result, we can write the parameter space as Cartesian product of non-resolvable subspace and resolvable subspace: $\mathcal{Z} := \mathcal{Z}_{\text{no-resolve}} \times \mathcal{Z}_{\text{resolve}}$. We perform smoothing on the resolvable subspace and sample enough points in the non-resolvable subspace. Next, we bound the interpolation error

between sampled points and arbitrary points in the non-resolvable subspace, using either ℓ_p difference as we did for rotation and scaling or other regimes. Specifically, our Lemma E.4 shown in Appendix E.6 is a useful tool to bound ℓ_p difference caused by interpolation error. Finally, we instantiate Theorem 7.2 to compute the robustness certificate.

Theoretically, we can certify against the composition of all the discussed transformations: Gaussian blur, brightness, contrast, translation, rotation, and scaling. However, as justified in [146, Figure 3], the composition of more than two transformations leads to unrealistic images that are even hard to distinguish by humans. Moreover, if the composition contains too many transformations, the parameter space would be no longer low dimensional. Therefore, there would be much more axes that are differentially resolvable (instead of resolvable). As a consequence, much more samples are required to obtain a small bound on interpolation error (which is necessary for a nontrivial robustness certification). Therefore, we mainly evaluate on single transformation or composition of two transformations to simulate a practical attack.

7.7 EXPERIMENTS

We validate our framework TSS by certifying robustness over semantic transformations experimentally. We compare with state of the art for each transformation, highlight our main results, and present some interesting findings and ablation studies.

7.7.1 Experimental Setup

Dataset

Our experiments are conducted on three classical image classification datasets: MNIST, CIFAR-10, and ImageNet. For all images, the pixel color is normalized to $[0, 1]$. We follow common practice to resize and center cropping the ImageNet images to 224×224 size [77, 165, 300, 427]. To our best knowledge, we are the *first* to provide rigorous certifiable robustness against semantic transformations on the large-scale *standard* ImageNet dataset.

Model

The undefended model is very vulnerable even under simple random semantic attacks. Therefore, we apply existing data augmentation training [77] combined with consistency regularization [165] to train the base classifiers. We then use the introduced smoothing strategies, to obtain the models for robustness certification. On MNIST and CIFAR-10,

the models are trained from scratch while on ImageNet, we either finetune undefended models in `torchvision` library or finetune from state-of-the-art certifiably robust models against ℓ_2 perturbations [317]. Details are available in Appendix E.8.1. We remark that our framework focuses on robustness certification and did not fully explore the training methods for improving the certified robustness or tune the hyperparameters.

Implementation and Hardware

We implement our framework TSS based on `PyTorch`. We improve the running efficiency by tensor parallelism and embedding `C` modules. Details are available in Appendix E.8.2. All experiments were run on 24-core Intel Xeon Platinum 8259CL CPU and one Tesla T4 GPU with 15 GB RAM.

Evaluation Metric

On each dataset, we uniformly pick 500 samples from the test set and evaluate all results on this *test subset* following Cohen et al. [77]. In line with related work [77, 165, 317, 427] and Part II, we report the **certified robust accuracy** that is defined as the fraction of samples (within the test subset) that are both *certified robust* and *classified correctly*, and set the certification confidence level to $p = 0.1\%$. We use $n_s = 10^5$ samples to obtain a confidence lower bound $\underline{p_A}$ for resolvable transformations, and $n_s = 10^4$ samples to obtain each $\underline{p_A}^{(i)}$ for differentially resolvable transformations. Due to Progressive Sampling (Algorithm E.2), the actual samples used for differentially resolvable transformations are usually far fewer than n_s. In addition, we report the **benign accuracy** in Appendix E.8.5 defined as the fraction of *correctly classified* samples when no attack is present, and the **empirical robust accuracy**, defined as the fraction of samples in the test subset that are classified correctly under either a simple random attack (following [22, 118]) or two adaptive attacks (namely Random+ Attack and PGD Attack). We introduce all these attacks in Appendix E.8.3 and provide a detailed comparison in Appendix E.8.7. Note that the empirical robust accuracy under any attacks is lower bounded by the certified accuracy.

Notations for Robust Radii

In the tables, we use these notations: α for squared kernel radius for Gaussian blur; $\sqrt{\Delta x^2 + \Delta y^2}$ for translation distance; b and c for brightness shift and contrast change respectively as in $\boldsymbol{x} \mapsto (1 + c)\boldsymbol{x} + b \cdot \mathbb{1}_d$; r for rotation angle; s for size scaling ratio; and $\|\delta\|_2$ for ℓ_2 norm of additional perturbations.

Table 7.3: Comparison of certified robust accuracy achieved by our framework TSS and other known baselines and empirical robust accuracy achieved by TSS and vanilla models under random and adaptive attacks. "-" denotes the settings where the baselines cannot support. The parentheses show the weaker baseline settings. For certified robust accuracy, the existing state of the art is **bolded**. For empirical robust accuracy, the higher accuracy under each setting are **bolded**.

Transformation	Type	Dataset	Attack Radius	Certified Robust Accuracy						Empirical Robust Accuracy			
				TSS	DeepG [22]	Interval [343]	VeriVis [292]	Semantify-NN [268]	DistSPT [118]	Random Attack		Adaptive Attacks	
										TSS	Vanilla	TSS	Vanilla
Gaussian Blur	Resolvable	MNIST	Squared Radius $\alpha \leqslant 36$	**90.6**%	-	-	-	-	-	**91.4%**	12.2%	**91.2%**	12.2%
		CIFAR-10	Squared Radius $\alpha \leqslant 16$	**63.6**%	-	-	-	-	-	**65.8%**	3.4%	**65.8%**	3.4%
		ImageNet	Squared Radius $\alpha \leqslant 36$	**51.6**%	-	-	-	-	-	**52.8%**	8.4%	**52.6%**	8.2%
Translation (Reflection Pad.)	Resolvable, Discrete	MNIST	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 8$	**99.6**%	-	-	98.8%	98.8%	-	**99.6%**	0.0%	**99.6%**	0.0%
		CIFAR-10	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 20$	**80.8**%	-	-	65.0%	65.0%	-	**86.2%**	4.4%	**86.0%**	4.2%
		ImageNet	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 100$	**50.0**%	-	-	43.2%	43.2%	-	**69.2%**	46.6%	**69.2%**	46.2%
Brightness	Resolvable	MNIST	$b \pm 50\%$	**98.2**%	-	-	-	-	-	**98.2%**	96.6%	**98.2%**	96.6%
		CIFAR-10	$b \pm 40\%$	**87.0**%	-	-	-	-	-	**87.2%**	44.4%	**87.4%**	42.6%
		ImageNet	$b \pm 40\%$	**70.0**%	-	-	-	-	-	**70.4%**	19.6%	**70.4%**	18.4%
Contrast and Brightness	Resolvable, Composition	MNIST	$c \pm 50\%, b \pm 50\%$	**97.6**%	$\leqslant 0.4\%$ $(c, b \pm 30\%)$	0.0% $(c, b \pm 30\%)$	-	$\leqslant 74\%$ $(c \pm 5\%, b \pm 50\%)$	-	**98.0%**	94.6%	**98.0%**	93.2%
		CIFAR-10	$c \pm 40\%, b \pm 40\%$	**82.4**%	0.0% $(c, b \pm 30\%)$	0.0% $(c, b \pm 30\%)$	-	-	-	**86.0%**	21.0%	**85.8%**	9.6%
		ImageNet	$c \pm 40\%, b \pm 40\%$	**61.4**%	-	-	-	-	-	**68.4%**	1.2%	**68.4%**	0.0%
Gaussian Blur, Translation, Brightness, and Contrast	Resolvable, Composition	MNIST	$\alpha \leqslant 1, \sqrt{\Delta x^2 + \Delta y^2} \leqslant 5, c, b \pm 10\%$	**90.2%**	-	-	-	-	-	**97.2%**	0.4%	**97.0%**	0.4%
		CIFAR-10	$\alpha \leqslant 1, \sqrt{\Delta x^2 + \Delta y^2} \leqslant 5, c, b \pm 10\%$	**58.2%**	-	-	-	-	-	**67.6%**	9.6%	**67.8%**	5.6%
		ImageNet	$\alpha \leqslant 10, \sqrt{\Delta x^2 + \Delta y^2} \leqslant 10, c, b \pm 20\%$	**32.8%**	-	-	-	-	-	**48.8%**	9.4%	**47.4%**	4.0%
Rotation	Differentially Resolvable	MNIST	$r \pm 50°$	**97.4**%	$\leqslant 85.8\%$ $(r \pm 30°)$	$\leqslant 6.0\%$ $(r \pm 30°)$	-	$\leqslant 92.48\%$	82%	**98.4%**	12.2%	**98.2%**	11.0%
		CIFAR-10	$r \pm 10°$	**70.6**%	62.5%	20.2%	-	-	37%	**76.6%**	65.6%	**76.4%**	65.4%
			$r \pm 30°$	**63.6**%	10.6%	0.0%	-	$\leqslant 49.37\%$	22%	**69.2%**	21.6%	**69.4%**	21.4%
		ImageNet	$r \pm 30°$	**30.4**%	-	-	-	-	16% (rand. attack)	37.8%	**40.0%**	**37.8%**	37.0%
Scaling	Differentially Resolvable	MNIST	$s \pm 30\%$	**97.2**%	85.0%	16.4%	-	-	-	**99.2%**	90.2%	**99.2%**	89.2%
		CIFAR-10	$s \pm 30\%$	**58.8**%	0.0%	0.0%	-	-	-	**67.2%**	51.6%	**67.0%**	51.2%
		ImageNet	$s \pm 30\%$	**26.4**%	-	-	-	-	-	37.4%	**50.0%**	36.4%	**49.8%**
Rotation and Brightness	Differentially Resolvable, Composition	MNIST	$r \pm 50°, b \pm 20\%$	**97.0**%	-	-	-	-	-	**98.2%**	11.0%	**98.0%**	10.4%
		CIFAR-10	$r \pm 10°, b \pm 10\%$	**70.2**%	-	-	-	-	-	**76.6%**	59.4%	**76.0%**	56.8%
			$r \pm 30°, b \pm 20\%$	**61.4**%	-	-	-	-	-	**68.4%**	13.0%	**68.2%**	9.0%
		ImageNet	$r \pm 30°, b \pm 20\%$	**26.8**%	-	-	-	-	-	**37.4%**	22.4%	**36.8%**	21.2%
Scaling and Brightness	Differentially Resolvable, Composition	MNIST	$s \pm 50\%, b \pm 50\%$	**96.6**%	-	-	-	-	-	**97.8%**	24.8%	**97.8%**	15.6%
		CIFAR-10	$s \pm 30\%, b \pm 30\%$	**54.2**%	-	-	-	-	-	**67.2%**	17.4%	**66.8%**	11.6%
		ImageNet	$s \pm 30\%, b \pm 30\%$	**23.4**%	-	-	-	-	-	**36.4%**	16.0%	**36.0%**	8.8%
Rotation, Brightness, and ℓ_2	Differentially Resolvable, Composition	MNIST	$r \pm 50°, b \pm 20\%, \|\delta\|_2 \leqslant .05$	**96.6**%	-	-	-	-	-	**97.6%**	10.8%	**97.4%**	9.0%
		CIFAR-10	$r \pm 10°, b \pm 10\%, \|\delta\|_2 \leqslant .05$	**64.2**%	-	-	-	-	-	**71.6%**	31.8%	**71.2%**	29.6%
			$r \pm 30°, b \pm 20\%, \|\delta\|_2 \leqslant .05$	**55.2**%	-	-	-	-	-	**65.2%**	0.8%	**64.0%**	0.4%
		ImageNet	$r \pm 30°, b \pm 20\%, \|\delta\|_2 \leqslant .05$	**26.6**%	-	-	-	-	-	**37.0%**	17.6%	**36.4%**	14.0%
Scaling, Brightness, and ℓ_2	Differentially Resolvable, Composition	MNIST	$s \pm 50\%, b \pm 50\%, \|\delta\|_2 \leqslant .05$	**96.4**%	-	-	-	-	-	**97.6%**	22.2%	**97.6%**	12.2%
		CIFAR-10	$s \pm 30\%, b \pm 30\%, \|\delta\|_2 \leqslant .05$	**51.2**%	-	-	-	-	-	**65.0%**	4.4%	**61.8%**	2.6%
		ImageNet	$s \pm 30\%, b \pm 30\%, \|\delta\|_2 \leqslant .05$	**22.6**%	-	-	-	-	-	**36.0%**	7.4%	**35.6%**	4.8%

Vanilla Models and Baselines

We compare with vanilla (undefended) models and baselines from related work. The vanilla models are trained to achieve high accuracy only on clean data. For fairness, on all datasets we use the same model architectures as in our approach. On the test subset, the *benign accuracy* of vanilla models is 98.6%/88.6%/74.4% on MNIST/CIFAR-10/ImageNet. We also report their empirical robust accuracy under attacks in Table 7.3. Since vanilla models are not smoothed, we cannot have certified robust accuracy for them. In terms of baselines, we consider the approaches that provide certification against semantic transformations: DeepG [22], Interval [343], VeriVis [292], Semantify-NN [268], and DistSPT [118]. In Appendix E.8.4, we provide more detailed discussion and comparison with these baseline approaches, and list how we run these approaches for fair comparison.

7.7.2 Main Results

Here, we present our main results from five aspects: (1) certified robustness compared to baselines; (2) empirical robustness comparison; (3) certification time statistics; (4) empirical robustness under unforeseen physical attacks; (5) certified robustness under attacks exceeding the certified radii.

Certified Robustness Compared to Baselines

Our results are summarized in Table 7.3. For each transformation, we ensure that our setting is either the same as or strictly stronger than all other baselines.[5] When our setting is strictly stronger, the baseline setting is shown in corresponding parentheses, and our certified robust accuracy implies a higher or equal certified robust accuracy in the corresponding baseline setting. To our best knowledge, we are the first to provide certified robustness for Gaussian blur, brightness, composition of rotation and brightness, etc. Moreover, on the large-scale standard ImageNet dataset, we are the first to provide nontrivial certified robustness against certain semantic attacks. Note that DistSPT [118] is theoretically feasible to provide robustness certification for the ImageNet dataset. However, its certification is not tight enough to handle ImageNet and it provides robustness certification for only a certain random attack instead of arbitrarily worst-case attacks [118, Section 7.4]. We observe that, across transformations, our framework *significantly* outperforms the state of the art, if present, in terms of robust accuracy. For example, on the composition of contrast and brightness, we improve the certified robust accuracy from 74% to 97.6% on MNIST, from 0.0% (failing to certify) to 82.4% on CIFAR-10, and from 0% (absence of baseline) to 61.4% on ImageNet. On the rotation transformation, we improve the certified robust accuracy from 92.48% to 97.4% on MNIST, from 49.37% to 63.6% on CIFAR-10 (rotation angle within 30°), and from 16% against a certain random attack to 30.4% against arbitrary attacks on ImageNet. Some baselines are able to provide certification under other certification goals and the readers can refer to Appendix E.8.4 for a detailed discussion.

Comparison of Empirical Robust Accuracy

In Table 7.3, we report the empirical robust accuracy for both (undefended) vanilla models and trained TSS models. The empirical robust accuracy is either evaluated under random

[5]The only exception is Semantify-NN [268] on brightness and contrast changes, where Semantify-NN considers these changes composed with clipping to $[0, 1]$ while we consider pure brightness and contrast changes to align with other baselines. We refer the reader to Appendix E.8.4 for a detailed discussion.

attack or two adaptive attacks–Random+ and PGD attack. When it is under adaptive attacks, we report the lower accuracy to evaluate against stronger attackers.

1. For almost all settings, TSS models have significantly higher *empirical robust accuracy*, which means that TSS models are also practical in terms of defending against existing attacks. The only exception is rotation and scaling on ImageNet. The reason is that a single rotation/scaling transformation is too weak to attack even an undefended model. At the same time, our robustness certification comes at the cost of benign accuracy, which also affects the empirical robust accuracy. This exception is eliminated when rotation and scaling are composed with other transformations.

2. Similar observations arise when comparing the *empirical robust accuracy of the vanilla model with the certified robust accuracy of ours*. Hence, even compared to *empirical* metrics, our *certified* robust accuracy is nontrivial and guarantees high accuracy.

3. Our *certified* robust accuracy is always lower or equal compared to the *empirical* one, verifying the validity of our robustness certification. The gaps range from $\sim$ 2% on MNIST to $\sim$ 10% - 20% on ImageNet. Since empirical robust accuracy is an upper bound of the certified accuracy, this implies that our certified bounds are usually tight, particularly on small datasets.

4. The adaptive attack decreases the empirical accuracy of TSS models *slightly*, while it decreases that of vanilla models significantly. Taking contrast and brightness on CIFAR-10 as example, TSS accuracy decreases from 86% to 85.8% while the vanilla model accuracy decreases from 21.0% to 9.6%. Thus, TSS is still robust against adaptive attacks. Indeed, TSS has robustness guarantee against any attack within the certified radius.

Certification Time Statistics

Our robustness certification time is usually less than 100 s on MNIST and 200 s on CIFAR-10; on ImageNet it is around 200 s - 2000 s. Compared to other baselines, ours is slightly faster and achieves much higher certified robustness. For fairness, we give 1000 s time limit per instance when running baselines on MNIST and CIFAR-10. Note that other baselines cannot scale up to ImageNet. Our approach is scalable due to the blackbox nature of smoothing-based certification, the tight interpolation error upper bound, and the efficient progressive sampling strategy. Details on hyperparameters including smoothing variance and average certification time are given in Appendix E.8.6.

Table 7.4: Comparison of **empirical accuracy** of different models under physical corruptions (CIFAR-10-C and ImageNet-C) and **certified accuracy** against composition of transformations. TSS achieves higher or comparable empirical accuracy against unforeseen corruptions and significantly higher certified accuracy (under attack radii in Table 7.3).

	CIFAR-10			ImageNet		
	Vanilla	AugMix [146]	TSS	Vanilla	AugMix [146]	TSS
Empirical Accuracy on CIFAR-10-C and ImageNet-C	53.9%	65.6%	**67.4%**	18.3%	**25.7%**	21.9%
Certified Accuracy against Composition of Gaussian Blur, Translation, Brightness, and Contrast	0.0%	0.4%	**58.2%**	0.0%	0.0%	**32.8%**

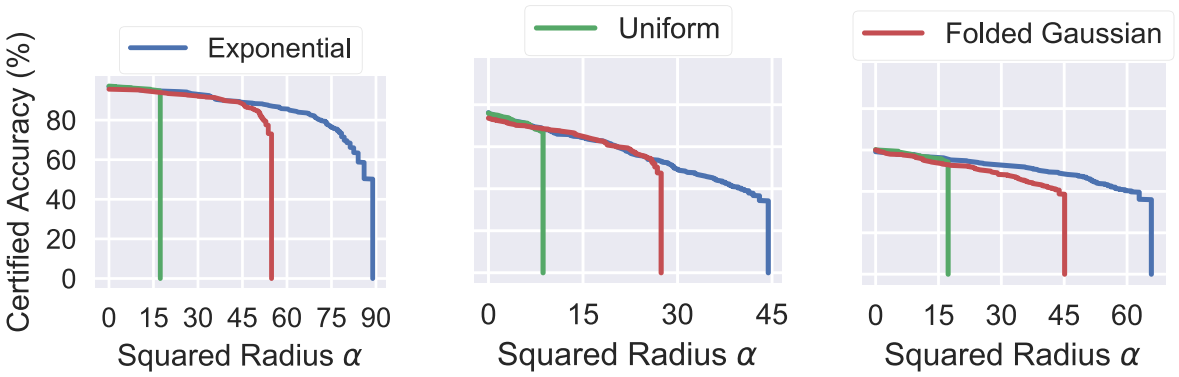
Generalization to Unforeseen Common Corruptions

Are TSS models still more robust when it comes to potential unforeseen physical attacks? To answer this question, we evaluate the robustness of TSS models on the realistic CIFAR-10-C and ImageNet-C datasets [144]. These two datasets are comprised of corrupted images from CIFAR-10 and ImageNet. They apply around 20 types of common corruptions to model *physical attacks*, such as fog, snow, and frost. We evaluate the *empirical robust accuracy* against the highest corruption level (level 5) to model the strongest physical attacker. We apply TSS models trained against a transformation composition attack, Gaussian blur + brightness + contrast + translation, to defend against these corruptions. We select two baselines: vanilla models and AugMix [146]. AugMix is the state-of-the-art model on CIFAR-10-C and ImageNet-C [86].

The results are shown in Table 7.4. The answer is *yes*—TSS models are more robust than undefended vanilla models. It even exceeds the state of the art, AugMix, on CIFAR-10-C. On ImageNet-C, TSS model's empirical accuracy is between vanilla and AugMix. We emphasize that in contrast to TSS, both vanilla and AugMix fail to provide robustness certification. Details on evaluation protocols and additional findings are in Appendix E.8.8.

Evaluation on Attacks beyond Certified Radii

The semantic attacker in the physical world may not constrain itself to be within the specified attack radii. In Appendix E.8.9 we present a thorough evaluation of TSS's robustness when the attack radii go beyond the certified ones. We show, for example, for TSS model defending against $\pm 40\%$ brightness change on ImageNet, when the radius increases to 50%, the certified accuracy only slightly drops from 70.4% to 70.0%. In a nutshell, there is no significant or immediate degradation on both certified robust accuracy and empirical robust accuracy when the attack radii go beyond the certified ones.



(a) MNIST (b) CIFAR-10 (c) ImageNet

Figure 7.8: Certified accuracy for different smoothing distributions for Gaussian blur. On MNIST/CIFAR-10/ImageNet the noise std. is 10/5/10.

7.7.3 Ablation Studies

Here, we provide two ablation studies: (1) Comparison of different smoothing distributions; (2) Comparison of different smoothing variances. In Appendix E.8.11, we present another ablation study on different numbers of samples for differentially resolvable transformations, which reveals a tightness-efficiency trade-off.

Comparison of Smoothing Distributions

To study the effects of different smoothing distributions, we compare the certified robust accuracy for Gaussian blur when the model is smoothed by different smoothing distributions. We consider three smoothing distributions, namely exponential (blue line), uniform (green line), and folded Gaussian (red line). On each dataset, we adjust the distribution parameters such that each distribution has the same variance. All other hyperparameters are kept the same throughout training and certification. As shown Figure 7.8, we notice that on all three datasets, the exponential distribution has the highest average certified radius. This observation is in line with our theoretical reasoning in Section 7.4.2.

Comparison of Different Smoothing Variances

The variance of the smoothing distribution is a hyperparameter that controls the accuracy-robustness trade-off. In Table 7.5, we evaluate different smoothing variances for several transformations on ImageNet and report both the certified accuracy and benign accuracy. The results on MNIST and CIFAR-10 and more discussions are in Appendix E.8.10. From these results, we observe that usually, when the smoothing variance increases, the benign

Table 7.5: Study of the impact of different smoothing variance levels on certified robust accuracy and benign accuracy on **ImageNet** for TSS. The attack radii are consistent with Table 7.3. "Dist." refers to both training and smoothing distribution. The variance used in Table 7.3 is labeled in gray.

Transformation	Attack Radii	Certified Accuracy and Benign Accuracy under Different Variance Levels			
Gaussian Blur	$\alpha \leqslant 36$	Dist. of α	Exp(1/5)	Exp(1/10)	Exp(1/20)
		Cert. Rob. Acc.	0.0%	**51.6%**	48.4%
		Benign Acc.	**63.4%**	59.2%	53.2%
Translation (Reflection Pad.)	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 100$	Dist. of $(\Delta x, \Delta y)$	$\mathcal{N}(0, 20^2 I)$	$\mathcal{N}(0, 30^2 I)$	$\mathcal{N}(0, 40^2 I)$
		Cert. Rob. Acc.	0.0%	50.0%	**55.4%**
		Benign Acc.	70.0%	**72.6%**	70.0%
Brightness	$b \pm 40\%$	Dist. of (c, b)	$\mathcal{N}(0, 0.3^2 I)$	$\mathcal{N}(0, 0.4^2 I)$	$\mathcal{N}(0, 0.5^2 I)$
		Cert. Rob. Acc.	**70.2%**	70.0%	67.6%
		Benign Acc.	**73.2%**	72.2%	69.4%
Contrast	$c \pm 40\%$	Dist. of (c, b)	$\mathcal{N}(0, 0.3^2 I)$	$\mathcal{N}(0, 0.4^2 I)$	$\mathcal{N}(0, 0.5^2 I)$
		Cert. Rob. Acc.	58.4%	63.6%	**65.0%**
		Benign Acc.	**72.8%**	71.4%	68.6%
Rotation	$r \pm 30^\circ$	Dist. of ϵ	$\mathcal{N}(0, 0.25^2 I)$	$\mathcal{N}(0, 0.50^2 I)$	$\mathcal{N}(0, 1.00^2 I)$
		Cert. Rob. Acc.	9.8%	**30.4%**	20.0%
		Benign Acc.	**55.6%**	46.2%	32.2%
Scaling	$s \pm 30\%$	Dist. of ϵ	$\mathcal{N}(0, 0.25^2 I)$	$\mathcal{N}(0, 0.50^2 I)$	$\mathcal{N}(0, 1.00^2 I)$
		Cert. Rob. Acc.	7.2%	**26.4%**	17.4%
		Benign Acc.	**58.8%**	50.8%	33.8%

accuracy drops and the certified robust accuracy first rises and then drops. This tendency is also observed in classical randomized smoothing [77, 427]. However, the range of acceptable variance is usually wide. Thus, without careful tuning the smoothing variances, we are able to achieve high certified and benign accuracy as reported in Table 7.3 and Table E.1.

7.8 RELATED WORK

Semantic Attacks for Neural Networks. Recent work has shown that semantic transformations are able to mislead ML models [124, 153, 416]. For instance, image rotations and translations can attack ML models with 40% - 99% degradation on MNIST, CIFAR-10, and ImageNet on both vanilla models and models that are robust against ℓ_p-bounded perturbations [110]. Brightness/contrast attacks can achieve 91.6% attack success on CIFAR-10, and 71%-100% attack success rate on ImageNet [144]. Our evaluation on empirical robust accuracy (Table 7.3) for vanilla models also confirms these observations. Moreover, brightness attacks have been shown to be of practical concern in autonomous driving [291]. Empirical defenses against semantic transformations have been investigated [110, 144].

Certified Robustness against Semantic Transformations. While heuristic defenses against semantic attacks have been proposed, *provable* robustness requires further investigation. Existing certified robustness against transformations is based on heuristic enumeration, interval bound propagation, linear relaxation, or smoothing. Efficient enumeration in VeriVis [292] can handle only discrete transformations. Interval bound propagation has been used to certify common semantic transformations [22, 118, 343]. To tighten the interval bounds, linear relaxations are introduced. DeepG [22] optimizes linear relaxations for given semantic transformations, and Semantify-NN [268] encodes semantic transformations by neural networks and applies linear relaxations for NNs [402, 446]. However, linear relaxations are loose and computationally intensive compared to our TSS. Recently, Fischer et al. [118] have applied a smoothing scheme to provide provable robustness against transformations but on the large ImageNet dataset, it can provide certification only against random attacks that draw transformation parameters from a pre-determined distribution. More details are available in Appendix E.8.4.

7.9 CONCLUSION

In this chapter, we have presented a unified framework, TSS, for certifying ML robustness against general semantic adversarial transformations. Extensive experiments have shown that TSS significantly outperforms the state of the art or, if no previous work exists, set new baselines. In future work, we plan to further improve the efficiency and tightness of our robustness certification and explore more transformation-specific smoothing strategies.

CHAPTER 8: ROBUSTNESS IN REINFORCEMENT LEARNING AGAINST OBSERVATION PERTURBATIONS: CROP

Reinforcement learning (RL) has been widely applied to different applications, such as robotics [92, 188, 296], autonomous driving vehicles [316, 328], and trading [7, 95, 434]. However, recent studies have shown that learning algorithms are vulnerable to adversarial attacks [112, 132, 170, 199, 270], and a range of attacks have also been proposed against the input states and trained policies of RL [28, 156, 189, 232]. As more and more safety-critical applications are being deployed in real-world [68, 74, 116, 376], how to test and improve their robustness before massive production is of great importance.

To defend against adversarial attacks in RL, different *empirical defenses* have been proposed [29, 98, 113, 117, 254, 280, 290, 332, 447]. In particular, adversarial training [29, 189, 290] and regularized RL algorithms by enforcing the smoothness of the trained models [332, 447] have been studied to improve the robustness of trained policies. However, several strong adaptive attacks have been proposed against these empirical defenses [127, 158, 314] and it is important to provide *robustness certification* for a given learning algorithm to end such repeated game between attackers and defenders.

To provide *robustness certification*, several studies have been conducted on classification. Considering the sequential decision making property of RL, which makes it more challenging to be directly certified compared to classification, in this chapter we ask: *How to provide efficient and effective robustness certification for RL algorithms? What criteria should be used to certify the robustness of RL algorithms?*

Different from classification which involves one-step prediction only, RL algorithms provide both action prediction and reward feedback, making *what to certify* and *how to certify* robustness of RL challenging. In this chapter we focus on Q-learning and propose two certification criteria: *per-state action stability* and *lower bound of perturbed cumulative reward*. In particular, to certify the *per-state action stability*, we propose the *local smoothing* on each input state and therefore derive the certified radius for perturbation at each state, within which the action prediction will not be altered. To certify the *lower bound of cumulative reward*, we propose both *global smoothing* over the finite trajectory to obtain the expectation or percentile bounds given trajectories smoothed with sampled noise sequences; and *local smoothing* to calculate an absolute lower bound based on our adaptive search algorithm.

We leverage our framework to test nine empirically robust RL algorithms on multiple RL environments. We show that the certified robustness depends on both the algorithm and the environment properties. For instance, RadialRL [280] is the most certifiably robust method on Freeway. In addition, based on the per-state certification, we observe that for

some environments such as Pong, some states are more certifiably robust and such pattern is periodic. Given the information of which states are more vulnerable, it is possible to design robust algorithms to specifically focus on these vulnerable states. Based on the lower bound of perturbed cumulative reward, we show that our certification is tight by comparing our bounds with empirical results under adversarial attacks.

Technical Contributions. In this chapter, we take an important step towards providing robustness certification for Q-learning. We make contributions on both theoretical and empirical fronts.

- We propose a framework for certifying the robustness of Q-learning algorithms, which is notably the *first* that provides the robustness certification w.r.t. the cumulative reward.
- We propose two robustness certification criteria for Q-learning algorithms, together with corresponding certification algorithms based on global and local smoothing strategies.
- We theoretically prove the certification radius for input state and lower bound of perturbed cumulative reward under bounded adversarial state perturbations.
- We conduct extensive experiments to provide certification for nine empirically robust RL algorithms on multiple RL environments. We provide several interesting observations which would further inspire the development of robust RL algorithms. Our implementation is publicly available at `https://github.com/AI-secure/CROP`.

8.1 PRELIMINARIES: Q-LEARNING AND DEEP Q-NETWORKS (DQNS)

Markov decision processes (MDPs) are at the core of RL. Our focus is on discounted discrete-time MDPs, which are defined by tuple $(\mathcal{S}, \mathcal{A}, R, P, \gamma, d_0)$, where $\mathcal{S}$ is a set of states (each with dimensionality N), $\mathcal{A}$ represents a set of discrete actions, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $P : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ is the transition function with $\mathcal{P}(\cdot)$ defining the set of probability measures, $\gamma \in [0, 1]$ is the discount factor, and $d_0 \in \mathcal{P}(\mathcal{S})$ is the distribution over the initial state. At time step t, the agent is in the state $s_t \in \mathcal{S}$[6]. After choosing action $a_t \in \mathcal{A}$, the agent transitions to the next state $s_{t+1} \sim P(s_t, a_t)$ and receives reward $R(s_t, a_t)$.

[6]Following the routine in RL literature, in this chapter and Chapter 9, we use normal instead of the bolded font for s_t and other vectors.

The goal is to learn a policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ that maximizes the expected cumulative reward $\mathbb{E}[\sum_t \gamma^t r_t]$.

Q-learning [398] learns an action-value function (Q-function), $Q^\star(s,a)$, which is the maximum expected cumulative reward the agent can achieve after taking action a in state s: $Q^\star(s,a) = R(s,a) + \gamma \mathop{\mathbb{E}}_{s' \sim P(s,a)} \left[\max_{a'} Q^\star(s',a')\right]$. In deep Q-Networks (DQNs) [265], $Q^\star$ is approximated using a neural network parametrized by θ, i.e., $Q^\pi(s,a;\theta) \approx Q^\star(s,a)$. Let $\rho \in \mathcal{P}(\mathcal{S} \times \mathcal{A})$ be the observed distribution defined over states s and actions a, the network can be trained via minimizing loss function

$$\mathcal{L}(\theta) = \mathop{\mathbb{E}}_{(s,a)\sim\rho, s'\sim P(s,a)} \left[\left(R(s,a) + \gamma \max_{a'} Q^\pi\left(s', a'; \theta\right) - Q^\pi(s,a;\theta) \right)^2 \right]. \tag{8.1}$$

The greedy policy π is defined as taking the action with highest Q^π value in each state s: $\pi(s) = \arg\max_{a\in\mathcal{A}} Q^\pi(s,a)$.

8.2 ROBUSTNESS CERTIFICATION IN Q-LEARNING

In this section, we first introduce the threat model, followed by two robustness certification criteria for the Q-learning algorithm: *per-state action* and *cumulative reward*. We consider the standard adversarial setting in Q-learning [156, 189, 447], where the adversary can apply ℓ_2-bounded perturbation $B_\epsilon = \{\delta \in \mathbb{R}^n : \|\delta\|_2 \leqslant \epsilon\}$ to input state observations of the agent during decision (test) time to cause the policy to select suboptimal actions. The agent observes the perturbed state and takes action $a' = \pi(s+\delta)$, following policy π. Following the Kerckhoff's principle [329], we consider a *worst-case* adversary who applies adversarial perturbations to *every* state at decision time. Our analysis and methods are generalizable to other ℓ_p norms following [204, 427].

8.2.1 Robustness Certification for Q-learning with Different Criteria

To provide the robustness certification for Q-learning, we propose two certification criteria: *per-state action robustness* and *lower bound of the cumulative reward*.

Robustness Certification for Per-State Action. We first aim to explore the robustness (stability/consistency) of the per-state action given adversarially perturbed input states.

Definition 8.1 (Certification for Per-State Action)**.** Given a trained network Q^π with policy π, we define the robustness certification for per-state action as the *maximum perturbation*

magnitude $\bar{\epsilon}$, such that for any perturbation $\delta \in B_{\bar{\epsilon}}$, the predicted action under the perturbed state will be the same as the action taken in the clean environment: $\pi(s+\delta) = \pi(s)$, $\forall \delta \in B_{\bar{\epsilon}}$.

Robustness Certification for Cumulative Reward. Given that the cumulative reward is important for RL, here in addition to the per-state action, we also define the robustness certification regarding the cumulative reward under input state perturbation.

Definition 8.2 (Cumulative reward)**.** Let $P : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ be the transition function of the environment with $\mathcal{P}(\cdot)$ defining the set of probability measures. Let $R, d_0, \gamma, Q^\pi, \pi$ be the reward function, initial state distribution, discount factor, a given trained Q-network, and the corresponding greedy policy as introduced in Section 8.1. $J(\pi)$ represents the *cumulative reward* and $J_\epsilon(\pi)$ represents the *perturbed cumulative reward* under perturbations $\delta_t \in B_\epsilon$ at each time step t:

$$\begin{aligned} J(\pi) &:= \sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)), \\ \text{where } s_{t+1} &\sim P(s_t, a_t), s_0 \sim d_0, \end{aligned} \quad \text{and} \quad \begin{aligned} J_\epsilon(\pi) &:= \sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t + \delta_t)), \\ \text{where } s_{t+1} &\sim P(s_t, \pi(s_t + \delta_t)), s_0 \sim d_0. \end{aligned} \tag{8.2}$$

The randomness of $J(\pi)$ arises from the environment dynamics, while that of $J_\epsilon(\pi)$ includes additional randomness from the perturbations $\{\delta_t\}$. We focus on a finite horizon H in this chapter, where a sufficiently large H can approximate $J(\pi)$ and $J_\epsilon(\pi)$ to arbitrary precision when $\gamma < 1$ and reward is bounded.

Definition 8.3 (Robustness Certification for Cumulative Reward)**.** The robustness certification for cumulative reward is the *lower bound* of *perturbed cumulative reward* $\underline{J}$ such that $\underline{J} \leqslant J_\epsilon(\pi)$ under perturbation in $B_\epsilon = \{\delta \in \mathbb{R}^n : \|\delta\|_2 \leqslant \epsilon\}$ applied to all time steps.

We will provide details on the certification of per-state action in Section 8.3 and the certification of cumulative reward in Section 8.4 based on different smoothing strategies and certification methods.

8.3 ROBUSTNESS CERTIFICATION STRATEGIES FOR PER-STATE ACTION

In this section, we discuss the robustness certification for *per-state action*, aiming to calculate a lower bound of *maximum perturbation magnitude* $\bar{\epsilon}$ in Definition 8.1.

8.3.1 Certification for Per-State Action via Action-value Functional Smoothing

Let Q^π be the action-value function given by the trained network Q with policy π. We derive a smoothed function $\widetilde{Q}^\pi$ through per-state *local smoothing*. Specifically, at each time step t, for each action $a \in \mathcal{A}$, we draw random noise from a Gaussian distribution $\mathcal{N}(0, \sigma^2 \boldsymbol{I}_N)$ to smooth $Q^\pi(\cdot, a)$.

$$\widetilde{Q}^\pi(s_t, a) := \mathop{\mathbb{E}}_{\Delta_t \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_N)} Q^\pi(s_t + \Delta_t, a) \quad \forall s_t \in \mathcal{S}, a \in \mathcal{A}, \tag{8.3}$$

$$\text{and} \quad \tilde{\pi}(s_t) := \arg\max_a \widetilde{Q}^\pi(s_t, a) \quad \forall s_t \in \mathcal{S}. \tag{8.4}$$

Lemma 8.1 (Lipschitz Continuity of Smoothed Value Function)**.** Given the action-value function $Q^\pi : \mathcal{S} \times \mathcal{A} \to [V_{\min}, V_{\max}]$, the smoothed function $\widetilde{Q}^\pi$ with smoothing parameter σ is L-Lipschitz continuous with $L = \frac{V_{\max} - V_{\min}}{\sigma} \sqrt{\frac{2}{\pi}}$ w.r.t. the state input.

The proof is given in Appendix F.1.1. Leveraging the Lipschitz continuity in Lemma 8.1, we derive the following theorem for certifying the robustness of per-state action.

Theorem 8.1. Let $Q^\pi : \mathcal{S} \times \mathcal{A} \to [V_{\min}, V_{\max}]$ be a trained value network, $\widetilde{Q}^\pi$ be the smoothed function with (8.3). At time step t with state s_t, we can compute the lower bound r_t of maximum perturbation magnitude $\bar{\epsilon}(s_t)$ (i.e., $r_t \leqslant \bar{\epsilon}(s_t)$, $\bar{\epsilon}$ defined in Definition 8.1) for locally smoothed policy $\tilde{\pi}$:

$$r_t = \frac{\sigma}{2} \left(\Phi^{-1} \left(\frac{\widetilde{Q}^\pi(s_t, a_1) - V_{\min}}{V_{\max} - V_{\min}} \right) - \Phi^{-1} \left(\frac{\widetilde{Q}^\pi(s_t, a_2) - V_{\min}}{V_{\max} - V_{\min}} \right) \right), \tag{8.5}$$

where Φ^{-1} is the inverse CDF function, a_1 is the action with the highest $\widetilde{Q}^\pi$ value at state s_t, and a_2 is the runner-up action. We name lower bound r_t as certified radius for the state s_t.

The proof is omitted to Appendix F.1.2. The theorem provides a certified radius r_t for per-state action given smoothed policy: As long as the perturbation is bounded by r_t, i.e., $\|\delta_t\|_2 \leqslant r_t$, the action does not change: $\tilde{\pi}(s_t + \delta_t) = \tilde{\pi}(s_t)$. To achieve high certified robustness for per-state action, Theorem 8.1 implies a tradeoff between value function smoothness and the margin between the values of top two actions: If a larger smoothing parameter σ is applied, the action-value function would be smoother and therefore more stable; however, it would shrink the margin between the top two action values leading to smaller certified radius. Thus, there exists a proper smoothing parameter to balance the tradeoff, which depends on the actual environments and algorithms.

8.3.2 CROP-LoAct: Local Randomized Smoothing for Certifying Per-State Action

Next we introduce the algorithm to achieve the certification for per-state action. Given a Q^π network, we apply (8.3) to derive a smoothed network $\widetilde{Q}^\pi$. At each state s_t, we obtain the greedy action $\tilde{a}_t$ w.r.t. $\widetilde{Q}^\pi$, and then compute the certified radius r_t. We present the complete algorithm in Appendix F.2.1.

There are some additional challenges in smoothing the value function and computing the certified radius in Q-learning compared with the standard classification task [77]. **Challenge 1:** In classification, the output range of the confidence $[0, 1]$ is known a priori; however, in Q-learning, for a given Q^π, its range $[V_{\min}, V_{\max}]$ is unknown. **Challenge 2:** In the classification task, the lower and upper bounds of the top two classes' prediction probabilities can be directly computed via the confidence interval base on multinomial proportions [133]. For Q-networks, the outputs are not probabilities and calculating the multinomial proportions becomes challenging.

Pre-processing. To address **Challenge 1**, we estimate the output range $[V_{\min}, V_{\max}]$ of a given network Q^π based on a finite set of valid states $\mathcal{S}_{\text{sub}} \subseteq \mathcal{S}$. In particular, we craft a sufficiently large set $\mathcal{S}_{\text{sub}}$ to estimate $V_{\min}$ and $V_{\max}$ for Q^π on $\mathcal{S}$, which can be used later in per-state smoothing. The details for estimating $V_{\min}$ and $V_{\max}$ are deferred to Appendix F.2.1.

Certification. To smooth a given state s_t, we use Monte Carlo sampling [77] to sample noise applied to s_t, and then estimate the corresponding smoothed value function $\widetilde{Q}^\pi$ at s_t with (8.3). In particular, we sample m Gaussian noise $\Delta_i \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_N)$, clip the Q-network output to ensure that it falls within the range $[V_{\min}, V_{\max}]$, and then take the average of the output to obtain the smoothed action prediction based on $\widetilde{Q}^\pi$. We then employ Theorem 8.1 to compute the certified radius r_t. We omit the detailed inference procedure to Appendix F.2.1. To address **Challenge 2**, we leverage Hoeffding's inequality [150] to compute a lower bound of $\widetilde{Q}^\pi(s_t, a_1)$ and an upper bound of $\widetilde{Q}^\pi(s_t, a_2)$ with one-sided confidence level parameter α given the top two actions a_1 and a_2. When the former is higher than the latter, we can certify a positive radius for the given state s_t.

8.4 ROBUSTNESS CERTIFICATION STRATEGIES FOR THE CUMULATIVE REWARD

In this section, we present robustness certification strategies for the cumulative reward. The goal is to provide the lower bounds for the *perturbed cumulative reward* in Definition 8.2.

In particular, we propose both *global smoothing* and *local smoothing* strategies to certify the perturbed cumulative reward. In the global smoothing, we view the whole state trajectory as a function to smooth, which would lead to relatively loose certification bound. We then propose the local smoothing by smoothing each state individually to obtain the absolute lower bound.

8.4.1 Certification of Cumulative Reward based on Global Smoothing

In contrast to Section 8.3 where we perform per-state smoothing to achieve the certification for per-state action, here, we aim to perform *global smoothing* on the state trajectory by viewing the entire trajectory as a function. In particular, we first derive the *expectation bound* of the cumulative reward based on global smoothing by estimating the Lipschitz constant for the cumulative reward w.r.t. the trajectories. Since the Lipschitz estimation in the expectation bound is algorithm agnostic and could lead to loose estimation bound, we subsequently propose a more practical and tighter *percentile bound*.

Definition 8.4 (σ-Randomized Trajectory and σ-Randomized Policy)**.** Given a state trajectory $(s_0, s_1, \ldots, s_{H-1})$ of length H where $s_{t+1} \sim P(s_t, \pi(s_t))$, $s_0 \sim d_0$, with π the greedy policy of the action-value function Q^π, we derive a *σ-randomized trajectory* as $(s'_0, s'_1, \ldots, s'_{H-1})$, where $s'_{t+1} \sim P(s'_t, \pi(s'_t + \Delta_t))$, $\Delta_t \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_N)$, and $s'_0 = s_0 \sim d_0$. We correspondingly define a *σ-randomized policy* π' based on π in the following form: $\pi'(s_t) := \pi(s_t + \Delta_t)$ where $\Delta_t \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_N)$.

Let the operator $\oplus$ concatenates given input states or noise that are added to each state. The sampled noise sequence is denoted by $\Delta = \oplus_{t=0}^{H-1} \Delta_t$, where $\Delta_t \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_N)$.

Definition 8.5 (Perturbed Return Function)**.** Let R, P, γ, d_0 be the reward function, transition function, discount factor, and initial state distribution in Definition 8.2. We define a bounded *perturbed return function* $F_\pi : \mathbb{R}^{H \times N} \to [J_{\min}, J_{\max}]$ representing cumulative reward with potential perturbation δ:

$$F_\pi \left(\oplus_{t=0}^{H-1} \delta_t \right) := \sum_{t=0}^{H} \gamma^t R(s_t, \pi(s_t + \delta_t)), \quad \text{where } s_{t+1} \sim P(s_t, \pi(s_t + \delta_t)), s_0 \sim d_0. \tag{8.6}$$

We can see when there is no perturbation ($\delta_t = \mathbf{0}$), $F_\pi(\oplus_{t=0}^{H-1} \mathbf{0}) = J(\pi)$; when there are adversarial perturbations $\delta_t \in B_\epsilon$ at each time step, $F_\pi(\oplus_{t=0}^{H-1} \delta_t) = J_\epsilon(\pi)$, i.e., *perturbed cumulative reward*.

Mean Smoothing: Expectation Bound. Here we propose to sample noise sequences Δ to perform *global smoothing* for the entire state trajectory, and calculate the lower bound of the expected perturbed cumulative reward $\mathbb{E}_{\Delta}\left[J_{\epsilon}(\pi')\right]$ under all possible ℓ_2-bounded perturbations within magnitude ϵ. The expectation is over the noise sequence Δ involved in the σ-randomized policy π' in Definition 8.4.

Lemma 8.2 (Lipschitz Continuity of Smoothed Perturbed Return Function)**.** Let F be the perturbed return function function defined in (8.6), the smoothed perturbed return function $\widetilde{F}_{\pi}$ is $\frac{(J_{\max}-J_{\min})}{\sigma}\sqrt{\frac{2}{\pi}}$-Lipschitz continuous, where

$$\widetilde{F}_{\pi}\left(\oplus_{t=0}^{H-1}\delta_t\right) := \mathop{\mathbb{E}}_{\Delta\sim\mathcal{N}(0,\sigma^2 I_{H\times N})} F_{\pi}\left(\oplus_{t=0}^{H-1}(\delta_t+\Delta_t)\right). \tag{8.7}$$

Theorem 8.2 (Expectation Bound)**.** Let

$$\underline{J_E} = \widetilde{F}_{\pi}\left(\oplus_{t=0}^{H-1}\mathbf{0}\right) - L\epsilon\sqrt{H}, \quad \text{where} \quad L = \frac{(J_{\max}-J_{\min})}{\sigma}\sqrt{\frac{2}{\pi}}. \tag{8.8}$$

Then $\underline{J_E} \leqslant \mathbb{E}\left[J_{\epsilon}(\pi')\right]$.

Proof sketch. We first derive the equality between expected perturbed cumulative reward $\mathbb{E}\left[J_{\epsilon}(\pi')\right]$ and the smoothed perturbed return function $\widetilde{F}_{\pi}(\oplus_{t=0}^{H-1}\delta_t)$. Thus, to lower bound the former, it suffices to lower bound the latter, which can be calculated leveraging the Lipschitz continuity of $\widetilde{F}$ in Lemma 8.2 (proved in Appendix F.1.3), noticing that the distance between $\oplus_{t=0}^{H-1}\mathbf{0}$ and the adversarial perturbations $\oplus_{t=0}^{H-1}\delta_t$ is bounded by $\epsilon\sqrt{H}$. The complete proof is omitted to Appendix F.1.4.

We obtain $J_{\min}$ and $J_{\max}$ in Lemma 8.2 from environment specifications which can be loose in practice. Thus the Lipschitz constant L estimation is coarse and mean smoothing is usually loose. We next present a method that circumvents estimating the Lipschitz constant and provides a tight percentile bound.

Percentile Smoothing: Percentile Bound. We now propose to apply *percentile smoothing* to smooth the *perturbed cumulative reward* and obtain the lower bound of the p-th percentile of $J_{\epsilon}(\pi')$, where π' is a σ-randomized policy defined in Definition 8.4.

$$\widetilde{F}_{\pi}^{p}\left(\oplus_{t=0}^{H-1}\delta_t\right) = \sup_y\left\{y\in\mathbb{R} : \Pr\left[F_{\pi}\left(\oplus_{t=0}^{H-1}(\delta_t+\Delta_t)\right)\leqslant y\right]\leqslant p\right\}. \tag{8.9}$$

Theorem 8.3 (Percentile Bound)**.** Let $\underline{J_p} = \widetilde{F}_{\pi}^{p'}\left(\oplus_{t=0}^{H-1}\mathbf{0}\right)$, where $p' := \Phi\left(\Phi^{-1}(p) - \frac{\epsilon\sqrt{H}}{\sigma}\right)$. Then $\underline{J_p} \leqslant$ the p-th percentile of $J_\epsilon(\pi')$.

The proof is provided in Appendix F.1.5 based on Chiang et al. [70]. There are several other advantages of percentile smoothing over mean smoothing. *First*, the certification given by percentile smoothing is among the cumulative rewards of the sampled σ-randomized trajectories and is therefore achievable by a real-world policy, while the expectation bound is less likely to be achieved in practice given the loose Lipschitz bound. *Second*, for a discrete function such as perturbed return function, the output of mean smoothing is continuous w.r.t. σ, while the results given by percentile smoothing remain discrete. Thus, the percentile smoothed function preserves properties of the base function before smoothing, and shares similar interpretation, e.g., the number of rounds that the agent wins. *Third*, taking $p = 50\%$ in percentile smoothing leads to the *median smoothing* which achieves additional properties such as robustness to outliers [255]. The detailed algorithm CROP-GRe, including the inference procedure, the estimation of $\widetilde{F}_\pi$, the calculation of the empirical order statistics, and the configuration of algorithm parameters $J_{\min}, J_{\max}$, are deferred to Appendix F.2.2.

8.4.2 Certification of Cumulative Reward based on Local Smoothing

Though global smoothing provides efficient and practical bounds for the perturbed cumulative reward, such bounds are still loose as they involve smoothing the entire trajectory at once. In this section, we aim to provide a tighter lower bound for $J_\epsilon(\tilde{\pi})$ by performing *local smoothing*.

Given a trajectory of H time steps which is guided by the locally smoothed policy $\tilde{\pi}$, we can compute the certified radius at each time step according to Theorem 8.1, which can be denoted as $r_0, r_1, \ldots, r_{H-1}$. Recall that when the perturbation magnitude $\epsilon < r_t$, the optimal action a_t at time step t will remain unchanged. This implies that when $\epsilon < \min_{t=0}^{H-1} r_t$, none of the actions in the entire trajectory will be changed, and therefore the lower bound of the cumulative reward when $\epsilon < \min_{t=0}^{H-1} r_t$ is the return of the current trajectory in a deterministic environment. Increasing ϵ has two effects. First, the *total* number of time steps where the action is susceptible to change will increase; second, at *each* time step, the action can change from the best to the runner-up or the rest. We next introduce an extension of certified radius r_t to characterize the two effects.

Theorem 8.4. Let $(r_t^1, \ldots, r_t^{|\mathcal{A}|-1})$ be a sequence of certified radii for state s_t at time step t, where r_t^k denotes the radius such that if $\epsilon < r_t^k$, the possible action at time step t will belong to the actions corresponding to top k action values of $\widetilde{Q}$ at state s_t. The definition of

r_t in Theorem 8.1 is equivalent to r_t^1 here. The radii can be computed similarly as follows:

$$r_t^k = \frac{\sigma}{2}\left(\Phi^{-1}\left(\frac{\widetilde{Q}^\pi(s_t, a_1) - V_{\min}}{V_{\max} - V_{\min}}\right) - \Phi^{-1}\left(\frac{\widetilde{Q}^\pi(s_t, a_{k+1}) - V_{\min}}{V_{\max} - V_{\min}}\right)\right), \quad 1 \leqslant k < |\mathcal{A}|, \quad (8.10)$$

where a_1 is the action of the highest $\widetilde{Q}$ value at state s_t and a_{k+1} is the $(k+1)$-th best action. We additionally define $r_t^0(s_t) = 0$, which is also compatible with the definition above.

We defer the proof in Appendix F.1.6. With Theorem 8.4, for any given ϵ, we can compute all possible actions under perturbations in B_ϵ. This allows an exhaustive search to traverse all trajectories satisfying that all certified radii along the trajectory are smaller than ϵ. Then, we can conclude that $J_\epsilon(\tilde{\pi})$ is lower bounded by the minimum return over all these possible trajectories.

CROP-LoRe: Local Smoothing for Certified Reward

Given a policy π in a deterministic environment, let the initial state be s_0, we propose CROP-LoRe to certify the lower bound of $J_\epsilon(\tilde{\pi})$. At a high-level, CROP-LoRe *exhaustively* explores new trajectories leveraging Theorem 8.4 with priority queue and *effectively* updates the lower bound of cumulative reward $\underline{J}$ by expanding a trajectory tree dynamically. The algorithm returns a collection of pairs $\{(\epsilon_i, \underline{J}_{\epsilon_i})\}_{i=1}^{|C|}$ sorted in ascending order of ϵ_i, where $|C|$ is the length of the collection. For all ϵ, let i be the largest integer such that $\epsilon_i \leqslant \epsilon < \epsilon_{i+1}$, then as long as the perturbation magnitude $\epsilon \leqslant \epsilon'$, the cumulative reward $J_\epsilon(\tilde{\pi}) \geqslant \underline{J}_{\epsilon_i}$. The algorithm is shown in Algorithm F.3 in Appendix F.2.3.

Algorithm Description. The method starts from the base case: when perturbation magnitude $\epsilon = 0$, the lower bound of cumulative reward $\underline{J}$ is exactly the benign reward. The method then gradually increases the perturbation magnitude ϵ (later we will explain how a new ϵ is determined). Along the increase of ϵ, the perturbation may cause the policy π to take different actions at some time steps, thus resulting in new trajectories. Thanks to the local smoothing, the method leverages Theorem 8.4 to figure out the exhaustive list of possible actions under current perturbation magnitude ϵ, and effectively explore these new trajectories by formulating them as expanded branches of a trajectory tree. Once all new trajectories are explored, the method examines all leaf nodes of the tree and figures out the minimum reward among them, which is the new lower bound of cumulative reward $\underline{J}$ under this new ϵ. To mitigate the explosion of branches, CROP-LoRe proposes several

optimization tricks. In the following, we will briefly introduce the *trajectory exploration and expansion* and *growth of perturbation magnitude* steps, as well as the *optimization* tricks. More algorithm details are deferred to Appendix F.2.3, where we also provide an analysis on the time complexity of the algorithm.

In **trajectory exploration and expansion**, CROP-LoRe organizes all possible trajectories in the form of search tree and progressively grows it. For each node (representing a state), leveraging Theorem 8.4, we compute a non-decreasing sequence $\{r^k(s)\}_{k=0}^{|\mathcal{A}|-1}$ representing the required perturbation radii for π to choose each alternative action. Suppose the current ϵ satisfies $r^i(s) \leqslant \epsilon < r^{i+1}(s)$, we can grow $(i+1)$ branches from current state s corresponding to the original action and i alternative actions since $\epsilon \geqslant r^j(s)$ for $1 \leqslant j \leqslant i$. We expand the tree branches using depth-first search [363]. In **perturbation magnitude growth**, when all trajectories for perturbation magnitude ϵ are explored, we increase ϵ to seek certification under larger perturbations. This is achieved by preserving a priority queue [377] of the critical ϵ's that we will expand on. Concretely, along the trajectory, at each tree node, we search for the possible actions and store actions corresponding to $\{r^k(s)\}_{k=i+1}^{|\mathcal{A}|-1}$ into the priority queue, since these actions are exactly those need to be explored when ϵ grows. We repeat the procedure of *perturbation magnitude growth* (i.e., popping out the head element from the queue) and *trajectory exploration and expansion* (i.e., exhaustively expand all trajectories given the perturbation magnitude) until the priority queue becomes empty or the perturbation magnitude ϵ reaches the predefined threshold. Additionally, we adopt a few **optimization** tricks commonly used in search algorithms to reduce the complexity of the algorithm, such as pruning and the memorization technique [261]. More potential improvements are discussed in Appendix F.2.3.

So far, we have presented all our certification methods.

8.4.3 Discussion on the Certification Methods

We discuss the *advantages* and *limitations* of our certification methods, as well as possible direct *extensions*, hoping to pave the way for future research along similar directions. We also provide more *detailed analysis* that help with the understanding of our algorithms.

CROP-LoAct. The algorithm provides state-wise robustness certification in terms of the stability/consistency of the per-state action. It treats each time step *independently*, smooths the given state at the given time step, and provides the corresponding certification. Thus, one potential *extension* is to expand the time window from one time step to a consecutive sequence of several time steps, and provide certification for the sequences of actions for the

given window of states.

CROP-GRe. As explained in Section 8.4, the expectation bound $\underline{J_E}$ is too loose to have any practical usage. In comparison, percentile bound $\underline{J_p}$ is much tighter and practical. However, one limitation of $\underline{J_p}$ is that there exists an upper bound of the attack magnitude ϵ that can be certified for each σ, as explained in Appendix F.2.2. For attack magnitudes that exceed the upper bound, we can obtain no useful information via this certification (though the upper bound is usually sufficiently large).

One limitation of CROP-GRe. Lemma 8.2 requires knowing the noise added to each step beforehand so as to generate m randomized trajectories given the known noise sequence $\{\delta_t\}_{t=0}^{H-1}$. Thus, it cannot be applied against an adaptive attacker. We clarify that our CROP-LoAct and CROP-LoRe does not have such limitation.

CROP-LoRe. The algorithm provides the absolute lower bound $\underline{J}$ of the cumulative reward for any finite-horizon trajectory with a given initial state. The advantage of the algorithm is that $\underline{J}$ is an absolute lower bound that bounds the worst-case situation (apart from the exceptions due to the probabilistic confidence α), rather than the statistical lower bounds $\underline{J_E}$ and $\underline{J_p}$ that characterize the statistical properties of the random variable J_ϵ.

One potential pitfall in understanding $\underline{J}$. What CROP-LoRe certifies is the lower bound for the trajectory with *a given initial state* $s_0 \sim d_0$, rather than all possible states in d_0. This is because our search starts from a root node of the tree, which is set to the fixed given state s_0.

Confidence of CROP-LoRe. Regarding the *confidence* of our probabilistic certification, we clarify that we consider the independent multiple-test. Concretely, due to the independence of decision-making errors, the confidence is $(1-\alpha)^N$, where N is the maximum number of possible attacked states explored by CROP-LoRe. Formally,

$$\begin{aligned}
&\Pr[\text{CROP-LoRe certification holds}] \qquad (8.11)\\
=&\prod_{\substack{\text{all } s \text{ explored}\\ \text{by CROP-LoRe}}} \Pr[\text{not make error on } s | \text{not make error on } s_{pre}]\\
\overset{\text{(i)}}{=}&\prod_{\text{all attackable } s} (1-\alpha) \times \prod_{\text{all unattackable } s} 1\\
\overset{\text{(ii)}}{=}&(1-\alpha)^N. \qquad (8.12)
\end{aligned}$$

In the above equation, we can see that (i) leverages the independence which in turn gives a bound of form $(1-\alpha)^N$. This is because in each step, the event of "certification does not hold"

is *independent* since we sample Gaussian noise *independently*. Leveraging such independence, we obtain a confidence lower bound of $(1-\alpha)^N$. From (ii), we see that the confidence is only related to the number of possible attacked states N. This is because for unattackable states, the certification deterministically holds since there is no attack at current step. Therefore, we only need to count the confidence intervals from all possibly attackable steps instead of all states explored. We remark that in practice, the attacker usually has the ability to perturb only a limited number of steps, i.e., N is typically small. Therefore, the confidence is non-trivial, and it could be further mitigated by adapting non-smoothing-based per-state action certification, e.g., GCP-CROWN in Chapter 3.

Main limitation of CROP-LoRe. The main *limitation* is the high time complexity of the algorithm (details see Appendix F.2.3). The algorithm has exponential time complexity in worst case despite the existence of several optimizations. Therefore, it is not suitable for environments with a large action set, or when the horizon length is set too long.

Extension to Policy-based Methods. Though we specifically study the robustness certification in Q-learning in this chapter, our two certification criteria and three certification strategies can be readily extended to policy-based methods. The intuition is that, instead of smoothing the value function in the Q-learning setting, we directly smooth the policy function in policy-based methods. With the smoothing module replaced and the theorems updated, other technical details in the algorithms for certifying the per-state action and the cumulative reward would then be similar.

8.5 EXPERIMENTS

In this section, we present evaluation for the proposed robustness certification framework CROP. Concretely, we apply our three certification algorithms (CROP-LoAct, CROP-GRe, and CROP-LoRe) to certify nine RL methods (**StdTrain** [265], **GaussAug** [189], **AdvTrain** [29], **SA-MDP (PGD,CVX)** [447], **RadialRL** [280], **CARRL** [111], **NoisyNet** [121], and **GradDQN** [290]) on two high-dimensional Atari games (Pong and Freeway), one low dimensional control environment (CartPole), and an autonomous driving environment (Highway). As a summary, we find that (1) SA-MDP (CVX), SA-MDP (PGD), and RadialRL achieve high certified robustness in different environments; (2) Large smoothing variance can help to improve certified robustness significantly on Freeway, while a more careful selection of the smoothing parameter is needed in Pong; (3) For methods that demonstrate high certified robustness, our certification of the cumulative reward is tight. We defer the detailed descriptions of the environments, the introduction to the RL methods, imple-

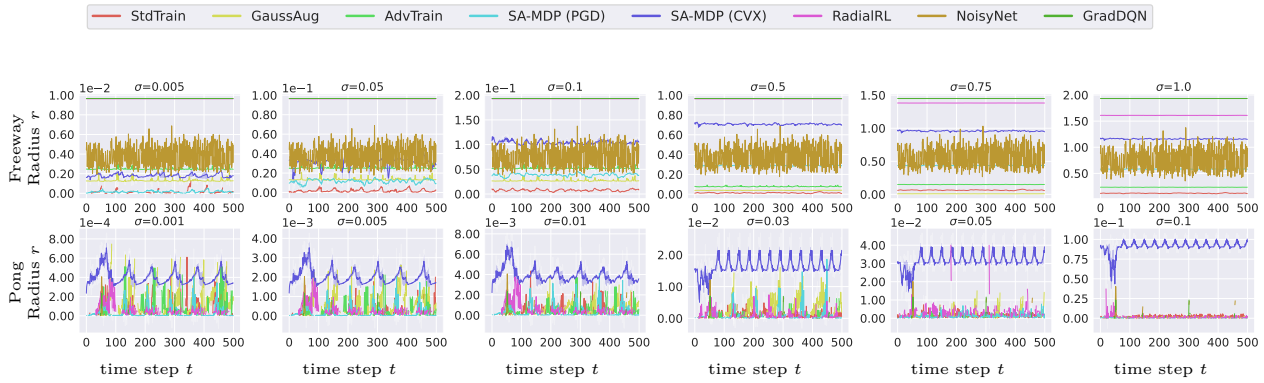


Figure 8.1: Robustness certification for *per-state action* in terms of certified radius r at all time steps. Each column corresponds to a smoothing variance σ. The shaded area represents the standard deviation which is small. RadialRL is the most certifiably robust method on Freeway, while SA-MDP (CVX) is the most robust on Pong.

mentation details, full results, and discussions to the full version of this chapter [408] and the open-source leaderboard: `https://crop-leaderboard.github.io/`.

8.5.1 Evaluation of Robustness Certification for Per-State Action

In this subsection, we provide the robustness certification evaluation for per-state action.

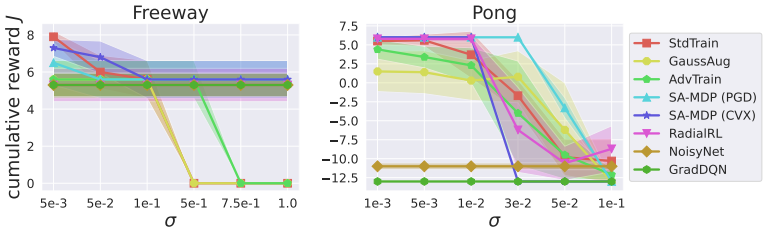


Figure 8.2: Benign performance of locally smoothed policy $\tilde{\pi}$ under different smoothing variance σ with clean state observations.

Experimental Setup and Metrics. We evaluate the *locally smoothed policy* $\tilde{\pi}$ derived in (8.3). We follow Theorem 8.1 and report the *certified radius* r_t at each step t. We additionally compute *certified ratio* η *at radius* r, representing the percentage of time steps that can be certified, formally denoted as $\eta_r = {}^1\!/_H \sum_{t=1}^{H} \mathbb{I}[r_t \geqslant r]$. More details are omitted to [408].

Evaluation Results of CROP-LoAct. We present the certification comparison in Figure 8.1 and the benign performance under different smoothing variances in Figure 8.2; we omit details of certified ratio to [408].

On *Freeway*, we evaluate with a large range of smoothing parameter σ up to 1.0, since Freeway can tolerate large noise as shown in Figure 8.2. Results under even larger σ are deferred to [408], showing that the certified robustness can be further improved for the empirically robust methods. From Figure 8.1, we see that *RadialRL consistently achieves the highest certified radius across all σ's.* This is because RadialRL explicitly optimizes over the worst-case perturbations. StdTrain, GaussAug, and AdvTrain are not as robust, and increasing σ will not make a difference. In *Pong*, SA-MDP (CVX) is the most certifiably robust, which may be due to the hardness of optimizing the worst-case perturbation in Pong. More interesting, all methods present similar periodic patterns for the certified radius on different states, which would inspire further robust training methods to take the "confident state" (e.g., when the ball is flying towards the paddle) into account. Overall, the certified robustness of these methods largely matches empirical observations [29, 280, 447].

8.5.2 Evaluation of Robustness Certification for Cumulative Reward

Here we discuss the evaluation for the robustness certification regarding cumulative reward in Section 8.4. We show the evaluation results for both CROP-GRe and CROP-LoRe.

Evaluation Setup and Metrics. We evaluate the *σ-randomized policy* π' derived in Definition 8.4 for CROP-GRe and the *locally smoothed policy* $\tilde{\pi}$ derived in (8.3) for CROP-LoRe. We compute the expectation bound $\underline{J_E}$, percentile bound $\underline{J_p}$, and absolute lower bound $\underline{J}$ following Theorem 8.2, Theorem 8.3, and Section 8.4.2. To validate the tightness of the bounds, we additionally perform empirical attacks.

Evaluation of CROP-GRe. We present the certification of reward in Figure 8.3 and mainly focus on analyzing percentile bound. We present the bound w.r.t. different attack magnitude ϵ under different smoothing parameter σ. For each σ, there exists an upper bound of ϵ that can be certified, which is positively correlated with σ. Details see Appendix F.2.2. We observe similar conclusion as in the per-state action certification that RadialRL is most certifiably robust for Freeway, while SA-MDP (CVX,PGD) are most robust for Pong although the highest robustness is achieved at different σ.

Evaluation of CROP-LoRe. In Figure 8.3, we note that in *Freeway*, RadialRL is the most robust, followed by SA-MDP (CVX), then SA-MDP (PGD), AdvTrain, and GaussAug. As for *Pong*, SA-MDP (CVX) outperforms RadialRL and SA-MDP (PGD). Remaining methods are not clearly ranked.

8.5.3 Discussion on Evaluation Results

Impact of Smoothing Parameter σ. We draw similar conclusions regarding the impact of smoothing variance from the results given by CROP-LoAct and CROP-GRe. In *Freeway*, as σ increases, the robustness of StdTrain, GaussAug, and AdvTrain barely increases, while that of SA-MDP (PGD), SA-MDP (CVX), and RadialRL steadily increases. In *Pong*, a σ in the range 0.01-0.03 shall be suitable for almost all methods. For CROP-LoRe, there are a few different conclusions. On *Freeway*, it is clear that under larger attack magnitude ϵ, larger smoothing parameter σ always secures higher lower bound $\underline{J}$. In *Pong*, CROP-LoRe can only achieve non-zero certification with small σ under a small range of attack magnitude, implying the difficulty of establishing non-trivial certification for these methods on Pong. We defer details on the selection of σ to [408].

Tightness of the certification $\underline{J_E}$, $\underline{J_p}$, and $\underline{J}$. We compare the empirical cumulative rewards achieved under PGD attacks with our certified lower bounds. First, the empirical results are consistently lower bounded by our certifications, validating the correctness of our bounds. Regarding the tightness, the improved $\underline{J_p}$ is much tighter than the loose $\underline{J_E}$, supported by discussions in Section 8.4.1. The tightness of $\underline{J}$ can be reflected by the zero gap between the certification and the empirical result under a wide range of attack magnitude ϵ. Furthermore, the certification for SA-MDP (CVX,PGD) are quite tight for Freeway under large attack magnitudes, and RadialRL demonstrates its superiority on Freeway. Additionally, different methods may achieve the same empirical results under attack, yet their certifications differ tremendously, indicating the importance of the robustness *certification*.

Game Properties. The certified robustness of any method on Freeway is much higher than that on Pong, indicating that Freeway is a more stable game than Pong, also shown in Mnih et al. [266].

8.6 RELATED WORK

8.6.1 Evasion Attacks in RL

We consider the adversarial attacks on state observations, where the attacker aims to add perturbations to the state during test time, so as to achieve certain **adversarial goals**, such as misleading the *action* selection and minimizing the *cumulative reward*. We discuss the related works that fall into these two categories below.

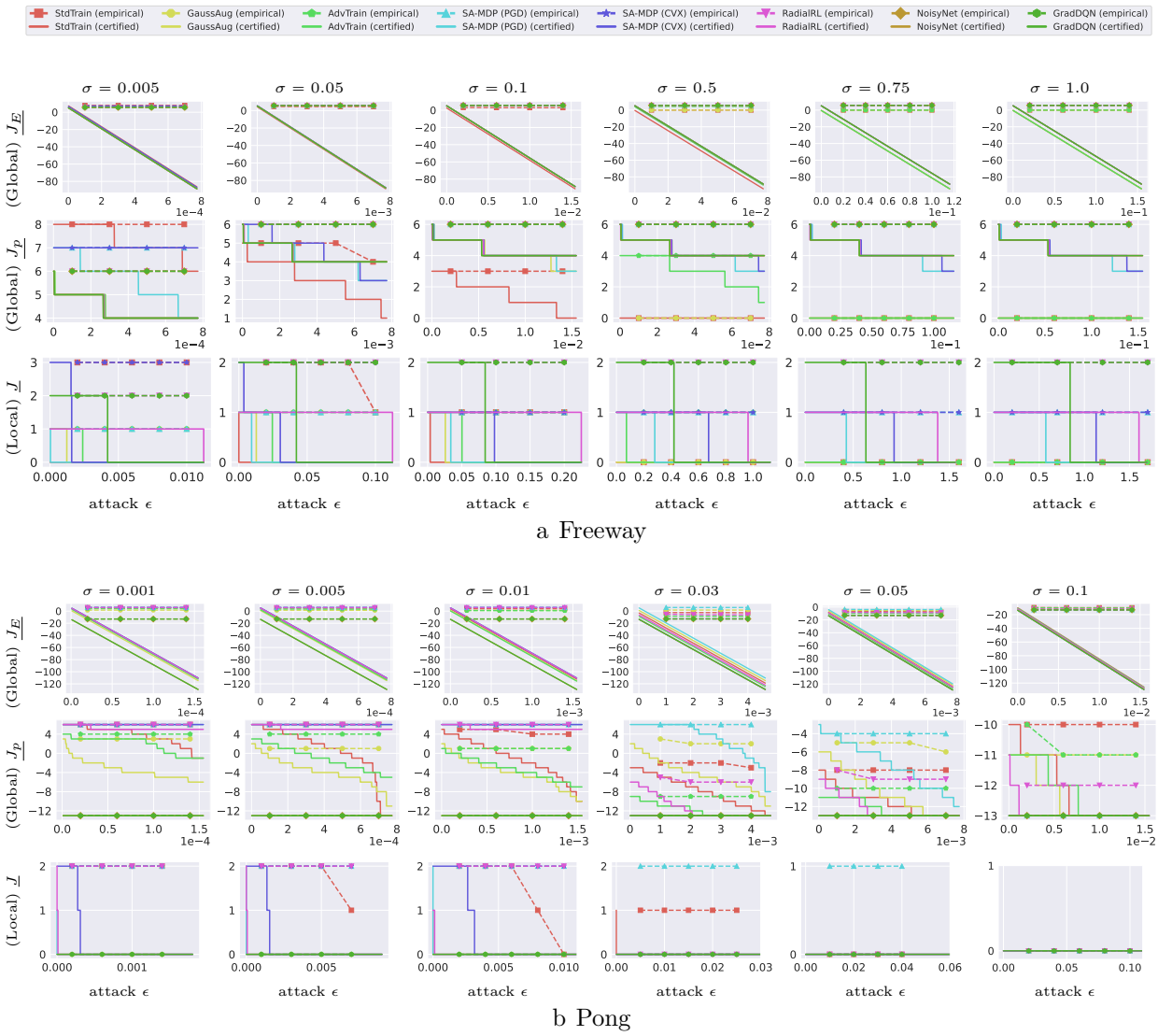


Figure 8.3: Robustness certification for cumulative reward, including *expectation bound* $\underline{J_E}$, *percentile bound* $\underline{J_p}$ ($p = 50\%$), and *absolute lower bound* $\underline{J}$. Each column corresponds to one smoothing variance. Solid lines represent the certified reward bounds, and dashed lines show the empirical performance under PGD.

Misleading Action Selection. It has been shown that different methods—applying *random noise* to simply interfere with the action selection [189], or adopting *adversarial attacks* (e.g., FGSM [132] and CW attacks [54]) to deliberately alter the probability of action selection—are effective to different degrees. More concretely, when manipulating the action selection probability, some works aim to reduce the probability of selecting the optimal action [29, 156, 189], while others try to increase the probability of selecting the worst action [232, 290].

Minimizing the Cumulative Reward. ATLA [447] and PA-AD [358] consider an *optimal adversary* under the SA-MDP framework [447], which aims to lead to minimal value functions under bounded state perturbations. To find this *optimal adversary* (i.e., the optimal adversarial state perturbation), ATLA [447] proposes to train an adversary whose action space is the perturbation set in the state space, while PA-AD [358] further decouples the problem of finding state perturbations into finding policy perturbations plus finding the state that achieves the lowest value policy, thus addressing the challenge of large state space.

8.6.2 Robust RL

Distributionally Robust RL. Nilim and El Ghaoui [277] and Iyengar [162] consider the problem of distributionally robust RL, where there is normally an uncertainty set with prior information regarding the distribution of the uncertain parameters. Iyengar [162] directly puts constraints on environmental dynamics and reward functions by assuming that they take values in an uncertainty set. Another line of research assumes the environmental dynamics and reward function as the uncertain parameters and are sampled from an uncertain distribution in the uncertain set. The uncertainty set is in general state-wise independent, i.e., "s-rectangularity" in Nilim and El Ghaoui [277]. Both types aim to derive a policy by solving a max-min problem, with the former taking the minimum directly over the parameters, while the latter taking the minimum over the distribution.

Empirically Robust RL against Evasion Attacks. *Randomization methods* [4, 369] were first proposed to encourage exploration. This type of method was later systematically studied for its potential to improve model robustness. NoisyNet [121] adds parametric noise to the network's weight during training, providing better resilience to both training-time and test-time attacks [29, 30], also reducing the transferability of adversarial examples, and enabling quicker recovery with fewer number of transitions during phase transition.

Under the *adversarial training* framework, Kos and Song [189] and Behzadan and Munir [29] show that re-training with random noise and FGSM perturbations increases the resilience against adversarial examples. Pattanaik et al. [290] leverage attacks using an engineered loss function specifically designed for RL to significant increase the robustness to parameter variations. RS-DQN [117] is an *imitation learning* based approach that trains a robust student-DQN in parallel with a standard DQN in order to incorporate the constrains such as SOTA adversarial defenses [251, 262].

SA-DQN [447] is a *regularization* based method that adds regularizers to the training loss function to encourage the top-1 action to stay unchanged under perturbation.

Built on top of the *neural network verification* algorithms [134, 402], Radial-RL [280] proposes to minimize an adversarial loss function that incorporates the upper bound of the perturbed loss, computed using certified bounds from verification algorithms. CARRL [111] aims to compute the lower bounds of action-values under potential perturbation and select actions according to the worst-case bound, but it relies on linear bounds [402] and is only suitable for low-dimensional environments.

We emphasize two main differences between the contributions of prior works and our CROP. **First**, most of these robust RL methods only provide empirical robustness against perturbed state inputs during test time, but cannot provide theoretical guarantees for the performance of the trained models under any bounded perturbations; while our CROP framework can provide practically computable *certified robustness* w.r.t. two robustness certification criteria (per-state action stability and cumulative reward lower bound), i.e., for a given RL algorithm, we can compute certifications for it that indicate its robustness. (We will discuss a few more related work that can provide certified robustness in Section 8.6.3.) **Second**, most of these Robust RL methods focus on only the per-state decision, while we additionally consider the *certification of trajectories* to obtain the lower bound of cumulative rewards.

8.6.3 Robustness Certification for RL

Despite the abundant literature in robustness certification in supervised learning, there is almost no work on robustness certification for RL, given the unclear criteria and the intrinsic difficulty of the task. In this part, we first introduce another two works that can achieve robustness certification at state-level, and then another concurrent work Kumar et al. [196] which also aims to provide provable robustness regarding the cumulative reward for RL.

Robustness Certification on State Level. Fischer et al. [117] and Zhang et al. [447] have discussed the state level robustness certification as well, but the way they obtain the certification is different from our CROP. Concretely, we achieve robustness by probabilistic certification via randomized smoothing [77], while Fischer et al. [117] and Zhang et al. [447] directly achieve robustness by deterministic certification via leveraging the neural network verification techniques [262, 424, 443].

Despite these works on state-level robustness certification for RL, we are the *first* to provide the certification of cumulative rewards. We emphasize that the certification of lower bound of the cumulative reward in RL is more challenging than the per-state certification considering the dynamic nature of RL. Our main contribution of the chapter indeed mainly

focuses on providing an efficient adaptive search based algorithm CROP-LoRe to certify the lower bound of the cumulative reward together with rigorous analysis of the certification.

Robustness Certification for Cumulative Rewards. We compare our robustness certification for cumulative rewards with another concurrent work [196].

In terms of the *threat model*, both Kumar et al. [196] and CROP consider the type of adversary that can perturb the state observations of the agent during test time. In our CROP, we consider a perturbation budget per time step following previous works [29, 156, 189, 290, 447], while they assume a budget for the entire episode. The two perspectives are closely related.

Regarding the *certification criteria*, we certify both *per-state action stability* and *cumulative reward lower bound*. Although they also consider the cumulative reward in their certification goal, they formulate the certification as a classification problem—certifying whether the cumulative reward is above a threshold or not, in contrast to directly certifying the lower bound as in our case.

For the *certification technique*, both works are developed based on randomized smoothing proposed in supervised learning [77]. We propose a global smoothing technique (CROP-GRe), as well as an adaptive search algorithm coupled with local smoothing (CROP-LoRe) to achieve the certification of cumulative reward; while they propose an adaptive version of the Neyman-Pearson lemma, which is similar with our global smoothing. Among the three algorithms (CROP-GRe, CROP-LoRe, and Kumar et al. [194]), CROP-GRe cannot defend against an adaptive adversary (as concretely stated in Section 8.4.3) while the other two can; specifically, CROP-LoRe is much more sophisticated and tight than the global smoothing ones, which is an important contribution in our work.

8.7 SUMMARY

To provide the robustness certification for RL methods, we propose a general framework CROP, aiming to provide certification based on two criteria. Our evaluations show that certain empirically robust RL methods are certifiably robust for specific RL environments.

CHAPTER 9: ROBUSTNESS IN OFFLINE REINFORCEMENT LEARNING AGAINST DATA POISONING: COPA

Reinforcement learning (RL) has been widely applied to a range of applications. In particular, offline RL [216] is proposed to leverage previously observed data to train the policy without requiring expensive interaction with environments or online data collection, and enables the reuse of training data [3]. However, offline RL also raises a great safety concern on poisoning attacks [184, 385, 392]. A recent survey of industry reports that data poisoning is significantly more concerning than other threats [198]. For offline RL, the situation is even worse, and a recent theoretical study shows that the robust offline RL against poisoning attacks is a strictly harder problem than online RL [456].

Although there are several empirical and certified defenses for classification tasks against poisoning attacks [213, 293, 400], it is challenging and shown ineffective to directly apply them to RL given its complex structure [184]. Thus, robust offline RL against poisoning attacks remains largely unexplored with no mention of robustness certification. In addition, though some theoretical analyses provide general robustness bounds for RL, they either assume a bounded distance between the learned and Bellman optimal policies or are limited to linear MDPs [456]. To the best of our knowledge, there is no robust RL method that is able to provide *practically computable certified robustness* against poisoning attacks. In this chapter, we tackle this problem by proposing the first framework of Certifying robust policies for general offline RL against poisoning attacks (**COPA**).

Certification Criteria. One critical challenge in certifying robustness for offline RL is the certification criteria, since the prediction consistency is no longer the only goal as in classification. We extend the two criteria in Chapter 8: *per-state action stability* and *cumulative reward bound*. The former guarantees that at a specific time, the policy learned with COPA will predict the same action before and after attacks under certain conditions. This is important for guaranteeing the safety of the policy at critical states, e.g., braking when seeing pedestrians. For cumulative reward bound, a lower bound of the cumulative reward for the policy learned with COPA is guaranteed under certain poisoning conditions. This directly guarantees the worst-case overall performance.

COPA Framework. COPA is composed of two components: policy partition and aggregation protocol and robustness certification method. We propose three policy partition aggregation protocols: **PARL** (Per-State Partition Aggregation), **TPARL** (Temporal Partition Aggregation), and **DPARL** (Dynamic Temporal Partition Aggregation), and propose

certification methods for each of them corresponding to both proposed certification criteria. In addition, for *per-state action stability*, we prove that our certifications for PARL and TPARL are theoretically tight. For *cumulative reward bound*, we propose an adaptive search algorithm, where we compute the possible action set for each state under certain poisoning conditions. Concretely, we propose a novel method to compute the precise action set for PARL and efficient algorithms to compute a superset of the possible action set which leads to sound certification for TPARL and DPARL. We further prove that for PARL our certification is theoretically tight, for TPARL the theoretically tight certification is NP-complete, and for DPARL it is open whether theoretically tight certification exists.

Technical Contributions. We take the first step towards certifying the robustness of offline RL against poisoning attacks, and we make contributions on both theoretical and practical fronts.

- We abstract and formulate the robustness certification for offline RL against poisoning attacks, and we propose two certification criteria: per-state action stability and cumulative reward bound.
- We propose the first framework COPA for certifying robustness of offline RL against poisoning attacks. COPA includes novel policy aggregation protocols and certification methods.
- We prove the tightness of the proposed certification methods for the aggregation protocol PARL. We also prove the computational hardness of the certification for TPARL.
- We conduct thorough experimental evaluation for COPA on different RL environments with three offline RL algorithms, demonstrating the effectiveness of COPA, together with several interesting findings. Code is open source at `https://github.com/AI-secure/COPA` for reproducibility, and all experimental results are actively maintained at `https://copa-leaderboard.github.io/`.

9.1 RELATED WORK

Poisoning attacks [97, 274] are critical threats in machine learning, which are claimed to be more concerning than other threats [198]. Poisoning attacks widely exist in classification [324], and both empirical defenses [57, 237, 293, 353] and certified defenses [167, 213, 400] have been proposed.

After Kiourti et al. [184] show the existence of effective backdoor poisoning attacks in RL, a recent work theoretically and empirically validates the existence of reward poisoning in online RL [453]. Furthermore, Zhang et al. [456] theoretically prove that the offline RL is more difficult to be robustified against poisoning than online RL considering linear MDP. From the defense side, Zhang et al. [456] propose robust variants of the Least-Square Value Iteration algorithm that provides probabilistic robustness guarantees under linear MDP assumption. In addition, Robust RL against reward poisoning is studied in Banihashem et al. [24], but robust RL against general poisoning is less explored. In this background, we aim to provide the certified robustness for general offline RL algorithms against poisoning attacks, which is the first work that achieves the goal.

9.2 CERTIFICATION CRITERIA OF COPA

In this section, we define two robustness certification criteria for offline RL against general poisoning attacks: per-state action stability and cumulative reward bound.

Offline RL. We model the RL environment by an episodic finite-horizon Markov decision process (MDP) $\mathcal{E} = (\mathcal{S}, \mathcal{A}, R, P, H, d_0)$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of discrete actions, $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $P : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ is the stochastic transition function with $\mathcal{P}(\cdot)$ defining the set of probability measures, H is the time horizon, and $d_0 \in \mathcal{P}(\mathcal{S})$ is the distribution of the initial state. At time step t, the RL agent is at state $s_t \in \mathcal{S}$[7]. After choosing action $a_t \in \mathcal{A}$, the agent transitions to the next state $s_{t+1} \sim P(s_t, a_t)$ and receives reward $r_t = R(s_t, a_t)$. After H time steps, the cumulative reward $J = \sum_{t=0}^{H-1} r_t$. We denote a consecutive sequence of all states between time step l and r as $s_{l:r} := [s_l, s_{l+1}, \ldots, s_r]$.

Here we focus on offline RL, for which the threat of poisoning attacks is practical and more challenging to deal with [456]. Concretely, in offline RL, a training dataset $D = \{\tau_i\}_{i=1}^N$ consists of logged trajectories, where each trajectory $\tau = \{(s_j, r_j, a_j, s'_j)\}_{j=1}^l \in (\mathcal{S} \times \mathcal{A} \times \mathbb{R} \times \mathcal{S})^l$ consists of multiple tuples denoting the transitions (i.e., starting from state s_j, taking the action a_j, receiving reward r_j, and transitioning to the next state s'_j).

Poisoning Attacks. Training dataset D can be poisoned in the following manner. For each trajectory $\tau \in D$, the adversary is allowed to replace it with an arbitrary trajectory $\widetilde{\tau}$, generating a manipulated dataset $\widetilde{D}$. We denote $D \ominus \widetilde{D} = (D \backslash \widetilde{D}) \bigcup (\widetilde{D} \backslash D)$ as the *symmetric*

[7]Following the routine in RL literature, in Chapter 8 and this chapter, we use normal instead of the bolded font for s_t and other vectors.

difference between two datasets D and $\widetilde{D}$. For instance, adding or removing one trajectory causes a symmetric difference of magnitude 1, while replacing one trajectory with a new one leads to a symmetric difference of magnitude 2. We refer to the size of the symmetric difference as the *poisoning size*.

Certification Goal. To provide the robustness certification against poisoning attacks introduced above, we aim to certify the *test-time* performance of the trained policy in a clean environment. Specifically, in the *training phase*, the RL training algorithm and our aggregation protocol can be jointly modeled by $\mathcal{M} : \mathcal{D} \to (\mathcal{S}^\star \to \mathcal{A})$ which provides an aggregated policy, where $\mathcal{S}^\star$ denotes the set of all consecutive state sequences. Our **goal** is to provide robustness certification for the poisoned aggregated policy $\tilde{\pi} = \mathcal{M}(\widetilde{D})$, given bounded poisoning size (i.e., $|D \ominus \widetilde{D}| \leqslant \bar{K}$).

Robustness Certification Criteria: Per-state Action Stability. We first aim to certify the robustness of the poisoned policy in terms of the stability of per-state action during test time.

Definition 9.1 (Robustness Certification for Per-state Action Stability)**.** Given a clean dataset D, we define the robustness certification for *per-state action stability* as that for *any* $\widetilde{D}$ satisfying $|D \ominus \widetilde{D}| \leqslant \bar{K}$, the action predictions of the poisoned and clean policies for the state (or state sequence) s are the same, i.e., $\tilde{\pi} = \mathcal{M}(\widetilde{D}), \pi = \mathcal{M}(D), \tilde{\pi}(s) = \pi(s)$, under the tolerable poisoning threshold $\bar{K}$. In an episode, we denote the tolerable poisoning threshold for the state at step t by $\bar{K}_t$.

The definition encodes the requirement that, for a particular state, any poisoned policy will always give the same action prediction as the clean one, as long as the poisoning size is within $\bar{K}$ ($\bar{K}$ computed in Section 9.3). In this definition, s could be either a state or a state sequence, since our aggregated policy (defined in Section 9.2) may aggregate multiple recent states to make an action choice.

Robustness Certification Criteria: Lower Bound of Cumulative Reward. We also aim to certify poisoned policy's overall performance in addition to the prediction at a particular state. Here we measure the overall performance by the cumulative reward $J(\pi)$.

Definition 9.2 (Cumulative Reward)**.** Let $P : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ be the transition function with $\mathcal{P}(\cdot)$ defining the set of probability measures. Let R, d_0, H be the reward function, initial state distribution, and time horizon. We denote $J(\pi)$ as the *cumulative reward* of the

given policy π:

$$J(\pi) := \mathbb{E} \sum_{t=0}^{H-1} R(s_t, a_t), \text{ where } a_t = \pi(s_t), s_{t+1} \sim P(s_t, a_t), s_0 \sim d_0, \tag{9.1}$$

It is natural to adapt this definition when there is a discount factor. If policy π considers recent n_t states instead of only s_t to make action predictions, we can change $a_t = \pi(s_t)$ to $a_t = \pi(s_{t-n_t+1:t})$ in Equation (9.1).

Now we are ready to define the robustness certification for cumulative reward bound.

Definition 9.3 (Robustness Certification for Cumulative Reward Bound)**.** Robustness certification for cumulative reward bound is the *lower bound* of *cumulative reward* $\underline{J}_K$ such that $\underline{J}_K \leqslant J(\tilde{\pi})$ for any $\tilde{\pi} = \mathcal{M}(\widetilde{\mathcal{D}})$ where $|D \ominus \widetilde{D}| \leqslant K$, i.e., $\tilde{\pi}$ is trained on poisoned dataset $\widetilde{D}$ within poisoning size K.

9.3 COPA TRAINING AND AGGREGATION PROTOCOLS

In this section, we introduce our framework COPA, which is composed of *training protocols* and *aggregation protocols*, and *certification methods*. The training protocol combined with an offline RL training algorithm provides subpolicies. The aggregation protocol aggregates the subpolicies as an aggregated policy. In Section 9.4, we will introduce the certification method for COPA. The certification method certifies the robustness of the aggregated policy against poisoning attacks corresponding to different certification criteria provided in Section 9.2.

9.3.1 Partition-Based Training Protocol

COPA's training protocol contains two stages: partitioning and training. We denote D as the entire offline RL training dataset. We abstract an offline RL training algorithm (e.g., DQN) by $\mathcal{M}_0 : 2^D \to \Pi$, where 2^D is the power set of D, and $\Pi = \{\pi : \mathcal{S} \to \mathcal{A}\}$ is the set of trained policies. Each trained policy in Π is a function mapping a given state to the predicted action.

Partitioning Stage. In this stage, we separate the training dataset D into u partitions $\{D_i\}_{i=0}^{u-1}$ that satisfy $\bigcup_{i=0}^{u-1} D_i = D$ and $\forall i \neq j, D_i \cap D_j = \varnothing$. Concretely, when performing partitioning, for each trajectory $\tau \in D$, we *deterministically* assign it to one unique partition. The assignment is only dependent on the trajectory τ itself, and not impacted by any modification to other parts of the training set. One design choice of such a deterministic

assignment is using a deterministic hash function h to compute the assignment, i.e., $D_i = \{\tau \in D \mid h(\tau) \equiv i \pmod{u}\}, \forall i \in [u]$.

Training Stage. In this stage, for each training data partition D_i, we independently apply an RL algorithm $\mathcal{M}_0$ to train a policy $\pi_i = \mathcal{M}_0(D_i)$. Hereinafter, we call these trained polices as *subpolicies* to distinguish from the aggregated policies. Concretely, let $[u] := \{0, 1, \ldots, u-1\}$. For these subpolicies, the *policy indicator* $\mathbb{I}_{i,a} : \mathcal{S} \to \{0, 1\}$ is defined by $\mathbb{I}_{i,a}(s) := \mathbb{I}[\pi_i(s) = a]$, indicating whether subpolicy π_i chooses action a at state s. The *aggregated action count* $n_a : \mathcal{S} \to \mathbb{N}_{\geqslant 0}$ is the number of votes across all the subpolicies for action a given state s: $n_a(s) := |\{i|\pi_i(s) = a, i \in [u]\}| = \sum_{i=0}^{u-1} \mathbb{I}_{i,a}(s)$. Specifically, we denote $n_a(s_{l:r})$ for $\sum_{j=l}^{r} n_a(s_j)$, i.e., the sum of votes for states between time step l and r. A detailed algorithm of the training protocol is in Appendix G.1.1.

Now we are ready to introduce the proposed aggregation protocols in COPA (PARL, TPARL, DPARL) that generate *aggregated policies* based on subpolicies, and corresponding certification.

9.3.2 Aggregation Protocols: PARL, TPARL, DPARL

With u learned subpolicies $\{\pi_i\}_{i=0}^{u-1}$, we propose three different aggregation protocols in COPA to form three types of aggregated policies for each certification criteria: PARL, TPARL, and DPARL.

Per-State Partition Aggregation (PARL). Inspired by aggregation in classification [213], PARL aggregates subpolicies by choosing actions with the highest votes. We denote the PARL aggregated policy by $\pi_{\mathrm{P}} : \mathcal{S} \to \mathcal{A}$. When there are multiple highest voting actions, we break ties deterministically by returning the "smaller" ($<$) action, which can be defined by numerical order, lexicographical order, etc. Throughout the chapter, we assume $\arg\max$ over $\mathcal{A}$ always uses $<$ operator to break ties.

Definition 9.4 (Per-State Partition Aggregation)**.** Given subpolicies $\{\pi_i\}_{i=0}^{u-1}$, the Per-State Partition Aggregation (PARL) protocol defines an aggregated policy $\pi_{\mathrm{P}} : \mathcal{S} \to \mathcal{A}$ such that

$$\pi_{\mathrm{P}}(s) := \arg\max_{a \in \mathcal{A}} n_a(s), \tag{9.2}$$

where $n_a(s)$ is defined in Section 9.2.

The intuition behind PARL is that the poisoning attack within size K can change at most K subpolicies. Therefore, as long as the margin between the votes for top and runner-

up actions is larger than $2K$ for the given state, after poisoning, we can guarantee that the aggregated PARL policy will not change its action choice. We will formally state the robustness guarantee in Section 9.4.1.

Temporal Partition Aggregation (TPARL). In the sequential decision making process of RL, it is likely that certain important states are much more vulnerable to poisoning attacks, which we refer to as *bottleneck states*. Therefore, the attacker may just change the action predictions for these bottleneck states to deteriorate the overall performance, say, the cumulative reward. For example, in Pong game, we may lose the round when choosing an immediate bad action when the ball is closely approaching the paddle. Thus, to improve the overall certified robustness, we need to focus on improving the tolerable poisoning threshold for these bottleneck states. Given such intuition and goal, we propose Temporal Partition Aggregation (TPARL) and the aggregated policy is denoted as π_{T}:

Definition 9.5 (Temporal Partition Aggregation)**.** Given subpolicies $\{\pi_i\}_{i=0}^{u-1}$ and window size W, at time step t, the Temporal Partition Aggregation (TPARL) defines an aggregated policy $\pi_{\mathrm{T}} : \mathcal{S}^{\min\{t+1,W\}} \to \mathcal{A}$ such that

$$\pi_{\mathrm{T}}(s_{\max\{t-W+1,0\}:t}) = \arg\max_{a\in\mathcal{A}} n_a(s_{\max\{t-W+1,0\}:t}), \tag{9.3}$$

where $s_{l:r}$ and n_a are defined in Section 9.2.

In the above definition, at time step t, the input of policy π_{T} contains all states between time step $\max\{t - W + 1, 0\}$ and t within window size W; and the output is the policy prediction with the highest votes across these states. Recall that in Section 9.2, we define $n_a(s_{l:r}) = \sum_{j=l}^{r} n_a(s_j)$. We break ties in the same way as in PARL.

TPARL is based on two insights: (1) Bottleneck states have lower tolerable poisoning threshold, which is because the vote margin between the top and runner-up actions is smaller at such state; (2) Some RL tasks satisfy temporal continuity [208, 379], indicating that good action choices are usually similar across states of adjacent time steps, i.e., adjacent states. Hence, we leverage the subpolicies' votes from adjacent states to enlarge the vote margin, and thus increase the tolerable poisoning threshold. To this end, in TPARL, we predetermine a *window size* W, and choose the action with the highest votes across recent W states.

Dynamic Temporal Partition Aggregation (DPARL). The TPARL uses a fixed window size W across all states. Since the specification of the window size W requires certain prior knowledge, plus that the same fixed window size W may not be suitable for all

states, it is preferable to perform *dynamic temporal aggregation* by using a flexible window size. Therefore, we propose Dynamic Temporal Partition Aggregation (DPARL), which dynamically selects the window size W towards maximizing the tolerable poisoning threshold *per step*. Intuitively, DPARL selects the window size W such that the average vote margin over selected states is maximized. To guarantee that only recent states are chosen, we further constrain the maximum window size ($W_{\max}$).

Definition 9.6 (Dynamic Temporal Partition Aggregation)**.** Given subpolicies $\{\pi_i\}_{i=0}^{u-1}$ and maximum window size $W_{\max}$, at time step t, the Dynamic Temporal Partition Aggregation (DPARL) defines an aggregated policy $\pi_{\mathrm{D}} : \mathcal{S}^{\min\{t+1,W_{\max}\}} \to \mathcal{A}$ such that

$$\pi_{\mathrm{D}}(s_{\max\{t-W_{\max}+1,0\}:t}) := \arg\max_{a\in\mathcal{A}} n_a(s_{t-W'+1:t}), \text{ where } W' = \arg\max_{1\leqslant W\leqslant \min\{W_{\max},t+1\}} \Delta_t^W. \quad (9.4)$$

In the above equation, n_a is defined in Section 9.2 and Δ_t^W is given by

$$\begin{aligned} \Delta_t^W &:= \frac{1}{W}\left(n_{a_1}(s_{t-W+1:t}) - n_{a_2}(s_{t-W+1:t})\right), \\ \text{where} \quad a_1 &= \arg\max_{a\in\mathcal{A}} n_a(s_{t-W+1:t}), a_2 = \arg\max_{a\in\mathcal{A},a\neq a_1} n_a(s_{t-W+1:t}). \end{aligned} \quad (9.5)$$

In the above definition, Δ_t^W encodes the average vote margin between top action a_1 and runner-up action a_2 if choosing window size W. Thus, W' locates the window size with maximum average vote margin, and its corresponding action is selected. Again, we use the mechanism described in PARL to break ties. Robustness certification methods for DPARL are in Sections 9.4.1 and 9.4.2.

9.3.3 Illustration of COPA Aggregation Protocols

We present a concrete example to demonstrate how different aggregation protocols induce different tolerable poisoning thresholds, illustrate bottleneck and non-bottleneck states, and provide additional discussions.

Suppose the action space contains two actions $\mathcal{A} = \{a_1, a_2\}$, and there are six subpolicies $\{\pi_i\}_{i=0}^{u-1}$ where $u = 6$. We are at time step $t = 7$ now. When there is no poisoning, the subpolicies' action predictions for state s_0 to s_7 are shown by Table 9.1. After aggregation, all aggregated policies will choose action a_1 at $t = 7$.

Table 9.1: A concrete example of action predictions, where "1" means action a_1 and "2" means action a_2. When there is no poisoning attack, the corresponding time window spans by PARL, TPARL, and DPARL are shown by green, blue, and pink respectively. All aggregated policies choose action a_1, but have different tolerable poisoning thresholds as shown in the last column.

Action Prediction for	s_0	s_1	s_2	s_3	s_4	s_5	s_6	s_7	$\bar{K}$ at s_7
π_0	1	1	1	1	1	1	1	1	
π_1	1	1	1	1	1	1	1	1	
π_2	1	1	1	1	1	1	1	1	
π_3	1	1	1	1	1	1	1	2	
π_4	1	1	1	1	1	1	1	2	
π_5	1	1	1	2	1	2	2	2	
PARL (π_{P}, Definition 9.4)								1	0
TPARL with $W = 7$ (π_{T}, Definition 9.5)								1	2
DPARL with $W_{\max} = 8$ (π_{D}, Definition 9.6)								1	1

PARL (Definition 9.4). The PARL aggregation protocol only uses the current state s_7 to aggregate the votes. On s_7, three subpolicies choose action a_1 and three others choose action a_2. By breaking the tie with $a_1 < a_2$, the PARL policy $\pi_{\mathrm{P}}(s_7) = a_1$. The tolerable poisoning threshold $\bar{K}_t = 0$, because one action flipping from a_1 to a_2 by poisoning only one subpolicy can change the aggregated policy to a_2.

TPARL (Definition 9.5). The TPARL aggregation protocol uses window size $W = 7$. This implies that, we aggregate all states $s_{1:7}$ to count the votes and decide the action. Since $n_{a_1}(s_{1:7}) = 36$ and $n_{a_2}(s_{1:7}) = 6$, after poisoning two subpolicies, we still $\widetilde{n}_{a_1}(s_{1:7}) \geqslant \widetilde{n}_{a_2}(s_{1:7})$. Thus, we achieve an tolerable poisoning threshold $\bar{K}_t = 2$.

Compared with PARL, the temporal aggregation in TPARL increases the vote margin between a_1 and a_2 and thus improve the tolerable poisoning threshold at current state.

DPARL (Definition 9.6). The DPARL aggregation protocol chooses the window size $W' = 8$ since this window size gives the largest average vote margin $\Delta_t^{W'}$ (defined in Equation (9.4)). We can easily find out that with poisoning size $K = 1$, we will still choose $\widetilde{W}' = 8$ as the window size and the resulting votes for a_1 can be kept higher than votes for a_2. However, when it comes to $K = 2$, if we totally flip subpolicies π_0 and π_1 to let them always predict action a_2, then the DPARL aggregated policy will turn to choose $W' = 1$ as the window size and then choose action a_2 as the action output. Thus, we achieve a tolerable poisoning threshold $\bar{K}_t = 1$ this time.

Illustrations of Bottleneck and Non-bottleneck States. According to our illustration in Section 9.3.2, bottleneck states are those where subpolicies vote for diverse actions but the best action is the same as previous states, and non-bottleneck states are those where subpolicies mostly vote for the same action. As we can see, s_0 to s_6 are non-bottleneck states since the subpolicies almost unanimously vote for one action. In contrast, s_7 is a bottleneck state.

From the perspectives of different *aggregation protocols*, we observe that for the bottleneck state s_7, both TPARL and DPARL exploit temporal aggregation to boost the certified robust poisoning threshold (from 0 to 2 and from 0 to 1 respectively), thus demonstrating the effectiveness of TPARL and DPARL on improving certified robustness. On the other hand, for non-bottleneck states s_0 to s_6, we can easily see that different aggregation protocols do not differ much in terms of provided certified robustness levels.

Explanations for Bottleneck and Non-bottleneck States. We provide additional discussions regarding *why bottleneck states are vulnerable* and *why our temporal aggregation strategy can effectively alleviate the problem.* We first explain the existence of the bottleneck states. Given the property of the bottleneck states that there is high disagreement among different subpolicies on such states, they may lie close to the decision boundary. This may be a result of poisoning, or simply due to the intrinsic difficulty of the state. On the other hand, such states naturally exist in the rollouts (like natural adversarial examples [147]), and may induce high instability of the rollouts during testing. We next discuss additional intuitions for our temporal aggregation. Essentially, temporal aggregation effectively leverages the adjacent non-bottleneck states to rectify the decisions at the bottleneck states, based on the assumption of temporal continuity which has been revealed and utilized in several previous works [208, 379, 433].

9.4 CERTIFICATION WITH COPA PROTOCOLS

In this section, we present the certification method for COPA. Table 9.2 presents an overview of our theoretical results supporting the certification method: For each proposed aggregation protocol and certification criterion, we provide the corresponding certification method and core theorems, and we also provide the tightness analysis for each certification.

Table 9.2: Overview of theoretical results in Section 9.3. "Certification" columns entail our certification theorems. "Analysis" columns entail the analyses of our certification bounds, where "tight" means our certification is theoretically tight, "NP-complete" means the tight certification problem is NP-complete, and "open" means the tight certification problem is still open. Theorem G.1 is in Appendix G.2.6.

Certification Criteria	Proposed Aggregation Protocol					
	PARL (π_{P}, Definition 9.4)		TPARL (π_{T}, Definition 9.5)		DPARL (π_{D}, Definition 9.6)	
	Certification	Analysis	Certification	Analysis	Certification	Analysis
Per-state Action	Theorem 9.1	tight (Proposition 9.1)	Theorem 9.2	tight (Proposition 9.2)	Theorem 9.3	open
Cumulative Reward	Theorem 9.4	tight (Theorem G.1)	Theorem 9.5	NP-complete (Theorem 9.6)	Theorem 9.7	open

9.4.1 Certification of Per-state Action Stability

In this section, we present our robustness certification theorems and methods for *per-state action*. For each of the aggregation protocols (PARL, TPARL, and DPARL), at each time step t, we will compute a valid *tolerable poisoning threshold* $\bar{K}$ as defined in Definition 9.1, such that the chosen action at step t does not change as long as the poisoning size $K \leqslant \bar{K}$.

Certification for PARL. We certify the robustness of PARL following Theorem 9.1.

Theorem 9.1. Let D be the clean training dataset; let $\pi_i = \mathcal{M}_0(D_i), 0 \leqslant i \leqslant u-1$ be the learned subpolicies according to Section 9.3.1 from which we define n_a (see Section 9.2); and let π_{P} be the Per-State Partition Aggregation policy: $\pi_{\mathrm{P}} = \mathcal{M}(D)$ where $\mathcal{M}$ abstracts the whole training-aggregation process. $\widetilde{D}$ is a poisoned dataset and $\widetilde{\pi_{\mathrm{P}}}$ is the poisoned policy: $\widetilde{\pi_{\mathrm{P}}} = \mathcal{M}(\widetilde{D})$.

For a given state s_t encountered at time step t during test time, let $a := \pi_{\mathrm{P}}(s_t)$, then

$$\bar{K}_t = \left\lfloor \frac{n_a(s_t) - \max_{a' \neq a} \left(n_{a'}(s_t) + \mathbb{I}[a' < a] \right)}{2} \right\rfloor. \tag{9.6}$$

Remark 9.1. The theorem provides a valid per-state action certification for PARL, since according to Definition 9.1, as long as the poisoning size is smaller or equal to $\bar{K}_t$, i.e., $|D \ominus \widetilde{D}| \leqslant \bar{K}_t$, the action of the poisoned policy is the same as the clean policy: $\widetilde{\pi_{\mathrm{P}}}(s_t) = \pi_{\mathrm{P}}(s_t) = a$. To compute $\bar{K}_t$, according to Equation (9.6), we rely on the aggregated action count n_a for state s_t from subpolicies. The $a' < a$ is the "smaller-than" operator introduced following Definition 9.4.

Proof of Theorem 9.1. Suppose the poisoning size is within K, then after poisoning, the aggregated action count $\widetilde{n}_a(s_t) \in [n_a(s_t) - K, n_a(s_t) + K]$ where $n_a(s_t)$ is such count before poisoning. Thus, to ensure the chosen action does not change, a necessary condition is that $\widetilde{n}_a(s_t) \geqslant \widetilde{n}_{a'}(s_t) + \mathbb{I}[a' < a]$ for any other $a' < a$. Solving K yields Equation (9.6). QED.

We use the theorem to compute the per-state action certification for each time step t along the trajectory, and the detailed algorithm is in Algorithm G.2 (Appendix G.1). Furthermore, we prove that this certification is theoretically tight as the following proposition shows. The proof is in Appendix G.2.1.

Proposition 9.1. Under the same condition as Theorem 9.1, for any time step t, there exists an RL learning algorithm $\mathcal{M}_0$, and a poisoned dataset $\widetilde{D}$, such that $|D \ominus \widetilde{D}| = \bar{K}_t + 1$, and $\widetilde{\pi_{\mathrm{P}}}(s_t) \neq \pi_{\mathrm{P}}(s_t)$.

Certification for TPARL. We certify the robustness of TPARL following Theorem 9.2.

Theorem 9.2. Let D be the clean training dataset; let $\pi_i = \mathcal{M}_0(D_i), 0 \leqslant i \leqslant u-1$ be the learned subpolicies according to Section 9.3.1 from which we define n_a (Section 9.2); and let π_{T} be the Temporal Partition Aggregation policy: $\pi_{\mathrm{T}} = \mathcal{M}(D)$ where $\mathcal{M}$ abstracts the whole training-aggregation process. $\widetilde{D}$ is a poisoned dataset and $\widetilde{\pi_{\mathrm{T}}}$ is the poisoned policy: $\widetilde{\pi_{\mathrm{T}}} = \mathcal{M}(\widetilde{D})$.

For a given state s_t encountered at time step t during test time, let $a := \pi_{\mathrm{T}}(s_{\max\{t-W+1,0\}:t})$, then at time step t the tolerable poisoning threshold (see Definition 9.1)

$$\bar{K}_t = \min_{a' \neq a, a' \in \mathcal{A}} \max \left\{ p \,\middle|\, \sum_{i=1}^{p} h_{a,a'}^{(i)} \leqslant \delta_{a,a'} \right\} \tag{9.7}$$

where $\{h_{a,a'}^{(i)}\}_{i=1}^{u}$ is a nonincreasing permutation of

$$\left\{ \sum_{j=0}^{\min\{W-1,t\}} \mathbb{I}_{i,a}(s_{t-j}) + \min\{W, t+1\} - \sum_{j=0}^{\min\{W-1,t\}} \mathbb{I}_{i,a'}(s_{t-j}) \right\}_{i=0}^{u-1} =: \{h_{i,a,a'}\}_{i=0}^{u-1}, \tag{9.8}$$

and $\delta_{a,a'} := n_a(s_{\max\{t-W+1,0\}:t}) - (n_{a'}(s_{\max\{t-W+1,0\}:t}) + \mathbb{I}[a' < a])$. Here, $\mathbb{I}_{i,a}(s) = \mathbb{I}[\pi_i(s) = a]$ (Section 9.2), and W is the window size.

Remark 9.2. We defer the detailed proof to Appendix G.2.2. The theorem provides a per-state action certification for TPARL. The detailed algorithm is in Algorithm G.3 (Appendix G.1.2). The certification time complexity per state is $O(|\mathcal{A}|u(W + \log u))$ and can be further optimized to $O(|\mathcal{A}|u \log u)$ with proper prefix sum caching across time steps. We prove the certification for TPARL is theoretically tight in Proposition 9.2 (proof in Appendix G.2.3). We also prove that directly extending Theorem 9.1 for TPARL (Corollary G.1) is loose in Appendix G.2.4.

Proposition 9.2. Under the same condition as Theorem 9.2, for any time step t, there exists an RL learning algorithm $\mathcal{M}_0$, and a poisoned dataset $\widetilde{D}$, such that $|D \ominus \widetilde{D}| = \bar{K}_t + 1$, and $\widetilde{\pi_{\mathrm{T}}}(s_{\max\{t-W+1,0\}:t}) \neq \pi_{\mathrm{T}}(s_{\max\{t-W+1,0\}:t})$.

Certification for DPARL. Theorem 9.3 provides certification for DPARL.

Theorem 9.3. Let D be the clean training dataset; let $\pi_i = \mathcal{M}_0(D_i), 0 \leqslant i \leqslant u-1$ be the learned subpolicies according to Section 9.3.1 from which we define n_a (see Section 9.2); and let π_{D} be the Dynamic Temporal Partition Aggregation: $\pi_{\mathrm{D}} = \mathcal{M}(D)$ where $\mathcal{M}$ abstracts the whole training-aggregation process. $\widetilde{D}$ is a poisoned dataset and $\widetilde{\pi_{\mathrm{D}}}$ is the poisoned policy: $\widetilde{\pi_{\mathrm{D}}} = \mathcal{M}(\widetilde{D})$.

For a given state s_t encountered at time step t during test time, let $a := \pi_{\mathrm{D}}(s_{\max\{t-W_{\max}+1,0\}:t})$ and W' be the chosen time window (according to Equation (9.4)), then tolerable poisoning threshold

$$\bar{K}_t^D = \min\left\{\bar{K}_t, \min_{1\leqslant W^*\leqslant \min\{W_{\max},t+1\}, W^*\neq W', a'\neq a, a''\neq a} L_{a',a''}^{W^*,W'}\right\} \tag{9.9}$$

where $\bar{K}_t$ is defined by Equation (9.7) with W as W' and $L_{a,a''}^{W^*,W'}$ defined by the below Definition 9.7.

Definition 9.7 (L in Theorem 9.3)**.** Under the same condition as Theorem 9.3, for given W^*, W', a, a', a'', we let $a^{\#} := \arg\max_{a_0\neq a', a_0\in\mathcal{A}} n_{a_0}(s_{t-W^*+1:t})$, then

$$L_{a',a''}^{W^*,W'} := \max\left\{p \,\middle|\, \sum_{i=1}^{p} g^{(i)} + W'(n_{a'}^{W^*} - n_{a^{\#}}^{W^*}) - W^*(n_a^{W'} - n_{a''}^{W'}) - \mathbb{I}[a' > a] < 0\right\} \tag{9.10}$$

where n_a^w is a shorthand of $n_a(s_{t-w+1:t})$ and $\{g^{(i)}\}_{i=1}^u$ is a nonincreasing permutation of $\{g_i\}_{i=0}^{u-1}$. Each g_i is defined by

$$g_i := \sum_{w=0}^{\max\{W^*,W'\}} \max_{a_0\in\mathcal{A}} \sigma^w(a_0) - \sigma^w(\pi_i(s_{t-w})), \text{ where} \tag{9.11}$$

$\sigma^w(a_0) := W'\mathbb{I}[a_0 = a', w \leqslant W^*] - W'\mathbb{I}[a_0 = a^{\#}, w \leqslant W^*] - W^*\mathbb{I}[a_0 = a, w \leqslant W'] + W^*\mathbb{I}[a_0 = a'', w \leqslant W']$.

Proof sketch. A successful poisoning attack should change the chosen action from a to an another a'. If after attack, the chosen window size is still W', the poisoning size should be at least larger than $\bar{K}_t$ according to Theorem 9.2. If the chosen window size is not W', we find out that the poisoning size is at least $\min_{a''\neq a} L_{a',a''}^{W^*,W'} + 1$ from a greedy-based analysis. Formal proof is in Appendix G.2.5. QED.

Remark 9.3. The theorem provides a valid per-state action certification for DPARL policy. The detailed algorithm is in Algorithm G.4 (Appendix G.1.2). The certification time complexity per state is $O(W_{\max}^2|\mathcal{A}|^2u + W_{\max}|\mathcal{A}|^2u\log u)$, which in practice adds similar overhead compared with TPARL certification. Unlike certification for PARL and TPARL, the certification given by Theorem 9.3 is not theoretically tight. An interesting future work would be providing a tighter per-state action certification for DPARL.

9.4.2 Certification of Cumulative Reward Bound

In this section, we present our robustness certification for *cumulative reward bound*. We assume the deterministic RL environment throughout the cumulative reward certification for convenience, i.e., the transition function is $P : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ and the initial state is a fixed $s_0 \in \mathcal{S}$. The certification goal, as listed in Definition 9.2, is to obtain a lower bound of cumulative reward under poisoning attacks, given bounded poisoning size K. The cumulative reward certification is based on a novel adaptive search algorithm COPA-LoRe inspired from [408]; we tailor the algorithm to certify against poisoning attacks. We defer detailed discussions and complexity analysis to Appendix G.1.3.

COPA-LoRe Algorithm Description. Algorithm G.5 in Appendix G.1.3 shows the pseudocode. The method starts from the base case: when the poisoning threshold $K_{\text{cur}} = 0$, the lower bound of cumulative reward $\underline{J}_{K_{\text{cur}}}$ is exactly the reward without poisoning. The method then gradually increases the poisoning threshold K_{cur}, by finding the immediate larger $K' > K_{\text{cur}}$ that can expand the possible action set along the trajectory. With the increase of $K_{\text{cur}} \leftarrow K'$, the attack may cause the poisoned policy $\tilde{\pi}$ to take different actions at some states, thus resulting in new trajectories. We need to figure out *a set of all possible actions* to exhaustively traverse all possible trajectories. We will introduce theorems to compute this set of possible actions. With this set, the method effectively explores these new trajectories by formulating them as expanded branches of a trajectory tree. Once all new trajectories are explored, the method examines all leaf nodes of the tree and figures out the minimum reward among them, which is the new lower bound of cumulative reward $\underline{J}_{K'}$ under new poisoning size K'. We then repeat this process of increasing poisoning size from K' and expanding with new trajectories until we reach a predefined threshold for poisoning size K.

Definition 9.8 (Possible Action Set)**.** Given previous states $s_{0:t}$, the subpolicies $\{\pi_i\}_{i=0}^{u-1}$, the aggregation protocol (PARL, TPARL, or DPARL), and the poisoning size K, the *possible*

action set A at step t is a subset of action space: $A \subseteq \mathcal{A}$, such that for any poisoned policy $\widetilde{\pi}$, as long as the poisoning size is within K, the chosen action at step t will always be in A, i.e., $a_t = \widetilde{\pi}(s_{0:t}) \in A$.

Possible Action Set for PARL. The following theorem gives the possible action set for PARL.

Theorem 9.4 (Tight PARL Action Set)**.** Under the condition of Definition 9.8, suppose the aggregation protocol is PARL as defined in Definition 9.4, then the *possible action set* at step t

$$A^T(K) = \left\{ a \in \mathcal{A} \,\middle|\, \sum_{a' \in \mathcal{A}} \max\{n_{a'}(s_t) - n_a(s_t) - K + \mathbb{I}[a' < a], 0\} \leqslant K \right\}. \tag{9.12}$$

We defer the proof to Appendix G.2.6. Furthermore, in Appendix G.2.6 we show that: 1) The theorem gives theoretically tight possible action set; 2) In contrast, directly extending PARL's per-state certification gives loose certification.

Possible Action Set for TPARL. The following theorem gives the possible action set for TPARL.

Theorem 9.5. Under the condition of Definition 9.8, suppose the aggregation protocol is TPARL as defined in Definition 9.5, then the *possible action set* at step t

$$A(K) = \left\{ a \in \mathcal{A} \,\middle|\, \sum_{i=1}^{K} h^{(i)}_{a',a} > \delta_{a',a}, \forall a' \neq a \right\}, \tag{9.13}$$

where $h^{(i)}_{a',a}$ and $\delta_{a',a}$ follow the definition in Theorem 9.2.

We defer the proof to Appendix G.2.7. The possible action set here is no longer theoretically tight. Indeed, the problem of computing a possible action set with minimum cardinality for TPARL is NP-complete as we shown in the following theorem (proved in Appendix G.2.8), where we reduce computing theoretically tight possible action set to the set cover problem [179]. This result can be viewed as the hardness of targeted attack. In other words, the optimal *untargeted* attack on TPARL can be found in polynomial time, while the optimal *targeted* attack on TPARL is NP-complete, which indicates the robustness property of proposed TPARL.

Theorem 9.6. Under the condition of Definition 9.8, suppose we use TPARL (Definition 9.5) as the aggregation protocol, then computing a possible action set $A(K)$ such that any possible action set S satisfies $|A(K)| \leqslant |S|$ is NP-complete.

Possible Action Set for DPARL. The following theorem gives the possible action set for DPARL.

Theorem 9.7. Under the condition of Definition 9.8, suppose the aggregation protocol is TPARL as defined in Definition 9.6, then the *possible action set* at step t

$$A(K) = \{a_t\} \cup \left\{ a' \in \mathcal{A} \,\middle|\, \min_{\substack{1 \leqslant W^* \leqslant \min\{W_{\max}, t+1\}, \\ W^* \neq W', a'' \neq a_t}} L_{a', a''}^{W^*, W'} \leqslant K \right\} \cup \left\{ a \in \mathcal{A} \,\middle|\, \sum_{i=1}^{K} h_{a',a}^{(i)} > \delta_{a',a}, \forall a' \neq a \right\} \tag{9.14}$$

where $a_t = \pi_{\mathrm{D}}(s_{\max\{t-W_{\max}+1,0\}:t})$ is the clean policy's chosen action, W' is defined by Equation (9.4), $L_{a',a''}^{W^*,W'}$ is defined by Definition 9.7 with a being replaced by a_t, and $h_{a',a}^{(i)}, \delta_{a',a}$ is defined in Theorem 9.2 with W replaced by W'.

We prove the theorem in Appendix G.2.8. As summarized in Table 9.2, we further prove the tightness or hardness of certification for PARL and TPARL, while for DPARL it is an interesting open problem on whether theoretical tight certification is possible in polynomial time.

9.5 EXPERIMENTS

In this section, we present the evaluation for our COPA framework, specifically, the aggregation protocols (Section 9.3.2) and the certification methods under different certification criteria (Sections 9.4.1 and 9.4.2). We defer the description of the offline RL algorithms (DQN [265], QR-DQN [88], and C51 [31]) used for training the subpolicies to Appendix G.3.1, and the concrete experimental procedures to Appendix G.3.2. As a summary, we obtain similar conclusions from per-state action certification and reward certification: 1) QR-DQN and C51 are oftentimes more certifiably robust than DQN; 2) temporal aggregation (TPARL and DPARL) achieves higher certification for environments satisfying temporal continuity, e.g., Freeway; 3) larger partition number improves the certified robustness; 4) Freeway is the most stable and robust environment among the three. Full experimental results and more discussions are deferred to the full version of this chapter [409].

9.5.1 Evaluation of Robustness Certification for Per-state Action Stability

We provide the robustness certification for per-state action stability based on Section 9.4.1.

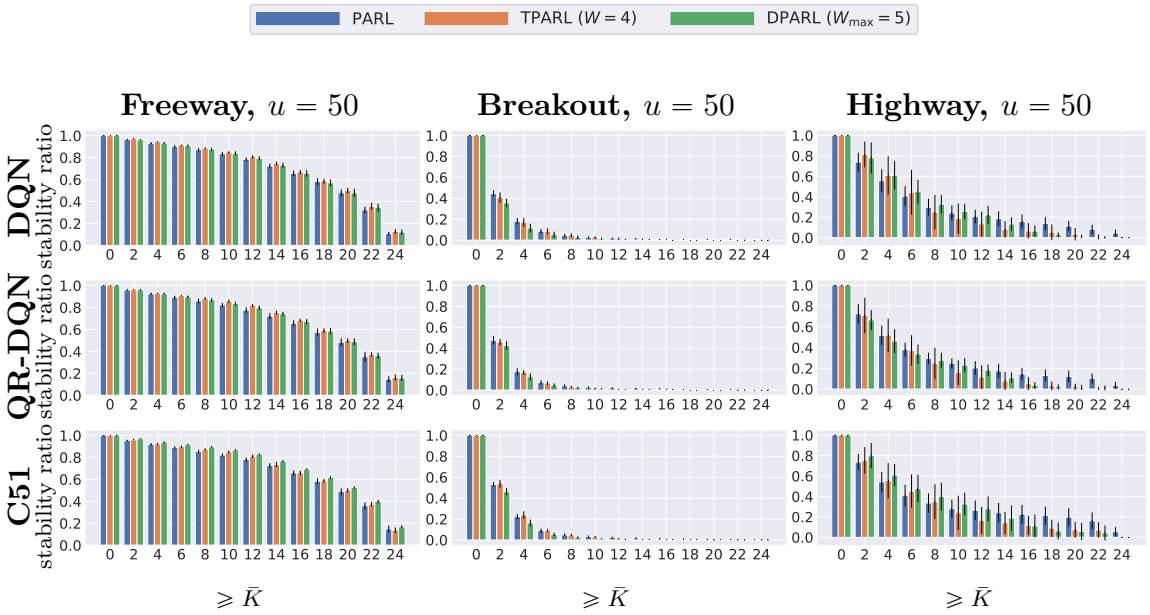


Figure 9.1: **Robustness certification for per-state action stability.** We plot the cumulative histogram of the *tolerable poisoning size* $\bar{K}$ for all time steps. We provide the certification for different aggregation protocols (PARL, TPARL, DPARL) on three environments and #partitions $u = 50$. The results are averaged over 20 runs with the vertical bar on top denoting the standard deviation.

Experimental Setup and Metrics. We evaluate the aggregated policies π_{P}, π_{T}, and π_{D} following Section 9.3.2. Basically, in each run, we run one trajectory (of maximum length H) using the derived policy, and compute $\bar{K}_t$ at each time step t. Given $\{\bar{K}_t\}_{t=0}^{H-1}$, we obtain a cumulative histogram—for each threshold $\bar{K}$, we count the time steps that achieve a threshold no smaller than it and then normalize, i.e., $\sum_{t=0}^{H-1} \mathbb{I}[\bar{K}_t \geqslant \bar{K}]/H$. We call this quantity *stability ratio* since it reflects the per-state action stability w.r.t. given poisoning thresholds. We also compute an *average tolerable poisoning thresholds* for a trajectory, defined as $\sum_{t=0}^{H-1} \bar{K}_t/H$. More details are deferred to Appendix G.3.2.

Evaluation Results. We present the comparison of per-state action certification for different RL methods and certification methods in Figure 9.1. We plot partial poisoning thresholds on the x-axes here, and omit full results in [409], where we also report the average tolerable poisoning thresholds. We additionally report benign empirical reward and the comparisons with standard training as well as more analytical statistics in [409].

The cumulative histograms in Figure 9.1 can be compared in different levels. Basically, we compare the *stability ratio* at each tolerable poisoning thresholds $\bar{K}$—*higher ratio at larger poisoning size indicates stronger certified robustness*. On the **RL algorithm** level, QR-DQN

and C51 consistently outperform the baseline DQN, and C51 has a substantial advantage particularly in Highway. On the **aggregation protocol** level, we observe different behaviors in different environments. On Freeway, methods with temporal aggregation (TPARL and DPARL) achieve higher robustness, and DPARL achieves the highest certified robustness in most cases; while on Breakout and Highway, the single-step aggregation PARL is oftentimes better. This difference is due to the different properties of environments. Our temporal aggregation is developed based on the assumption of consistent action selection in adjacent time steps. This assumption is true in Freeway while violated in Breakout and Highway. On the **partition number** level, a larger partition number generally allows larger tolerable poisoning thresholds as shown in [409]. Finally, on the **RL environment** level, Freeway achieves much higher certified robustness for per-state action stability than Highway, followed by Breakout, implying that Freeway is an environment that accommodates more stable and robust policies.

9.5.2 Evaluation of Robustness Certification for Cumulative Reward Bound

We provide the robustness certification for cumulative reward bound according to Section 9.4.2.

Experimental Setup and Metrics. We evaluate the aggregated policies π_{P}, π_{T}, and π_{D} following Theorem 9.4, Theorem 9.5 and Theorem 9.7. We compute the lower bounds of the cumulative reward $\underline{J}_K$ w.r.t. the poisoning size K using the COPA-LoRe algorithm introduced in Section 9.4.2. We provide details of the evaluated trajectory length along with the rationales in Appendix G.3.2.

Evaluation Results. We present the comparison of reward certification for different RL algorithms and certification methods in Figure 9.2. Essentially, at each poisoning size K, we compare the lower bound of cumulative reward achieved by different RL algorithms and certification methods—*higher value of the lower bound implies stronger certified robustness*. On the **RL algorithm** level, QR-DQN and C51 almost invariably outperform the baseline DQN algorithm. On the **aggregation protocol** level, methods with temporal aggregation consistently surpass the single-step aggregation PARL on Freeway but not the other two, as analyzed in Section 9.5.1. In addition, we note that DPARL is sometimes not as robust as TPARL. We hypothesize two reasons: 1) the dynamic mechanism is more susceptible to the attack, e.g., the selected optimal window size is prone to be manipulated; 2) the lower bound is looser for DPARL given the difficulty of computing the possible action set in DPARL

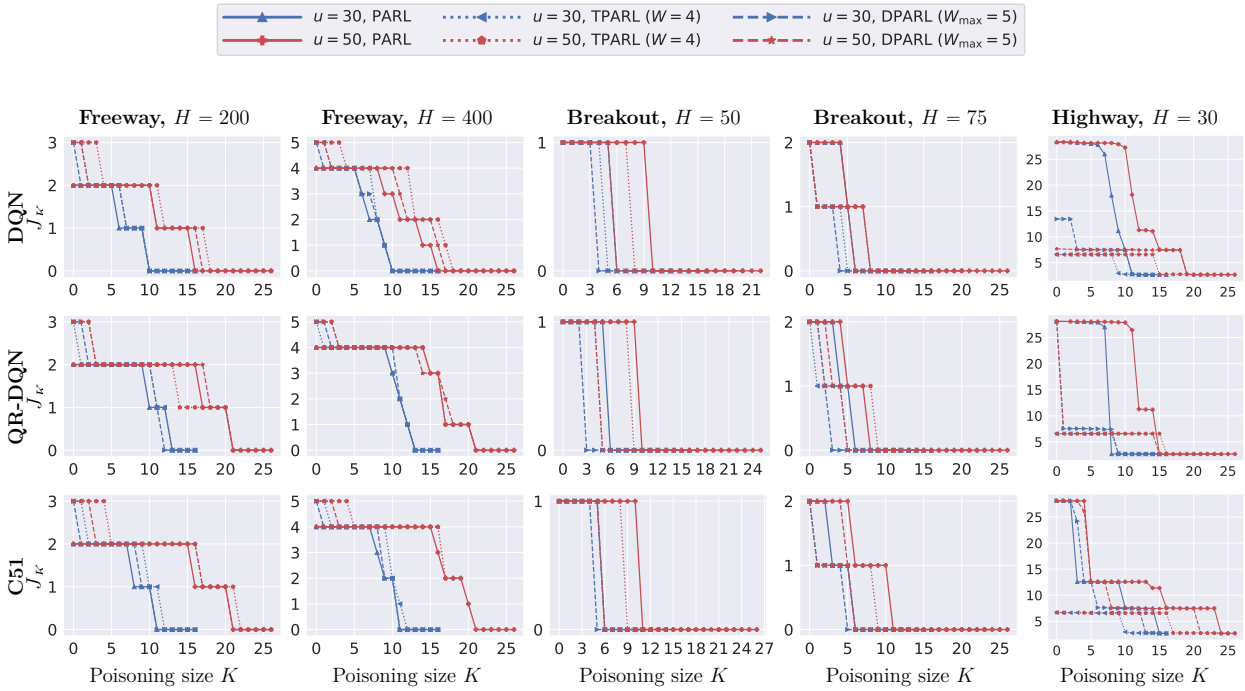


Figure 9.2: **Robustness certification for cumulative reward.** We plot the *lower bound of cumulative reward bound* $\underline{J}_K$ w.r.t. *poisoning size* K under three aggregation protocols (PARL, TPARL ($W = 4$), DPARL ($W_{\max} = 5$)) with two #partitions u, evaluated on three environments with different horizon lengths H.

(discussed in Theorem 9.7). On the **partition number** level, a larger partition number ($u = 50$) demonstrates higher robustness. On the **horizon length** level, the robustness ranking of different policies is similar under different horizon lengths with slight differences, corresponding to the property of finite-horizon RL. On the **RL environment** level, Freeway can tolerate a larger poisoning size than Breakout and Highway. More results and discussions are in [409].

9.6 SUMMARY

In this chapter, we proposed **COPA**, the first framework for certifying robust policies for offline RL against poisoning attacks. COPA includes three policy aggregation protocols. For each aggregation protocol, COPA provides a sound certification for both per-state action stability and cumulative reward bound. Experimental evaluations on different environments and different offline RL training algorithms show the effectiveness of our robustness certification in a wide range of scenarios.

CHAPTER 10: DISCUSSION FOR PART III

In this part, we discussed certified approaches for generic robustness in terms of two aspects: (1) certification based on transformation-specific smoothing to certify DL system's robustness against real-world semantic transformations, for both image models — TSS in Chapter 7 and point cloud models — TPC in Appendix A; (2) certification for reinforcement learning involving multiple rounds of interactions with the environment, against both observation perturbations — CROP in Chapter 8 and data poisoning attacks — COPA in Chapter 9.

These two aspects of extension open a wide range of future directions.

For certification based on transformation-specific smoothing, could such certification be extended beyond the vision domain, e.g., the language domain? Is there a canonical way to dig out the domain-specific transformation and synthesis certification strategy, instead of inspecting transformations one by one? Is there any difference between certifying against malicious attacks and certifying against natural distortions, in other words, intuitively, under natural distortions, DL systems should have better robustness compared with under malicious attacks, and could this be reflected as tighter robustness bounds? For the application side, our ongoing effort is to ship these robustness certification approaches into deployed DL systems, e.g., aircraft control systems and autonomous driving systems, to bring end-to-end safety guarantees. Along this line, this part can be viewed as a "toolbox" and the author optimistically believes that there are few technical challenges in applying the approaches in this thesis in deployment but only requiring engineering efforts.

For certification methods for reinforcement learning, a natural question is what connections and differences are between CROP and COPA, and whether they can be combined. First, both CROP and COPA are based on two ideas: (1) smoothing and aggregation; (2) tree search based exploration. In CROP, since we defend against observational perturbations, we apply smoothing on the input domain to cancel out the perturbations and hence aggregate predictions of noised inputs; in COPA, since we defend against training time data poisoning, we apply smoothing on the input data in the form of partitioning (which can be viewed as using a 0-1 mask on training data to mask out poisoning effects) and hence aggregate predictions from different training sets. This methodology of smoothing and aggregation is generic and can be used to defend future threats. On the other hand, since smoothing and aggregation in CROP and COPA is in orthogonal domains, they can be directly combined together to compose a joint defense against both threats. The second idea, tree-search-based exploration, is applied in CROP and COPA in almost the same way —

both approaches leverage stepwise certification to generate possible action set and maintain a tree structure to certify a lower bound of cumulative reward. Actually, this tree-search-based algorithm can also be directly combined with other robustness certification approaches beyond randomized smoothing. For example, for small-scale problems, we can deploy this algorithm in conjunction with white-box certification such as GCP-CROWN in Chapter 3. Moreover, when the transition function is known or can be precisely approximated, we can maintain action set and state set in more fine-grained structures, e.g., set of cutting planes, rather than a discrete trajectory tree, as an extension. We leave this promising direction as the future work. Another question would be: COPA applies temporal aggregation to boost the voting and hence the certified robustness. Can this idea be applied in other applications to defend under other threat models? The author believes that temporal aggregation can also improve robustness under other threat models, e.g., against observational perturbations. However, there are some practical challenges. For example, the temporal aggregation works in COPA because the dataset partitions are fixed prior to inference, so the attacker's ability is localized. In contrast, for free input perturbations, the attacker can change its perturbation per step, to achieve worst-case perturbations at each step. In this case, the temporal aggregation loses its advantage. In contrast, if the attacker is constrained to applying persistent perturbations, we believe a tighter certification can be derived with a temporal aggregation strategy similar to COPA.

As a concluding remark, the author believes that extending robustness certification beyond ℓ_p-bounded perturbations like the efforts in this thesis would still be an important direction toward practical and certifiably trustworthy deep learning. Other extensions of certified approaches in the literature can be found in Section 1.3.3 and [225].

Part IV

Certification of Trustworthy Properties beyond Robustness

In previous parts, though certified approaches are proposed and extended, they are all focused on certified robustness. The differences among these certified approaches are mainly in the applied domains, the perturbed elements, and the form of perturbations, while the certification goals are largely similar — maintaining good performance, e.g., yielding correct predictions. In this part, we propose certified approaches for two more novel trustworthy properties: distributional fairness and numerical reliability. Fairness is of core interest in AI ethics. In Chapter 11, we will see how robustness certification approaches can be extended to certify under this important notion. Numerical reliability is a lasting topic in programming languages and software engineering. DL systems can be viewed as numerical programs. Taking this perspective, we extend certified approaches to detect numerical reliability problems and assure numerical reliability for DL systems.

CHAPTER 11: FAIRNESS: CERTFAIR

As machine learning (ML) has become ubiquitous [25, 35, 73, 90, 139, 200], fairness of ML has attracted a lot of attention from different perspectives. For instance, some automated hiring systems are biased towards males due to gender imbalanced training data [14]. Different approaches have been proposed to improve ML fairness, such as regularized training [107, 182, 231, 250], disentanglement [82, 239, 322], duality [348], low-rank matrix factorization [282], and distribution alignment [23, 242, 463].

In addition to existing approaches that *evaluate* fairness, it is important and challenging to provide *certification* for ML fairness. Recent studies have explored the certified fair *representation* of ML [23, 294, 313]. However, there lacks certified fairness on the *predictions* of an end-to-end ML model trained on an arbitrary data distribution. In addition, current fairness literature mainly focuses on training an ML model on a potentially (im)balanced distribution and evaluate its performance in a target domain measured by existing statistical fairness definitions [128, 172]. Since in practice these selected target domains can encode certain forms of unfairness of their own (e.g., sampling bias), the evaluation would be more informative if we can evaluate and certify fairness of an ML model on an *objective* distribution. Taking these factors into account, in this work, we aim to provide the first definition of *certified fairness* given an ML model and a training distribution by bounding its end-to-end performance on an objective, *fairness constrained distribution*. In particular, we define *certified fairness* as the worst-case upper bound of the ML prediction loss on a fairness constrained test distribution $\mathcal{Q}$, which is within a bounded distance to the training distribution $\mathcal{P}$. For example, for an ML model of crime rate prediction, we can define the model performance as the expected loss within a specific age group. Suppose the model is deployed in a fair environment that does not deviate too much from the training, our fairness certificate can guarantee that the loss of crime rate prediction for a particular age group is upper bounded, which is an indicator of model's fairness.

We mainly focus on the base rate condition as the fairness constraint for $\mathcal{Q}$. We prove that our certified fairness based on a base rate constrained distribution will imply other fairness metrics, such as demographic parity (DP) and equalized odds (EO). Moreover, our framework is flexible to integrate other fairness constraints into $\mathcal{Q}$. We consider two scenarios: (1) *sensitive shifting* where only the joint distribution of sensitive attribute and label can be changed when optimizing $\mathcal{Q}$; and (2) *general shifting* where everything including the conditioned distribution of non-sensitive attributes can be changed. We then propose an effective *fairness certification framework*, CertFair, to compute the certificate.

In CertFair, we first formulate the problem as constrained optimization, where the fairness constrained distribution is encoded by base rate constraints. Our key technique is to decompose both training and the fairness constrained test distributions to several subpopulations based on sensitive attributes and target labels, which can be used to encode the base rate constraints. With such a decomposition, in sensitive shifting, we can decompose the distance constraint to subpopulation ratio constraints and prove the transformed low-dimensional optimization problem is convex and thus efficiently solvable. In general shifting case, we propose to solve it based on divide and conquer: we first partition the feasible space into different subpopulations, then optimize the density (ratio) of each subpopulation, apply relaxation on each subpopulation as a sub-problem, and finally prove the convexity of the sub-problems with respect to other low-dimensional variables. Our framework is applicable for any black-box ML models and any distributional shifts bounded by the Hellinger distance, which is a type of f-divergence studied in the literature [33, 100, 101, 201, 399].

To demonstrate the effectiveness and tightness of CertFair, we evaluate our fairness bounds on *six* real-world fairness related datasets [11, 14, 161, 403]. We show that our certificate is tight under different scenarios. In addition, we verify that our framework is flexible to integrate additional constraints on $\mathcal{Q}$ and evaluate the certified fairness with additional non-skewness constraints, with which our fairness certificate is tighter. Finally, as the first work on certifying fairness of an end-to-end ML model, we adapt existing distributional robustness bound [347] for comparison to provide more intuition. Note that directly integrating the fairness constraint to the existing distributional robustness bound is challenging, which is one of the main contributions for our framework. We show that with the fairness constraints and our effective solution, our bound is strictly tighter.

Technical Contributions. In this chapter, we take the first attempt towards formulating and computing the *certified fairness* on an end-to-end ML model, which is trained on a given distribution. We make contributions on both theoretical and empirical fronts.

1. We formulate the *certified fairness* of an end-to-end ML model trained on a given distribution $\mathcal{P}$ as the worst-case upper bound of its prediction loss on a fairness constrained distribution $\mathcal{Q}$, which is within bounded distributional distance with $\mathcal{P}$.

2. We propose an effective fairness certification framework, CertFair, that simulates the problem as constrained optimization and solve it by decomposing the training and fairness constrained test distributions into subpopulations and proving the convexity of each sub-problem to solve it.

3. We evaluate our certified fairness on six real-world datasets to show its tightness and scalability. We also show that with additional distribution constraints on $\mathcal{Q}$, our certification would be tighter.

4. We show that our bound is strictly tighter than adapted distributional robustness bound on Gaussian dataset due to the added fairness constraints and our effective optimization approach.

11.1 RELATED WORK ON MACHINE LEARNING FAIRNESS

Fairness in ML can be generally categorized into individual fairness and group fairness. Individual fairness guarantees that similar inputs should lead to similar outputs for a model and it is analyzed with optimization approaches [259, 404] and different types of relaxations [173]. Group fairness indicates to measure the *independence* between the sensitive features and model prediction, the *separation* which means that the sensitive features are statistically independent of model prediction given the target label, and the *sufficiency* which means that the sensitive features are statistically independent of the target label given the model prediction [238]. Different approaches are proposed to analyze group fairness via static analysis [375], interactive computation [325], and probabilistic approaches [5, 26, 71]. In addition, there is a line of work trying to certify the *fair representation* [23, 294, 313]. In [65], the authors have provided bounds for how group fairness transfers subject to bounded distribution shift. Our certified fairness differs from existing work from three perspectives: 1) we provide fairness certification considering the end-to-end model performance instead of the representation level, 2) we define and certify fairness based on a fairness constrained distribution which implies other fairness notions, and 3) our certified fairness can be computed for *any* black-box models trained on an arbitrary given data distribution.

11.2 CERTIFIED FAIRNESS BASED ON FAIRNESS CONSTRAINED DISTRIBUTION

In this section, we first introduce preliminaries, and then propose the definition of *certified fairness* based on a bounded fairness constrained distribution, which to the best of our knowledge is the first formal fairness certification on end-to-end model prediction. We also show that our proposed certified fairness relates to established fairness definitions in the literature.

Notations. We consider the general classification setting: we denote by $\mathcal{X}$ and $\mathcal{Y} = [C]$ the feature space and labels. $h_{\boldsymbol{\theta}} \colon \mathcal{X} \to \Delta^{|\mathcal{Y}|}$ represents a mapping function parameterized with $\boldsymbol{\theta} \in \Theta$, and $\ell \colon \Delta^{|\mathcal{Y}|} \times \mathcal{Y} \to \mathbb{R}_+$ is a non-negative loss function such as cross-entropy loss. Within feature space $\mathcal{X}$, we identify a *sensitive* or *protected attribute* $\mathcal{X}_s$ that takes a finite number of values: $\mathcal{X}_s := [S]$, i.e., for any $\boldsymbol{X} \in \mathcal{X}$, $X_s \in [S]$.

Definition 11.1 (Base Rate)**.** Given a distribution $\mathcal{P}$ supported over $\mathcal{X} \times \mathcal{Y}$, the base rate for sensitive attribute value $s \in [S]$ with respect to label $y \in [C]$ is $b^{\mathcal{P}}_{s,y} = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{P}}[Y = y \,|\, X_s = s]$.

Given the definition of base rate, we define a *fair base rate distribution* (in short as *fair distribution*).

Definition 11.2 (Fair Base Rate Distribution)**.** A distribution $\mathcal{P}$ supported over $\mathcal{X} \times \mathcal{Y}$ is a fair base rate distribution if and only if for any label $y \in [C]$, the base rate $b^{\mathcal{P}}_{s,y}$ is equal across all $s \in [S]$, i.e., $\forall i \in [S], \forall j \in [S], b^{\mathcal{P}}_{i,y} = b^{\mathcal{P}}_{j,y}$.

Remark 11.1. In the literature, the concepts of fairness are usually directly defined at the model prediction level, where the criterion is whether the model prediction is fair against individual attribute changes [294, 313, 435] or fair at population level [462]. In this work, to certify the fairness of model prediction, we define a fairness constrained distribution on which we will certify the model prediction (e.g., bound the prediction error), rather than relying on the empirical fairness evaluation. In particular, we first define the fairness constrained distribution through the lens of base rate parity, i.e., the probability of being any class should be independent of sensitive attribute values, and then define the certified fairness of a given model based on its performance on the fairness constrained distribution as we will show next.

The choice of focusing on fair base rate may look restrictive but its definition aligns very well with the celebrated fairness definition Demographic Parity [436], which promotes that $\Pr[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1 | X_s = i] = \Pr[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1 | X_s = j]$. In this case, the prediction performance of $h_{\boldsymbol{\theta}}$ on $\mathcal{Q}$ with fair base rate will relate directly to $\Pr[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1 | X_s = i]$. Secondly, under certain popular data generation process, the base rate sufficiently encodes the differences in distributions and a fair base rate will imply a homogeneous (therefore equal or "fair") distribution over X, Y: consider when $\Pr(\boldsymbol{X} | Y = y, X_s = i)$ is the same across different group X_s. Then $\Pr(\boldsymbol{X}, Y | X_s = i)$ is simply a linear combination of basis distributions $\Pr(\boldsymbol{X} | Y = y, X_s = i)$, and the difference between different groups' joint distribution of X, Y is fully characterized by the difference in base rate $\Pr(Y = y | X_s)$. This assumption will greatly enable trackable analysis and is not an uncommon modeling choice in the recent discussion of fairness when distribution shifts [306, 454].

11.2.1 Certified Fairness

Now we are ready to define the fairness certification based on the optimized fairness constrained distribution. We define the certification under two data generation scenarios: *general shifting* and *sensitive shifting*. In particular, consider the data generative model $\Pr(\boldsymbol{X}_o, X_s, Y) = \Pr(Y)\Pr(X_s|Y)\Pr(\boldsymbol{X}_o|Y, X_s)$, where $\boldsymbol{X}_o$ and X_s represent the non-sensitive and sensitive features, respectively. If all three random variables on the RHS are allowed to change, we call it *general shifting*; if both $\Pr(Y)$ and $\Pr(X_s|Y)$ are allowed to change to ensure the fair base rate (Definition 11.2) while $\Pr(\boldsymbol{X}_o|Y, X_s)$ is the same across different groups, we call it *sensitive shifting*. In Section 11.3 we will introduce our certification framework for both scenarios.

Problem 1 (Certified Fairness with General Shifting)**.** Given a training distribution $\mathcal{P}$ supported on $\mathcal{X} \times \mathcal{Y}$, a model $h_{\boldsymbol{\theta}}(\cdot)$ trained on $\mathcal{P}$, and distribution distance bound $\rho > 0$, we call $\bar{\ell} \in \mathbb{R}$ a *fairness certificate* with general shifting, if $\bar{\ell}$ upper bounds

$$\max_{\mathcal{Q}} \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \quad \text{s.t.} \quad \text{dist}(\mathcal{P}, \mathcal{Q}) \leqslant \rho, \quad \mathcal{Q} \text{ is a fair distribution}, \tag{11.1}$$

where $\text{dist}(\cdot, \cdot)$ is a predetermined distribution distance metric.

In the above definition, we define the fairness certificate as the upper bound of the model's loss among all fair base rate distributions $\mathcal{Q}$ within a bounded distance from $\mathcal{P}$. Besides the bounded distance constraint $\text{dist}(\mathcal{P}, \mathcal{Q}) \leqslant \rho$, there is no other constraint between $\mathcal{P}$ and $\mathcal{Q}$ so this satisfies "*general shifting*". This bounded distance constraint, parameterized by a tunable parameter ρ, ensures that the test distribution should not be too far away from the training. In practice, the model $h_{\boldsymbol{\theta}}$ may represent a DNN whose complex analytical forms would pose challenges for solving Problem 1. As a result, as we will show in Equation (11.5) we can query some statistics of $h_{\boldsymbol{\theta}}$ trained on $\mathcal{P}$ as constraints to characterize $h_{\boldsymbol{\theta}}$, and thus compute the upper bound certificate.

The feasible region of optimization problem 1 might be empty if the distance bound ρ is too small, and thus we cannot provide fairness certification in this scenario, indicating that there is no nearby fair distribution and thus the fairness of the model trained on the highly "unfaired" distribution is generally low. In other words, if the training distribution $\mathcal{P}$ is unfair (typical case) and there is no feasible fairness constrained distribution $\mathcal{Q}$ within a small distance to $\mathcal{P}$, fairness cannot be certified.

This definition follows the intuition of typical real-world scenarios: The real-world training dataset is usually biased due to the limitation in data curation and collection processes, which causes the model to be unfair. Thus, when the trained models are evaluated on the

real-world fairness constrained test distribution or ideal fair distribution, we hope that the model does not encode the training bias which would lead to low test performance. That is to say, the model performance on fairness constrained distribution is indeed a witness of the model's intrinsic fairness.

We can further constrain that the subpopulation of $\mathcal{P}$ and $\mathcal{Q}$ parameterized by X_s and Y does not change, which results in the following "*sensitive shifting*" fairness certification.

Problem 2 (Certified Fairness with Sensitive Shifting)**.** Under the same setting as Problem 1, we call $\bar{\ell}$ a fairness certificate against sensitive shifting, if $\bar{\ell}$ upper bounds

$$\begin{aligned} &\max_{\mathcal{Q}} \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] \\ \text{s.t.}\quad &\mathrm{dist}(\mathcal{P},\mathcal{Q}) \leqslant \rho, \quad \mathcal{P}_{s,y} = \mathcal{Q}_{s,y}\, \forall s \in [S], y \in [C], \quad \mathcal{Q} \text{ is a fair distribution,} \end{aligned} \tag{11.2}$$

where $\mathcal{P}_{s,y}$ and $\mathcal{Q}_{s,y}$ are the subpopulations of $\mathcal{P}$ and $\mathcal{Q}$ on the support $\{(\boldsymbol{X},Y) : \boldsymbol{X} \in \mathcal{X}, X_s = s, Y = y\}$ respectively, and $\mathrm{dist}(\cdot,\cdot)$ is a predetermined distribution distance metric.

The definition adds an additional constraint between $\mathcal{P}$ and $\mathcal{Q}$ that each subpopulation, partitioned by the sensitive attribute X_s and label Y, does not change. This constraint corresponds to the scenario where the distribution shifting between training and test distributions only happens on the proportions of different sensitive attributes and labels, and within each subpopulation the shifting is negligible.

In addition, to model the real-world test distribution, we may further request that the test distribution $\mathcal{Q}$ is not too skewed regarding the sensitive attribute X_s by adding constraint (11.3). We will show that this constraint can also be integrated into our fairness certification framework flexibly in Section 11.4.3.

$$\forall i \in [S], \forall j \in [S], \left| \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[X_s = i] - \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[X_s = j] \right| \leqslant \Delta_S. \tag{11.3}$$

Connections to Other Fairness Measurements. Though not explicitly stated, our goal of certifying the performance on a fair distribution $\mathcal{Q}$ relates to certifying established fairness definitions in the literature. Consider the following example: Suppose Problem 2 is feasible and returns a classifier $h_{\boldsymbol{\theta}}$ that achieves certified fairness per group and per label class $\bar{l} := \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y | Y = y, X_s = i] \leqslant \epsilon$ on $\mathcal{Q}$. We will then have the following proposition:

Proposition 11.1. $h_{\boldsymbol{\theta}}$ achieves ϵ-Demographic Parity (DP) [436] and ϵ-Equalized Odds (EO) [139]:

- ϵ-DP: $\left|\Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|X_s = i] - \Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|X_s = j]\right| \leqslant \epsilon,\ \forall i, j.$
- ϵ-EO: $\left|\Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y = y, X_s = i] - \Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y = y, X_s = j]\right| \leqslant \epsilon, \forall y, i, j.$

Remark 11.2. The detailed proof is omitted to appendix H.1.1. (1) When $\epsilon = 0$, Proposition 11.1 can guarantee perfect DP and EO simultaneously. We achieve so because we evaluate with a fair distribution $\mathcal{Q}$, where "fair distribution" stands for "equalized base rate" and according to [185, Theorem 1.1, page 5] both DP and EO are achievable for this fair distribution. This observation in fact motivated us to identify the fair distribution $\mathcal{Q}$ for the evaluation since it is this fair distribution that allows the fairness measures to hold at the same time. Therefore, another way to interpret our framework is: given a model, we provide a framework that certifies worst-case "unfairness" bound in the context where perfect fairness is achievable. Such a worse-case bound serves as the gap to a perfectly fair model and could be a good indicator of the model's fairness level. (2) In practice, ϵ is not necessarily zero. Therefore, Proposition 11.1 only provides an upper lower bound of DP and EO, namely ϵ-DP and ϵ-EO, instead of absolute DP and EO. The approximate fairness guarantee renders our results more general. Meanwhile, there is a higher flexiblity in simultaneously satisfying approximate fairness metrics (for example when DP = 0, but EO = ϵ, which is plausible for a proper range of epsilon, regardless of the distribution $\mathcal{Q}$ being fair or not). But again, similar to (1), ϵ-DP and ϵ-EO can be achieved at the same time easily since the test distribution satisfies base rate parity.

The bounds in Proposition 11.1 are tight. Consider the distribution $\mathcal{Q}$ with binary classes and binary sensitive attributes (i.e., $Y, X_s \in \{0, 1\}$). When the distribution $\mathcal{Q}$ and classifier $h_{\boldsymbol{\theta}}$ satisfy the conditions that $\Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y|Y = 0, X_s = 0] = \epsilon, \Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y|Y = 0, X_s = 1] = 0$ and $\Pr_{\mathcal{Q}}[Y = 0] = 1, \Pr_{\mathcal{Q}}[Y = 1] = 0$, the bounds in Proposition 1 are tight. From $\Pr_{\mathcal{Q}}[Y = 0] = 1, \Pr_{\mathcal{Q}}[Y = 1] = 0$, we can observe that ϵ-DP is equivalent to ϵ-EO. From $\Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y|Y = 0, X_s = 0] = \epsilon, \Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y|Y = 0, X_s = 1] = 0$ and $\Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y|Y = 0, X_s = i] = \Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y = 0, X_s = i]$ for $i \in \{0, 1\}$, we know that ϵ-EO holds with tightness since $\left|\Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y = 0, X_s = 0] - \Pr_{\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y = 0, X_s = 1]\right| = \epsilon$. To this point, we show that both bounds in Proposition 1 are tight.

11.3 CERTFAIR: THE FAIRNESS CERTIFICATION FRAMEWORK

We will introduce our fairness certification framework which efficiently computes the fairness certificate defined in Section 11.2.1. We first introduce our framework for *sensitive shifting* (Problem 2) which is less complex and shows our core methodology, then *general shifting* case (Problem 1).

Our framework focuses on using the Hellinger distance to bound the distributional distance in Problems 1 and 2. The Hellinger distance $H(\mathcal{P}, \mathcal{Q})$ is defined in Definition 11.3.

Definition 11.3 (Hellinger Distance)**.** Let $\mathcal{P}$ and $\mathcal{Q}$ be distributions on $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$ that are absolutely continuous with respect to a reference measure μ with $\mathcal{P}, \mathcal{Q} \ll \mu$. The Hellinger distance between $\mathcal{P}$ and $\mathcal{Q}$ is defined as

$$H(\mathcal{P}, \mathcal{Q}) := \sqrt{\frac{1}{2} \int_{\mathcal{Z}} \left(\sqrt{p(\boldsymbol{z})} - \sqrt{q(\boldsymbol{z})} \right)^2 \mathrm{d}\mu(\boldsymbol{z})} \tag{11.4}$$

where $p = \frac{d\mathcal{P}}{d\mu}$ and $q = \frac{d\mathcal{Q}}{d\mu}$ are the Radon-Nikodym derivatives of $\mathcal{P}$ and $\mathcal{Q}$ with respect to μ, respectively. The Hellinger distance is independent of the choice of the reference measure μ.

The Hellinger distance has some nice properties, e.g., $H(\mathcal{P}, \mathcal{Q}) \in [0, 1]$, and $H(\mathcal{P}, \mathcal{Q}) = 0$ if and only if $\mathcal{P} = \mathcal{Q}$ and the maximum value of 1 is attained when $\mathcal{P}$ and $\mathcal{Q}$ have disjoint support. The Hellinger distance is a type of f-divergences which are widely studied in ML distributional robustness literature [100, 399] and in the context of distributionally robust optimization [33, 101, 201]. Also, using Hellinger distance enables our certification framework to generalize to *total variation distance (or statistic distance)* $\delta(\mathcal{P}, \mathcal{Q})$[8] directly with the connection, $H^2(\mathcal{P}, \mathcal{Q}) \leqslant \delta(\mathcal{P}, \mathcal{Q}) \leqslant \sqrt{2} H(\mathcal{P}, \mathcal{Q})$ ([351], Equation 1). We leave the extension of our framework to other distance metrics as future work.

11.3.1 Core Idea: Subpopulation Decomposition

The core idea in our framework is (finite) subpopulation decomposition. Consider a generic optimization problem for computing the loss upper bound on a constrained test distribution $\mathcal{Q}$, given training distribution $\mathcal{P}$ and trained model $h_{\boldsymbol{\theta}}(\cdot)$, we first characterize model $h_{\boldsymbol{\theta}}(\cdot)$ based on some statistics, e.g., mean and variance for loss of the model: $h_{\boldsymbol{\theta}}(\cdot)$ satisfies $e_j(\mathcal{P}, h_{\boldsymbol{\theta}}) \leqslant v_j$, $1 \leqslant j \leqslant L$. Then we characterize the properties (e.g., fair base rate) of the test distribution $\mathcal{Q}$: $g_j(\mathcal{Q}) \leqslant u_j$, $1 \leqslant j \leqslant M$. As a result, we can upper bound the loss of $h_{\boldsymbol{\theta}}(\cdot)$ on $\mathcal{Q}$ as the following optimization:

$$\max_{\mathcal{Q}, \boldsymbol{\theta}} \mathop{\mathbb{E}}_{(\boldsymbol{X}, Y) \sim \mathcal{Q}} [\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \quad \text{s.t.} \quad H(\mathcal{P}, \mathcal{Q}) \leqslant \rho, \quad e_j(\mathcal{P}, h_{\boldsymbol{\theta}}) \leqslant v_j \, \forall j \in [L], \quad g_j(\mathcal{Q}) \leqslant u_j \, \forall j \in [M]. \tag{11.5}$$

Now we decompose the space $\mathcal{Z} := \mathcal{X} \times \mathcal{Y}$ to N partitions: $\mathcal{Z} := \biguplus \mathcal{Z}_i$, where $\mathcal{Z}$ is the support of both $\mathcal{P}$ and $\mathcal{Q}$. Then, we denote $\mathcal{P}$ conditioned on $\mathcal{Z}_i$ by $\mathcal{P}_i$ and similarly $\mathcal{Q}$

[8] $\delta(\mathcal{P}, \mathcal{Q}) = \sup_{A \in \mathcal{F}} |\mathcal{P}(A) - \mathcal{Q}(A)|$ where $\mathcal{F}$ is a σ-algebra of subsets of the sample space Ω.

conditioned on $\mathcal{Z}_i$ by $\mathcal{Q}_i$. As a result, we can write $\mathcal{P} = \sum_{i\in[N]} p_i\mathcal{P}_i$ and $\mathcal{Q} = \sum_{i\in[N]} q_i\mathcal{Q}_i$. Since $\mathcal{P}$ is known, p_i's are known. In contrast, both $\mathcal{Q}_i$ and q_i's are optimizable. Our key observation is that

$$H(\mathcal{P},\mathcal{Q}) \leqslant \rho \iff 1-\rho^2 - \sum_{i=1}^{N} \sqrt{p_i q_i}(1 - H(\mathcal{P}_i,\mathcal{Q}_i)^2) \leqslant 0 \tag{11.6}$$

which leads to the following theorem.

Theorem 11.1. The following constrained optimization upper bounds Equation (11.5):

$$\max_{\mathcal{Q}_i, q_i, \rho_i, \boldsymbol{\theta}} \quad \sum_{i=1}^{N} q_i \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_i}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] \tag{11.7a}$$

$$\text{s.t.} \quad 1-\rho^2 - \sum_{i=1}^{N} \sqrt{p_i q_i}(1-\rho_i^2) \leqslant 0, \tag{11.7b}$$

$$H(\mathcal{P}_i,\mathcal{Q}_i) \leqslant \rho_i \quad \forall i \in [N], \quad \sum_{i=1}^{N} q_i = 1, \quad q_i \geqslant 0 \quad \forall i \in [N], \quad \rho_i \geqslant 0 \quad \forall i \in [N], \tag{11.7c}$$

$$e'_j(\{\mathcal{P}_i\}_{i\in[N]}, \{p_i\}_{i\in[N]}, h_{\boldsymbol{\theta}}) \leqslant v'_j \; \forall j \in [L], \quad g'_j(\{\mathcal{Q}_i\}_{i\in[N]}, \{q_i\}_{i\in[N]}) \leqslant u'_j \; \forall j \in [M], \tag{11.7d}$$

if $e_j(\mathcal{P}, h_{\boldsymbol{\theta}}) \leqslant v_j$ implies $e'_j(\{\mathcal{P}_i\}_{i\in[N]}, \{p_i\}_{i\in[N]}, h_{\boldsymbol{\theta}}) \leqslant v'_j$ for any $j \in [L]$, and $g_j(\mathcal{Q}) \leqslant u_j$ implies $g'_j(\{\mathcal{Q}_i\}_{i\in[N]}, \{q_i\}_{i\in[N]}) \leqslant u'_j$ for any $j \in [M]$.

In Problem 11.5, the challenge is to deal with the fair base rate constraint. Our core technique in Theorem 11.1 is subpopulation decomposition. At a high level, thanks to the disjoint support among different subpopulations, we get Equation (11.6). This equation gives us an equivalence relationship between distribution-level (namely, $\mathcal{P}$ and $\mathcal{Q}$) distance constraint and subpopulation-level (namely, $\mathcal{P}_i$'s and $\mathcal{Q}_i$'s) distance constraint. As a result, we can rewrite the original problem (11.5) using sub-population as decision variables as in Equation (11.7b) and then imposing the unity constraint (Equation (11.7c)) to get Theorem 11.1. We provide a detailed proof in Appendix H.1.2. Although the optimization problem (Equation (11.7)) may look more complicated then the original Equation (11.5), this optimization simplifies the challenging fair base rate constraint, allows us to upper bound each subpopulation loss $\mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_i}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)]$ individually, and hence makes the whole optimization tractable.

11.3.2 Certified Fairness with Sensitive Shifting

For the sensitive shifting case, we instantiate Theorem 11.1 and obtain the following fairness certificate.

Theorem 11.2. Given a distance bound $\rho > 0$, the following constrained optimization, which is **convex**, when feasible, provides a **tight** fairness certificate for Problem 2:

$$\max_{k_s, r_y} \sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y E_{s,y}, \text{ s.t. } \sum_{s=1}^{S} k_s = 1, \quad \sum_{y=1}^{C} r_y = 1, \quad k_s \geqslant 0 \quad \forall s \in [S], \quad r_y \geqslant 0 \quad \forall y \in [C],$$
$$1 - \rho^2 - \sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y} \leqslant 0, \tag{11.8}$$

where $E_{s,y} := \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)]$ and $p_{s,y} := \Pr_{(\boldsymbol{X},Y)\in\mathcal{P}}[X_s = s, Y = y]$ are constants.

Proof sketch. We decompose distribution $\mathcal{P}$ and $\mathcal{Q}$ to $\mathcal{P}_{s,y}$'s and $\mathcal{Q}_{s,y}$'s according to their sensitive attribute and label values. In sensitive shifting, $\Pr(\boldsymbol{X}_o|Y, X_s)$ is fixed, i.e., $\mathcal{P}_{s,y} = \mathcal{Q}_{s,y}$, which means $\mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] = E_{s,y}$ and $\rho_{s,y} = H(\mathcal{P}_{s,y}, \mathcal{Q}_{s,y}) = 0$. We plug these properties into Theorem 11.1. Then, denoting $q_{s,y}$ to $\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[X_s = s, Y = y]$, we can represent the fairness constraint in Definition 11.2 as $q_{s_0,y_0} = \left(\sum_{s=1}^{S} q_{s,y_0}\right)\left(\sum_{y=1}^{C} q_{s_0,y}\right)$ for any $s_0 \in [S]$ and $y_0 \in [C]$. Next, we parameterize $q_{s,y}$ with $k_s r_y$. Such parameterization simplifies the fairness constraint and allow us to prove the convexity of the resulting optimization. Since all the constraints are encoded equivalently, the problem formulation provides a tight certification. Detailed proof in Appendix H.1.3. QED.

As Theorem 11.2 suggests, we can exploit the expectation information

$$E_{s,y} = \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \tag{11.9}$$

and density information

$$p_{s,y} = \mathop{\Pr}_{(\boldsymbol{X},Y)\sim\mathcal{P}}[X_s = s, Y = y] \tag{11.10}$$

of each $\mathcal{P}$'s subpopulation to provide a tight fairness certificate in sensitive shifting. The convex optimization problem with $(S + C)$ variables can be efficiently solved by off-the-shelf packages.

11.3.3 Certified Fairness with General Shifting

For the general shifting case, we leverage Theorem 11.1 and the parameterization trick $q_{s,y} := k_s r_y$ used in Theorem 11.2 to reduce Problem 1 to the following constrained optimization.

Lemma 11.1. Given a distance bound $\rho > 0$, the following constrained optimization, when feasible, provides a **tight** fairness certificate for Problem 1:

$$\max_{k_s, r_y, \mathcal{Q}, \rho_{s,y}} \quad \sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \tag{11.11a}$$

$$\text{s.t.} \quad \sum_{s=1}^{S} k_s = 1, \quad \sum_{y=1}^{C} r_y = 1, \quad k_s \geqslant 0 \quad \forall s \in [S], \quad r_y \geqslant 0 \quad \forall y \in [C], \tag{11.11b}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y}(1 - \rho_{s,y}^2) \geqslant 1 - \rho^2 \tag{11.11c}$$

$$H(\mathcal{P}_{s,y}, \mathcal{Q}_{s,y}) \leqslant \rho_{s,y} \quad \forall s \in [S], y \in [C], \tag{11.11d}$$

where $p_{s,y} := \Pr_{(\boldsymbol{X},Y)\in\mathcal{P}}[X_s = s, Y = y]$ is a fixed constant. The $\mathcal{P}_{s,y}$ and $\mathcal{Q}_{s,y}$ are the subpopulations of $\mathcal{P}$ and $\mathcal{Q}$ on the support $\{(\boldsymbol{X}, Y) : \boldsymbol{X} \in \mathcal{X}, X_s = s, Y = y\}$ respectively.

Proof sketch. We show that (11.11b) ensures a parameterization of $q_{s,y} = \Pr_{(\boldsymbol{X},Y)\in\mathcal{Q}}[X_s = s, Y = y]$ that satisfies fairness constraints on $\mathcal{Q}$. Then, leveraging Theorem 11.1 we prove that the constrained optimization provides a fairness certificate. Since all the constraints are either kept or equivalently encoded, this resulting certification is *tight*. Detailed proof in Appendix H.1.4. QED.

Now the main obstacle is to solve the non-convex optimization in Problem 11.11. Here, as the first step, we upper bound the loss of $h_{\boldsymbol{\theta}}(\cdot)$ within each shifted subpopulation $\mathcal{Q}_{s,y}$, i.e., upper bound $\mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)]$ in Equation (11.11a), by [399, Theorem 2.2] (stated as Theorem H.1). Then, we apply variable transformations to make some decision variables convex. For the remaining decision variables, we observe that they are non-convex but bounded. Hence, we propose the technique of grid-based sub-problem construction. Concretely, we divide the feasible region regarding non-convex variables into small grids and consider the optimization problem in each region individually. For each sub-problem, we relax the objective by pushing the values of non-convex variables to the boundary of the current grid and then solve the convex optimization sub-problems. Concretely, the following theorem states our computable certificate for Problem 1, with detailed proof in Appendix H.1.5.

Theorem 11.3. If for any $s \in [S]$ and $y \in [Y]$,

$$H(\mathcal{P}_{s,y}, \mathcal{Q}_{s,y}) \leqslant \bar{\gamma}_{s,y} \text{ and } 0 \leqslant \sup_{(\boldsymbol{X},Y)\in\mathcal{X}\times\mathcal{Y}} \ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y) \leqslant M, \tag{11.12}$$

given a distance bound $\rho > 0$, for any region granularity $T \in \mathbb{N}_+$, the following expression provides a fairness certificate for Problem 1:

$$\bar{\ell} = \max_{\{i_s\in[T]:s\in[S]\},\{j_y\in[T]:y\in[C]\}} \mathsf{C}\left(\left\{\left[\frac{i_s-1}{T}, \frac{i_s}{T}\right]\right\}_{s=1}^{S}, \left\{\left[\frac{j_y-1}{T}, \frac{j_y}{T}\right]\right\}_{y=1}^{C}\right), \text{ where} \tag{11.13}$$

$$\mathsf{C}\left(\{[\underline{k_s}, \overline{k_s}]\}_{s=1}^{S}, \{[\underline{r_y}, \overline{r_y}]\}_{y=1}^{C}\right) = \max_{x_{s,y}} \sum_{s=1}^{S}\sum_{y=1}^{C}\Big(\overline{k_s}\,\overline{r_y}\left[E_{s,y}+C_{s,y}\right]_+ + \underline{k_s}\,\underline{r_y}\left[E_{s,y}+C_{s,y}\right]_-$$
$$+2\overline{k_s}\,\overline{r_y}\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{V_{s,y}} - \underline{k_s}\,\underline{r_y}x_{s,y}[C_{s,y}]_+ - \overline{k_s}\,\overline{r_y}x_{s,y}[C_{s,y}]_-\Big) \tag{11.14a}$$

$$\text{s.t.} \quad \sum_{s=1}^{S}\underline{k_s} \leqslant 1, \quad \sum_{s=1}^{S}\overline{k_s} \geqslant 1, \quad \sum_{y=1}^{C}\underline{r_y} \leqslant 1, \quad \sum_{y=1}^{C}\overline{r_y} \geqslant 1, \tag{11.14b}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}\overline{k_s}\,\overline{r_y}x_{s,y}} \geqslant 1-\rho^2, \quad (1-\bar{\gamma}_{s,y}^2)^2 \leqslant x_{s,y} \leqslant 1 \quad \forall s\in[S], y\in[C], \tag{11.14c}$$

where $E_{s,y} = \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)]$, $V_{s,y} = \mathbb{V}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)]$ is the variance, $p_{s,y} = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{P}}[X_s = s, Y = y]$, $C_{s,y} = M - E_{s,y} - \frac{V_{s,y}}{M-E_{s,y}}$, and $\bar{\gamma}_{s,y}^2 = 1 - (1 + (M - E_{s,y})^2/V_{s,y})^{-\frac{1}{2}}$. Equation (11.13) only takes C's value when it is feasible, and each C queried by Equation (11.13) is a **convex optimization**.

Implications. Theorem 11.3 provides a fairness certificate for Problem 1 under two assumptions: (1) The loss function is bounded (by M). This assumption holds for several typical losses such as 0-1 loss and JSD loss. (2) The distribution shift between training and test distribution within each subpopulation is bounded by $\bar{\gamma}_{s,y}$, where $\bar{\gamma}_{s,y}$ is determined by the model's statistics on $\mathcal{P}$. In practice, this additional distance bound assumption generally holds, since $\bar{\gamma}_{s,y} \gg \rho$ for common choices of ρ.

In Theorem 11.3, we exploit three types of statistics of $h_{\boldsymbol{\theta}}(\cdot)$ on $\mathcal{P}$ to compute the fairness certificates: the expectation $E_{s,y} = \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)]$, the variance $V_{s,y} = \mathbb{V}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)]$, and the density $p_{s,y} = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{P}}[X_s = s, Y = y]$, all of which are at the subpopulation level and a high-confidence estimation of them based on finite samples are tractable (Section 11.3.4).

Using Theorem 11.3, after determining the region granularity T, we can provide a fairness certificate for Problem 1 by solving T^{SC} convex optimization problems, each of which has SC

decision variables. Note that the computation cost is independent of $h_{\boldsymbol{\theta}}$, and therefore we can numerically compute the certificate for large DNN models used in practice. Specifically, when $S = 2$ (binary sensitive attribute) or $C = 2$ (binary classification) which is common in the fairness evaluation setting, we can construct the region for only one dimension k_1 or r_1, and use $1 - k_1$ or $1 - r_1$ for the other dimension. Thus, for the typical setting $S = 2, C = 2$, we only need to solve T^2 convex optimization problems.

Note that for Problem 2, our certificate in Theorem 11.2 is tight, whereas for Problem 1, our certificate in Theorem 11.3 is not. This is because in Problem 1, extra distribution shift exists within each subpopulation, i.e., $\Pr(\boldsymbol{X}_o|Y, X_s)$ changes from $\mathcal{P}$ to $\mathcal{Q}$, and to bound such shift, we need to leverage Theorem 2.2 in [399] which has no tightness guarantee. Future work providing tighter bounds than [399] can be seamlessly incorporated into our framework to tighten our fairness certificate for Problem 1.

11.3.4 Dealing with Finite Sampling Error

In Section 11.3.2 and Section 11.3.3, we present Theorem 11.2 and Theorem 11.3 that provide computable fairness certificates for sensitive shifting and general shifting scenarios respectively. In these theorems, we need to know the quantities related to the training distribution and trained $\mathcal{P}$ and model $h_{\boldsymbol{\theta}}(\cdot)$:

$$E_{s,y} = \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)], V_{s,y} = \mathop{\mathbb{V}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)], p_{s,y} = \mathop{\Pr}_{(\boldsymbol{X},Y)\sim\mathcal{P}}[X_s = s, Y = y]. \tag{11.15}$$

Section 11.3.3 further requires $C_{s,y}$ and $\bar{\gamma}_{s,y}$ which are functions of $E_{s,y}$ and $V_{s,y}$. However, a practical challenge is that common training distributions do not have an analytical expression that allows us to precisely compute these quantities. Indeed, we only have access to a finite number of individually drawn samples, i.e., the training dataset, from $\mathcal{P}$. Thus, we will provide high-confidence bounds for $E_{s,y}$, $V_{s,y}$, and $p_{s,y}$ in Lemma H.1 (stated in Appendix H.2.1).

Then, for Theorem 11.2, we can replace $E_{s,y}$ in the objective by the upper bounds of $E_{s,y}$ and replace the concrete quantities of $p_{s,y}$ by interval constraints and the unit constraint $\sum_s \sum_y p_{s,y} = 1$, which again yields a convex optimization that can be effectively solved. For Theorem 11.3, we compute the confidence intervals of $C_{s,y}$ and $\rho_{s,y}$, then plug in either the lower bounds or the upper bounds to the objective (11.14a) based on the coefficient, and finally replace the concrete quantities of $p_{s,y}$ by interval constraints and the unit constraint $\sum_s \sum_y p_{s,y} = 1$. The resulting optimization is proved to be convex and provides an upper bound for any possible values of $E_{s,y}$, $V_{s,y}$, and $p_{s,y}$ within the confidence intervals. We

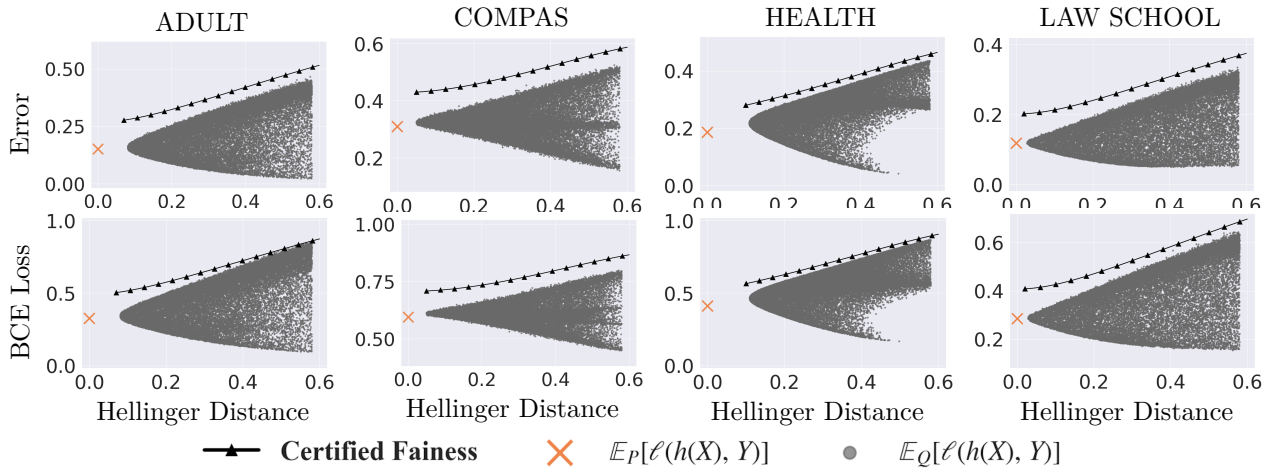


Figure 11.1: Certified fairness with sensitive shifting. Grey points are results on generated distributions ($\mathcal{Q}$) and the black line is our fairness certificate based on Theorem 11.2. We observe that our fairness certificate is usually tight.

defer the statement of Theorem 11.2 and Theorem 11.3 considering finite sampling error to Appendix H.2.2. To this point, we have presented our framework for computing high-confidence fairness certificates given access to model $h_{\boldsymbol{\theta}}(\cdot)$ and a finite number of samples drawn from $\mathcal{P}$.

11.4 EXPERIMENTS

In this section, we evaluate the certified fairness under both *sensitive shifting* and *general shifting* scenarios on six real-world datasets. We observe that under the sensitive shifting, our certified fairness bound is *tight* (Section 11.4.1); while the bound is less tight under general shifting (Section 11.4.2) which depends on the tightness of generalization bounds within each subpopulation (details in Section 11.3.3). In addition, we show that our certification framework can flexibly integrate more constraints on $\mathcal{Q}$, leading to a tighter fairness certification (Section 11.4.3). Finally, we compare our certified fairness bound with existing distributional robustness bound [347] (section 11.4.4), since both consider a shifted distribution while our bound is optimized with an additional fairness constraint which is challenging to be directly integrated to the existing distributional robustness optimization. We show that with the fairness constraint and our optimization approach, our bound is much tighter.

Dataset & Model. We validate our certified fairness on *six* real-world datasets: Adult [14], Compas [11], Health [161], Lawschool [403], Crime [14], and German [14]. Details on the

datasets and data processing steps are provided in Appendix H.3.1. Following the standard setup of fairness evaluation in the literature [253, 310, 313, 331], we consider the scenario that the sensitive attributes and labels take binary values. The ReLU network composed of 2 hidden layers of size 20 is used for all datasets.

Fairness Certification. We perform vanilla model training and then leverage our fairness certification framework to calculate the fairness certificate. Concretely, we input the trained model information on $\mathcal{P}$ and the framework would output the fairness certification for both sensitive shifting and general shifting scenarios following Theorem 11.2 and Theorem 11.3, respectively.

Code, model, and all experimental data are publicly available at `https://github.com/AI-secure/Certified-Fairness`.

11.4.1 Certified Fairness with Sensitive Shifting

Generating Fair Distributions. To evaluate how well our certificates capture the fairness risk in practice, we compare our certification bound with the empirical loss evaluated on randomly generated $30,000$ fairness constrained distributions $\mathcal{Q}$ shifted from $\mathcal{P}$. The detailed steps for generating fairness constrained distributions $\mathcal{Q}$ are provided in Appendix H.3.2. Under sensitive shifting, since each subpopulation divided by the sensitive attribute and label does not change (Section 11.2.1), we tune only the portion of each subpopulation $q_{s,y}$ satisfying the base rate fairness constraint, and then sample from each subpopulation of $\mathcal{P}$ individually according to the proportion $q_{s,y}$. In this way, our protocols can generate distributions with different combinations of subpopulation portions. If the classifier is biased toward one subpopulation (i.e., it achieves high accuracy in the group but low accuracy in others), the worst-case accuracy on generated distribution is low since the portion of the biased subpopulation in the generated distribution can be low; in contrast, a fair classifier which performs uniformly well for each group can achieve high worst-case accuracy (high certified fairness). Therefore, we believe that our protocols can demonstrate real-world training distribution bias as well as reflect the model's unfairness and certification tightness in real-world scenarios.

Results. We report the classification error (Error) and BCE loss as the evaluation metric. Figure 11.1 illustrates the certified fairness on Adult, Compas, Health, and Lawschool under sensitive shifting. More results on two relatively small datasets (Crime, German) are shown in Appendix H.3.5. From the results, we see that our certified fairness is tight in practice.

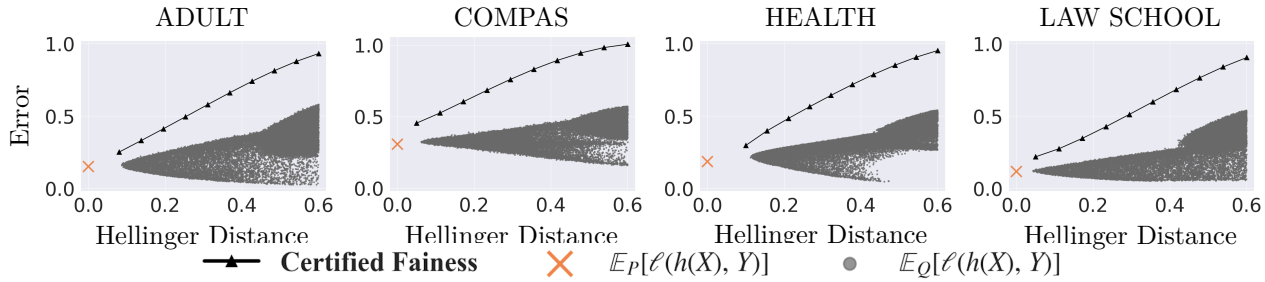


Figure 11.2: Certified fairness with general shifting. Grey points are results on generated distributions ($\mathcal{Q}$) and the black line is our fairness certificate based on Theorem 11.3. We observe that our fairness certificate is non-trivial.

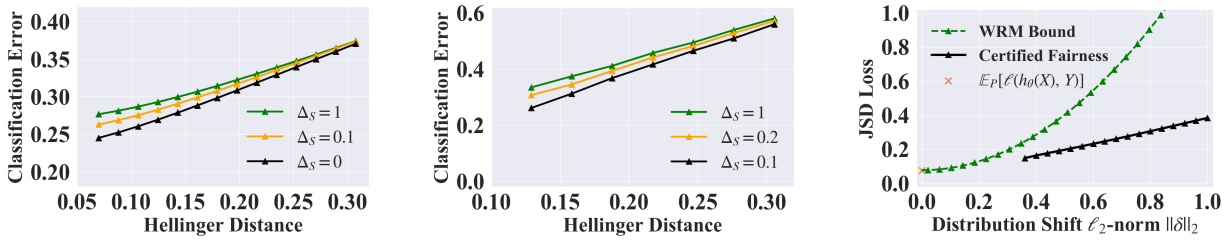
11.4.2 Certified Fairness with General Shifting

In the general shifting scenario, we similarly randomly generate $30,000$ fair distributions $\mathcal{Q}$ shifted from $\mathcal{P}$. Different from sensitive shifting, the distribution conditioned on sensitive attribute X_s and label Y can also change in this scenario. Therefore, we construct another distribution $\mathcal{Q}'$ disjoint with $\mathcal{P}$ on non-sensitive attributes and mix $\mathcal{P}$ and $\mathcal{Q}'$ in each subpopulation individually guided by mixing parameters satisfying fair base rate constraint. Detailed generation steps are given in Appendix H.3.2. Since the fairness certification for general shifting requires bounded loss, we select classification error (Error) and Jensen-Shannon loss (JSD Loss) as the evaluation metric. Figure 11.2 illustrates the certified fairness with classification error metric under general shifting. Results of JSD loss and more results on two relatively small datasets (Crime, German) are in Appendix H.3.5.

11.4.3 Certified Fairness with Additional Non-Skewness Constraints

In Section 11.2.1, we discussed that to represent different real-world scenarios we can add more constraints such as Equation (11.3) to prevent the skewness of $\mathcal{Q}$, which can be flexibly incorporated into our certificate framework. Concretely, for sensitive shifting, we only need to add one more box constraint[9] $0.5 - \Delta_s/2 \leqslant k_s \leqslant 0.5 + \Delta_s/2$ where Δ_s is a parameter controlling the skewness of $\mathcal{Q}$, which still guarantees convexity. For general shifting, we only need to modify the region partition step[9], where we split $[0.5 - \Delta_s/2, 0.5 + \Delta_s/2]$ instead of $[0, 1]$. The certification results with additional constraints are in Figures 11.3a and 11.3b, which suggests that if the added constraints are strict (i.e., smaller Δ_s), the bound is tighter. More constraints w.r.t. labels can also be handled by our framework and the corresponding results as well as results on more datasets are in Appendix H.3.6.

[9]Note that such modification is only viable when sentive attributes take binary values, which is the typical scenario in the literature of fairness evaluation [253, 310, 313, 331].



a Sentitive Shifting b General Shifting c Comparison with WRM

Figure 11.3: Certified fairness with additional non-skewness constraints on Adult dataset is shown in (a) (b). Δ_s controls the skewness of $\mathcal{Q}$ ($|\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[X_s = 0] - \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[X_s = 1]| \leqslant \Delta_s$). More analysis in Section 11.4.3. In (c), we compare our certified fairness bound with the distributional robustness bound [347]. More analysis in Section 11.4.4.

11.4.4 Comparison with Distributional Robustness Bound

To the best of our knowledge, there is no existing work providing *certified fairness* on the end-to-end model performance. Thus, we try to compare our bound with the distributional robustness bound since both consider certain distribution shifts. However, it is challenging to directly integrate the fairness constraints into existing bounds. Therefore, we compare with the state-of-the-art distributional robustness certification WRM [347], which solves the similar optimization problem as ours except for the fairness constraint. For fair comparison, we construct a synthetic dataset following [347], on which there is a one-to-one correspondence between the Hellinger and Wasserstein distance used by WRM. We randomly select one dimension as the sensitive attribute. Since WRM has additional assumptions on smoothness of models and losses, we use JSD loss and a small ELU network with 2 hidden layers of size 4 and 2 following their setting. More implementation details are in Appendix H.3.4. Results in Figure 11.3c suggest that 1) our certified fairness bound is much tighter than WRM given the additional fairness distribution constraint and our optimization framework; 2) with additional fairness constraint, our certificate problem could be infeasible under very small distribution distances since the fairness constrained distribution $\mathcal{Q}$ does not exist near the skewed original distribution $\mathcal{P}$; 3) with the fairness constraint, we provide non-trivial fairness certification bound even when the distribution shift is large.

11.5 BROADER IMPACT

This chapter aims to calculate a *fairness certificate* under some distributional fairness constraints on the performance of an end-to-end ML model. We believe that the rigorous fairness certificates provided by our framework CertFair will significantly benefit and advance

social fairness in the era of deep learning. Especially, such fairness certificate can be directly used to measure the fairness of an ML model regardless the target domain, which means that it will measure the unique property of the model itself with theoretical guarantees, and thus help people understand the risks of existing ML models. As a result, the ML community may develop ML training algorithms that explicitly reduce the fairness risks by regularizing on this fairness certificate.

A possible negative societal impact may stem from the misunderstanding or inaccurate interpretation of our fairness certificate. As a first step towards distributional fairness certification, we define the fairness through the lens of worst-case performance loss on a fairness constrained distribution. This fairness definition may not explicitly imply an absoluate fairness guarantee under some other criterion. For example, it does not imply that for any possible individual input, the ML model will give fair prediction. We tried our best in Section 11.2 to define the certification goal, and the practitioners may need to understand this goal well to avoid misinterpretation or misuse of our fairness certification.

11.6 SUMMARY

In this chapter, we provide the first *fairness certification* on end-to-end model performance, based on a fairness constrained distribution which has bounded distribution distance from the training distribution. We show that our fairness certification has strong connections with existing fairness notions such as group parity, and we provide an effective framework, CertFair, to calculate the certification under different scenarios. We provide both theoretical and empirical analysis of our fairness certification.

CHAPTER 12: NUMERICAL RELIABILITY: RANUM

With the wide deployment of deep learning (DL) systems, reliability issues of DL systems have become a serious concern, where malfunctioning DL systems have led to serious consequences such as fatal traffic accidents [258].

To assure the reliability of DL systems, it is highly critical to detect and fix numerical defects for two main reasons. First, numerical defects widely exist in DL systems. For example, in the DeepStability database [186], over 250 defects are identified in deep learning algorithms where over 60% of them are numerical defects. Moreover, since numerical defects exist at the architecture level, any model using the architecture naturally inherits these defects. Second, numerical defects can result in serious consequences. Once numerical defects (such as divide-by-zero) are exposed, the faulty DNN model will output `NaN` or `INF` instead of producing any meaningful prediction, resulting in numerical failures and system crashes [452, 457]. Thus, numerical defects hinder the application of DNNs in scenarios with high reliability and availability requirements such as threat monitoring in cybersecurity [297] and cloud system controlling [164, 330].

To address numerical defects in DNN architectures in an actionable manner [423], in this chapter, we propose a workflow of reliability assurance (as shown in Figure 12.1), consisting of three tasks: potential-defect detection, feasibility confirmation, and fix suggestion, along with our proposed approach to support all these three tasks.

Potential-Defect Detection. In this task, we detect all potential numerical defects in a DNN architecture via proposing a certification method, with a focus on operators with numerical defects (in short as defective operators) that potentially exhibit *inference-phase* numerical failures for two main reasons, following the literature [426, 460]. First, these defective operators can be exposed after the model is deployed and thus are more devastating than those that potentially exhibit training-phase numerical failures [278, 460]. Second, a defective operator that potentially exhibits training-phase numerical failures can usually be triggered to exhibit inference-phase numerical failures, thus also being detected by our task. For example, the type of training-phase `NaN` gradient failures is caused by an operator's input that leads to invalid derivatives, and this input also triggers failures in the inference phase [426].

Feasibility Confirmation. In this task, we confirm the feasibility of these potential numerical defects by generating failure-exhibiting system tests. As shown in Figure 12.1, a system

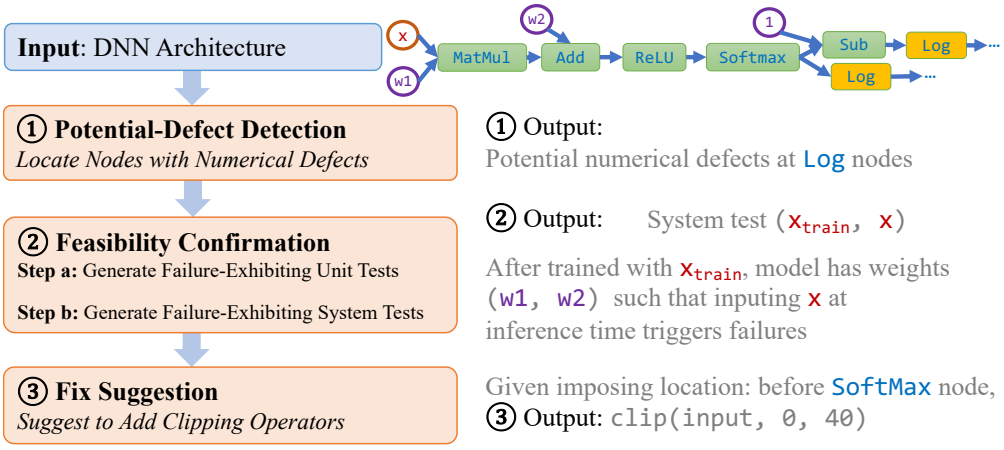


Figure 12.1: Workflow for reliability assurance against numerical defects in DNN architectures. The left-hand side shows three tasks and the right-hand side shows corresponding examples. RANUM supports all the three tasks, and is the first automatic approach for system test generation and fix suggestion.

test is a tuple of training example[10] $\boldsymbol{x}_{\text{train}}$ and inference example $\boldsymbol{x}$ such that after the training example is used to train the architecture under consideration, applying the resulting model on the inference example exhibits a numerical failure.

Fix Suggestion. In this task, we fix a feasible numerical defect. To determine the fix form, we have inspected the developers' fixes of the numerical defects collected by Zhang et al. [457] by looking at follow-up Stack Overflow posts or GitHub commits. Among the 13 numerical defects whose fixes can be located, 12 fixes can be viewed as explicitly or implicitly imposing interval preconditions on different locations, such as after inputs or weights are loaded and before defective operators are invoked. Thus, imposing an interval precondition, e.g., by clipping (i.e., chopping off the input parts that exceed the specified input range) the input for defective operator(s), is an effective and common strategy for fixing a numerical defect. Given a location (i.e., one related to an operator, input, or weight where users prefer to impose a fix), we suggest a fix for the numerical defect under consideration.

To support all the three tasks of the **r**eliability **a**ssurance process against DNN **num**erical defects, we propose the **RANUM** approach in this chapter.

For task ① and task ②a, which are already supported by two existing tools (DEBAR [460] and GRIST [426]), RANUM introduces novel extensions and optimizations that substantially improve the effectiveness and efficiency. (1) DEBAR [460] is the state-of-the-art tool for potential-defect detection; however, DEBAR can handle only static computational graphs

[10]In real settings, multiple training examples are used to train an architecture, but generating a single training example to exhibit failures (targeted by our work) is desirable for ease of debugging while being more challenging than generating multiple training examples to exhibit failures.

and does not support widely used dynamic graphs in PyTorch programs [289]. RANUM supports dynamic graphs thanks to our novel technique of *backward fine-grained node labeling*. (2) GRIST [426] is the state-of-the-art tool for generating failure-exhibiting unit tests to confirm potential-defect feasibility; however, GRIST conducts gradient back-propagation by using the original inference input and weights as the starting point. Recent studies [132, 252] on DNN adversarial attacks suggest that using a randomized input as the starting point leads to stronger attacks than using the original input. Taking this observation, we combine gradient back-propagation with random initialization in RANUM.

For task ② and task ③, which are not supported by any existing tool, RANUM is the first automatic approach for them.

For **feasibility confirmation**, RANUM is the **first** approach that generates failure-exhibiting **system** tests that contain training examples. Doing so is a major step further from the existing GRIST tool, which generates failure-exhibiting unit tests ignoring the practicality of generated model weights. Given that in practice model weights are determined by training examples, we propose the technique of *two-step generation* for this task. First, we generate a failure-exhibiting unit test. Second, we generate a training example that leads to the model weights in the unit test when used for training. For the second step, we extend the deep-leakage-from-gradient (DLG) attack [466] by incorporating the straight-through gradient estimator [34].

For **fix suggestion**, RANUM is the **first** automatic approach. RANUM is based on the novel technique of *abstraction optimization*. We observe that a defect fix in practice is typically imposing interval clipping on some operators such that each later-executed operator (including those defective ones) can never exhibit numerical failures. Therefore, we propose the novel technique of abstraction optimization to "deviate away" the input range of a defective operator from the invalid range, falling in which can cause numerical failures.

For RANUM, we implement a tool[11] and evaluate it on the benchmarks [426] of 63 real-world DNN architectures containing 79 real numerical defects; these benchmarks are the largest benchmarks of DNN numerical defects to the best of our knowledge. The evaluation results show that RANUM is both effective and efficient in all the three tasks for DNN reliability assurance. (1) For potential-defect detection, RANUM detects >60% more true defects than the state-of-the-art DEBAR approach. (2) For feasibility confirmation, RANUM generates failure-exhibiting unit tests to confirm potential numerical defects in the benchmarks with 100% success rate; in contrast, with the much higher time cost (17.32X), the state-of-the-art GRIST approach generates unit tests to confirm defects with 96.96%

[11]Open source at `https://github.com/llylly/RANUM`.

```
input_data = tf.placeholder("float", [1, n_features], name='x-input')
input_labels = tf.placeholder("float", [1, n_classes], name='y-input')
self.W_ = tf.Variable(tf.zeros([n_features, n_classes]),
                      name='weights')
self.b_ = tf.Variable(tf.zeros([n_classes]),
                      name='biases')
model_output = tf.nn.softmax(tf.matmul(input_data, self.W_) +
                             self.b_)
cost = -tf.reduce_mean(input_labels * tf.log(model_output) +
                       (1 - input_labels) * tf.log(1 - model_output),
                       name='cost')
self.obj_function = tf.reduce_min(tf.abs(model_output),
                                  name='obj_function')
```

Figure 12.2: A DL program snippet that defines a linear regression model from benchmarks of real-world numerical defects (Case 2a in [426]).

success rate. More importantly, for the first time, RANUM generates failure-exhibiting system tests that confirm defects (with 92.78% success rate). (3) For fix suggestion, RANUM proposes fix suggestions for numerical defects with 100% success rate. In addition, when the RANUM-generated fixes are compared with developers' fixes on open-source projects, in 37 out of 40 cases, RANUM-generated fixes are equivalent to or even better than human fixes.

This chapter makes the following main contributions:

- We formulate the reliability assurance problem for DNN architectures against numerical defects and elaborate on three important tasks for this problem.
- We propose RANUM—the first automatic approach that solves all these three tasks. RANUM includes three novel techniques (backward fine-grained node labeling, two-step test generation, and abstraction optimization) and solves system test generation and fix suggestion for the first time.
- We implement RANUM and apply it on 63 real-world DNN architectures, showing the high effectiveness and efficiency of RANUM compared to both the state-of-the-art approaches and developers' fixes.

12.1 BACKGROUND AND APPROACH OVERVIEW

In this section, we introduce the background of DNN numerical defects and failures, and then give an overview of the RANUM approach with a running example.

12.1.1 Background

DL developers define the DNN architecture with code using modern DL libraries such as PyTorch [289] and TensorFlow [1]. The DNN architecture can be expressed by a computational graph. Figures 12.2 and 12.3 depict a real-world example. Specifically, the DNN architecture in a DL program can be automatically converted to an ONNX-format computational graph [365].

The computational graph can be viewed as a Directed Acyclic Graph (DAG): $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V}$ and $\mathcal{E}$ are sets of nodes and edges, respectively. We call nodes with zero in-degree as *initial nodes*, which correspond to input, weight, or constant nodes. Initial nodes provide concrete data for the DNN models resulted from training the DNN architecture. The data from each node is formatted as a tensor, i.e., a multidimensional array, with a specified data type and array shape annotated alongside the node definition. We call nodes with positive in-degree as *internal nodes*, which correspond to concrete operators, such as matrix multiplication (`MatMul`) and addition (`Add`). During model training, the model weights, i.e., data from weight nodes, are generated by the training algorithm. Then, in the deployment phase (i.e., model inference), with these trained weights and a user-specified input named inference example, the output of each operator is computed in the topological order. The output of some specific node is used as the prediction result.

We let $\boldsymbol{x}$ and $\boldsymbol{w}$ denote the concatenation of data from all input nodes and data from all weight nodes, respectively. For example, in Figure 12.3, $\boldsymbol{x}$ concatenates data from nodes 1 and 11; and $\boldsymbol{w}$ concatenates data from nodes 2 and 4. Given specific $\boldsymbol{x}$ and $\boldsymbol{w}$, the input and output for each node are deterministic.[12] We use $f_n^{\mathsf{in}}(\boldsymbol{x}; \boldsymbol{w})$ and $f_n^{\mathsf{out}}(\boldsymbol{x}; \boldsymbol{w})$ to express input and output data of node n, respectively, given $\boldsymbol{x}$ and $\boldsymbol{w}$.

Numerical Defects in DNN Architecture. We focus on inference-phase numerical defects. These defects lead to numerical failures when specific operators receive inputs within invalid ranges so that the operators output `NaN` or `INF`.

Definition 12.1. For the given computational graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, if there is a node $n_0 \in \mathcal{V}$, such that there exists a valid input and valid weights that can let the input of node n_0 fall within the invalid range, we say there is a ***numerical defect*** at node n_0.

$$\begin{aligned} &\text{Formally, } \exists \boldsymbol{x}_0 \in \mathcal{X}_{\mathsf{valid}}, \boldsymbol{w}_0 \in \mathcal{W}_{\mathsf{valid}}, f_{n_0}^{\mathsf{in}}(\boldsymbol{x}_0; \boldsymbol{w}_0) \in \mathcal{I}_{n_0,\mathsf{invalid}} \\ &\quad \Longrightarrow \exists \text{ numerical defect at node } n_0. \end{aligned} \tag{12.1}$$

[12] An architecture may contain stochastic nodes. We view these nodes as nodes with randomly sampled data, so the architecture itself is deterministic.

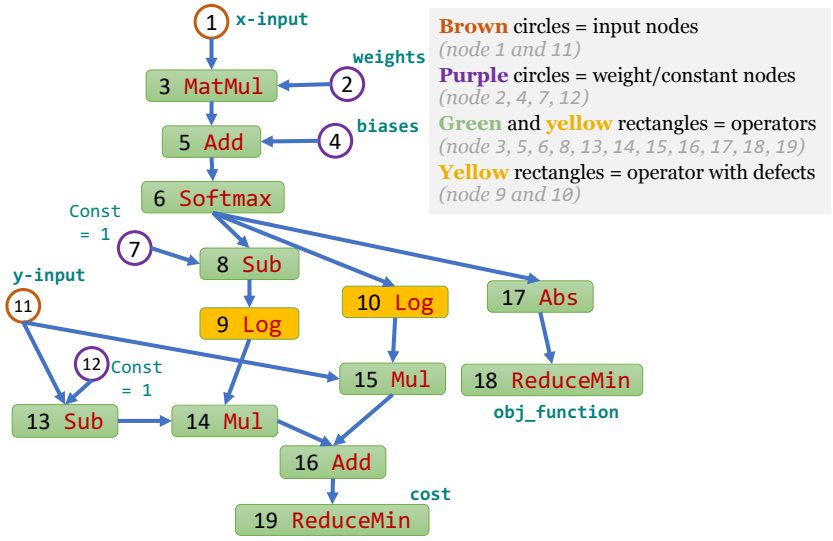


Figure 12.3: Computational graph encoded by the snippet in Figure 12.2.

In the definition, $\mathcal{X}_{\text{valid}}$ and $\mathcal{W}_{\text{valid}}$ are valid input range and weight range, respectively, which are clear given the deployed scenario. For example, ImageNet `Resnet50` models have valid input range $\mathcal{X}_{\text{valid}} = [0, 1]^{3\times224\times224}$ since image pixel intensities are within $[0, 1]$, and valid weight range $\mathcal{W}_{\text{valid}} = [-1, 1]^p$ where p is the number of parameters since weights of well-trained `Resnet50` models are typically within $[-1, 1]$. The invalid range $\mathcal{I}_{n_0,\text{invalid}}$ is determined by n_0's operator type with detailed definitions in Appendix I.2. For example, for the `Log` operator, the invalid range $\mathcal{I}_{n_0,\text{invalid}} = (-\infty, U_{\min})$ where $U_{\min}$ is the smallest positive number of a tensor's data type.

12.1.2 Approach Overview

In Figure 12.4, we show the overview structure of the RANUM approach. RANUM takes a DNN architecture as the input. Note that although RANUM is mainly designed and illustrated for a DNN architecture, the RANUM approach can also be directly applied to general neural network architectures since they can also be expressed by computational graphs. First, the DNN static analysis framework (task ① in Figure 12.1) in RANUM detects all potential numerical defects in the architecture. Second, the two-step test generation component (task ② in Figure 12.1), including unit test generation and training example generation, confirms the feasibility of these potential numerical defects. Third, the abstraction optimization component (task ③ in Figure 12.1) takes the input/output abstractions produced by the DNN static analysis framework along with the user-specified fix locations, and produces preconditions to fix the confirmed defects.

We next go through the whole process in detail taking the DNN architecture shown in

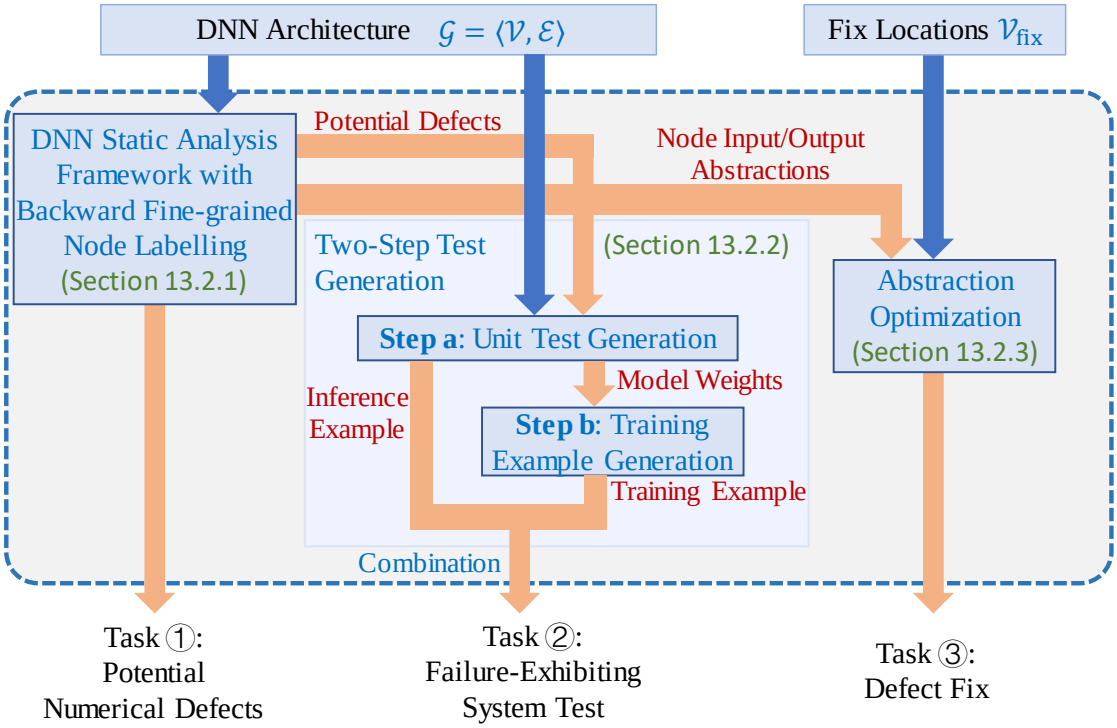


Figure 12.4: Overview of the RANUM approach. The output of RANUM indicates confirmation and manifestation of numerical defects (that can be feasibly exposed at the system level) for a given DNN architecture and effective fixes for architecture's confirmed defects.

Figure 12.3 as a running example.

Task ①: Potential-Defect Detection via Static Analysis. The DNN static analysis framework within RANUM first computes the numerical intervals of possible inputs and outputs for all nodes within the given DNN architecture, and then flags any nodes whose input intervals overlap with their invalid ranges as nodes with potential numerical defects. This framework can be viewed as a certification approach for numerical reliability.

In Figure 12.3, suppose that the user-specified input `x-input` (node 1) is within (elementwise, same below) range $[(-10, -10)^\intercal, (10, 10)^\intercal]$; `weights` (node 2) are within range $[\begin{bmatrix} -10 & -10 \\ -10 & -10 \end{bmatrix}, \begin{bmatrix} 10 & 10 \\ 10 & 10 \end{bmatrix}]$; and `biases` (node 4) are within range $[(-10, -10)^\intercal, (10, 10)^\intercal]$. Our DNN static analysis framework computes these interval abstractions for node inputs:

1. Node 5 (after `MatMul`): $[(-200, -200)^\intercal, (200, 200)^\intercal]$;
2. Node 6 (after `Add`): $[(-210, -210)^\intercal, (210, 210)^\intercal]$;
3. Node 8 (after `Softmax` in `float32`): $[(0, 0)^\intercal, (1, 1)^\intercal]$;
4. Node 9 (after `Sub` of $[1, 1]$ and node 8), 10: $[(0, 0)^\intercal, (1, 1)^\intercal]$.

Since nodes 9 and 10 use the `Log` operator whose invalid input range $(-\infty, U_{\min})$ overlaps with their input range $[(0,0)^\intercal, (1,1)^\intercal]$, we flag nodes 9 and 10 as potential numerical defects.

This static analysis process follows the state-of-the-art DEBAR tool [460]. However, we extend DEBAR with a novel technique named backward fine-grained node labeling. This technique detects all nodes that require fine-grained abstractions, e.g., nodes that determine the control flow in a dynamic graph. For these nodes, we apply interval abstractions with the finest granularity to reduce control flow ambiguity. For other nodes, we let some neighboring elements share the same interval abstraction to improve efficiency while preserving tightness. As a result, the static analysis in RANUM has high efficiency and supports much more DNN operators including dynamic control-flow operators like `Loop` than DEBAR does.

Task ②: Feasibility Confirmation via Two-Step Test Generation. Given nodes that contain potential numerical defects (nodes 9 and 10 in our example), we generate failure-exhibiting system tests to confirm their feasibility. A failure-exhibiting system test is a tuple $\langle \boldsymbol{x}_{\mathsf{train}}, \boldsymbol{x}_{\mathsf{infer}} \rangle$, such that after training the architecture with the training example $\boldsymbol{x}_{\mathsf{train}}$,[13] with the trained model weights $\boldsymbol{w}_{\mathsf{infer}}$, the inference input $\boldsymbol{x}_{\mathsf{infer}}$ triggers a numerical failure. The name "system test" is inspired by traditional software testing, where we test the method sequence (`m = train(`$\boldsymbol{x}_{\mathsf{train}}$`); m.infer(`$\boldsymbol{x}_{\mathsf{infer}}$`)`). In contrast, GRIST [426] generates model weights $\boldsymbol{w}_{\mathsf{infer}}$ along with inference input $\boldsymbol{x}_{\mathsf{infer}}$ that tests only the inference method `m.infer()`, and the weights may be infeasible from training. Hence, we view the GRIST-generated tuple $\langle \boldsymbol{w}_{\mathsf{infer}}, \boldsymbol{x}_{\mathsf{infer}} \rangle$ as a "unit test".

We propose a two-step test generation technique to generate failure-exhibiting system tests.

***Step a**: Generate failure-exhibiting unit test $\langle \boldsymbol{w}_{\mathsf{infer}}, \boldsymbol{x}_{\mathsf{infer}} \rangle$.* The state-of-the-art GRIST tool supports this step. However, GRIST solely relies on gradient back-propagation, which is relatively inefficient. In RANUM, we augment GRIST by combining its gradient back-propagation with random initialization inspired by recent research on DNN adversarial attacks [132, 252]. As a result, RANUM achieves 17.32X speedup with 100% success rate. Back to the running example in Figure 12.3, RANUM can generate $\begin{bmatrix} 5 & -5 \\ -5 & 5 \end{bmatrix}$ for node 2 and $(0.9, -0.9)^\intercal$ for node 4 as model weights $\boldsymbol{w}_{\mathsf{infer}}$; and $(10, -10)^\intercal$ for node 1 and $(1,0)^\intercal$ for node 11 as the inference input $\boldsymbol{x}_{\mathsf{infer}}$. Such $\boldsymbol{w}_{\mathsf{infer}}$ and $\boldsymbol{x}_{\mathsf{infer}}$ induce input $(0,1)^\intercal$ and $(1,0)^\intercal$ for nodes 9 and 10, respectively. Since both nodes 9 and 10 use the `log` operator and $\log 0$

[13] In particular, if our generation technique outputs $\boldsymbol{x}_{\mathsf{train}}$, the numerical failure can be triggered if the training dataset contains *only* $\boldsymbol{x}_{\mathsf{train}}$ or *only* multiple copies of $\boldsymbol{x}_{\mathsf{train}}$ and the inference-time input is $\boldsymbol{x}_{\mathsf{infer}}$. Our technique can also be applied for generating a batch of training examples by packing the batch as a single example: $\boldsymbol{x}_{\mathsf{train}} = (\boldsymbol{x}_{\mathsf{train}\,1}, \boldsymbol{x}_{\mathsf{train}\,2}, \ldots, \boldsymbol{x}_{\mathsf{train}\,B})$.

is undefined, both nodes 9 and 10 trigger numerical failures.

***Step b**: Generate training example $\boldsymbol{x}_{\mathsf{train}}$ that achieves model weights $\boldsymbol{w}_{\mathsf{infer}}$.* To the best of our knowledge, there is no automatic approach for this task yet. RANUM provides support for this task based on our extension of DLG attack [466]. The DLG attack is originally designed for recovering the training data from training-phase gradient leakage. Here, we figure out the required training gradients to trigger the numerical failure at the inference phase and then leverage the DLG attack to generate $\boldsymbol{x}_{\mathsf{train}}$ that leads to such training gradients. Specifically, many DNN architectures contain operators (such as `ReLU`) on which DLG attack is hard to operate [327]. We combine straight-through estimator [34] to provide proxy gradients and bypass this barrier. Back to the running example in Figure 12.3, supposing that the initial weights are $\begin{bmatrix} -0.1 & 0.1 \\ 0.1 & -0.1 \end{bmatrix}$ for node 2 and $(0,0)^\intercal$ for node 4, RANUM can generate training example $\boldsymbol{x}_{\mathsf{train}}$ composed of $(5.635, -5.635)^\intercal$ for node 1 and $(1,0)^\intercal$ for node 11, such that one-step training with learning rate 1 on this example leads to $\boldsymbol{w}_{\mathsf{infer}}$. Combining $\boldsymbol{x}_{\mathsf{train}}$ from this step with $\boldsymbol{x}_{\mathsf{infer}}$ from *step a*, we obtain a failure-exhibiting system test that confirms the feasibility of potential defects in nodes 9 and 10.

Task ③: Fix Suggestion via Abstract Optimization. In this task, we suggest fixes for the confirmed numerical defects. RANUM is the first approach for this task to our knowledge.

The user may prefer different fix locations, which correspond to a user-specified set of nodes $\mathcal{V}_{\mathsf{fix}} \subseteq \mathcal{V}$ to impose the fix. For example, if the fix method is clipping the inference input, $\mathcal{V}_{\mathsf{fix}}$ are input nodes (e.g., nodes 1, 11 in Figure 12.3); if the fix method is clipping the model weights during training, $\mathcal{V}_{\mathsf{fix}}$ are weight nodes (e.g., nodes 2, 4 in Figure 12.3); if the fix method is clipping before the defective operator, $\mathcal{V}_{\mathsf{fix}}$ are nodes with numerical defects (e.g., nodes 9, 10 in Figure 12.3).

According to the empirical study of developers' fixes at the beginning of this chapter, 12 out of 13 defects are fixed by imposing interval preconditions for clipping the inputs of $\mathcal{V}_{\mathsf{fix}}$. Hence, we suggest interval precondition, which is interval constraint $\boldsymbol{l}_n \leqslant f_n^{\mathsf{in}}(\boldsymbol{x}; \boldsymbol{w}) \leqslant \boldsymbol{u}_n$ for nodes $n \in \mathcal{V}_{\mathsf{fix}}$, as the defect fix in this chapter. A fix should satisfy that, when these constraints $\bigwedge_{n \in \mathcal{V}_{\mathsf{fix}}} (\boldsymbol{l}_n \leqslant f_n^{\mathsf{in}}(\boldsymbol{x}; \boldsymbol{w}) \leqslant \boldsymbol{u}_n)$ are imposed, the input of any node in the computational graph should always be valid, i.e., $f_{n_0}^{\mathsf{in}}(\boldsymbol{x}; \boldsymbol{w}) \notin \mathcal{I}_{n_0,\mathsf{invalid}}, \forall n_0 \in \mathcal{V}$.

In RANUM, we formulate the fix suggestion task as a constrained optimization problem, taking the endpoints of interval abstractions for nodes in $\mathcal{V}_{\mathsf{fix}}$ as optimizable variables. We then propose the novel technique of abstraction optimization to solve this constrained optimization problem. Back to the Figure 12.3 example, if users plan to impose a fix on

inference input, RANUM can suggest the fix $-1 \leqslant$ `x-input` $\leqslant 1$; if users plan to impose a fix on nodes with numerical defects, RANUM can suggest the fix $10^{-38} \leqslant$ node 9 & node 10`.input` $\leqslant +\infty$.

12.1.3 RANUM as Certified Approach

Viewing numerical reliability as a trustworthy property, we can use terms in Section 1.1 to characterize the three tasks above. In task ①, the RANUM approach is a white-box certification approach since the approach certifies whether the given DL system is numerically reliable or not. The approach is incomplete, since it may alert when the numerical defect is not feasible to trigger. In task ②, the RANUM approach is an attack since it generates concrete examples to trigger numerical defects. This attack is a complement to the incomplete certification for task ① by filtering possibly false alarms. In task ③, the RANUM approach is a certified training approach since the approach enhances numerical reliability. Unlike other training approaches that generate weights, RANUM improves numerical reliability by adding clipping operations, and the clipping intervals are derived from optimization similar to but improved from gradient descent as we will show in Section 12.2.3.

12.2 THE RANUM APPROACH

In this section, we introduce the three novel techniques in RANUM: backward fine-grained node labeling in Section 12.2.1; two-step test generation in Section 12.2.2; and abstraction optimization in Section 12.2.3.

12.2.1 DNN Static Analysis Framework with Backward Fine-grained Node Labeling for Potential-Defect Detection

RANUM contains a static analysis framework to enable potential-defect detection and support downstream tasks as shown in Figure 12.4. Given a DNN architecture and valid ranges for input and weight nodes, the static analysis framework computes interval abstractions for possible inputs and outputs of each node. As a result, we can check whether an overlap exists between the interval abstraction and invalid input ranges for all nodes in the graph to detect potential numerical defects. Then, the defective nodes are fed into the two-step test generation component to confirm the feasibility of potential defects; and the differentiable abstractions are fed into the abstract optimization component to produce fixes.

Formally, for given valid ranges of inference input and model weights, namely $\mathcal{X}$ and $\mathcal{W}$, for each node $n \in \mathcal{V}$, our framework computes *sound* input interval abstraction $[\boldsymbol{l}_n, \boldsymbol{u}_n] := \{\boldsymbol{x} : \boldsymbol{l}_n \leqslant \boldsymbol{x} \leqslant \boldsymbol{u}_n\}$ such that $[\boldsymbol{l}_n, \boldsymbol{u}_n]$ always captures all possible inputs of the node: $[\boldsymbol{l}_n, \boldsymbol{u}_n] \supseteq \{f_n^{\mathsf{in}}(\boldsymbol{x}, \boldsymbol{w}) : \boldsymbol{x} \in \mathcal{X}, \boldsymbol{w} \in \mathcal{W}\}$. We also compute output interval abstractions similarly.

Compared with traditional analysis tools for numerical software [136, 339], RANUM's static analysis framework designs abstractions for DNN primitives operating on multi-dimensional tensors that are not supported by traditional tools. Compared with the state-of-the-art DEBAR tool [460], RANUM uses the same abstraction domain (interval domain with tensor partitioning), but incorporates a novel technique (backward fine-grained node labeling) to improve abstraction precision and support a wider range of DNN architectures.

Abstract Domain: Interval with Tensor Partitioning. Following DEBAR's design, we use the interval with tensor partitioning [460] as the abstraction domain. This abstraction domain partitions a tensor into multiple subblocks and shares the interval abstractions at the block level instead of imposing abstractions at the element level. Therefore, we can compute the abstraction of a smaller size than the original tensor to improve efficiency.

Our Technique: Backward Fine-grained Node Labeling. The interval domain with tensor partitioning provides a degree of freedom in terms of the partition granularity, i.e., we can choose the subblock size for each node's abstraction. When the finest granularity, i.e., elementwise abstraction, is chosen, the abstraction interval is the most concrete. When the coarsest granularity (i.e., one scalar to summarize the node tensor) is chosen, the abstraction saves the most space and computational cost but loses much precision.

Example. Suppose that the possible input range of a node is $([-1, 0], [0, 1], [1, 2], [-1, 0])$, where each interval $[l, u]$ specifies the range of corresponding elements in the four-dimensional vector. If we choose the finest granularity, we use $[\boldsymbol{l}_n, \boldsymbol{u}_n] = [(-1, 0, 1, -1), (0, 1, 2, 0)]$ as the input range abstraction. If we choose the coarsest granularity, we use $[\boldsymbol{l}_n, \boldsymbol{u}_n] = [-1, 2]$ as the abstraction where the same interval is shared for all elements. As we can see, finer granularity provides tighter abstraction at the expense of larger computational and space costs.

In DEBAR, the coarsest granularity is used by default for most operators. However, we find that using the finest instead of the coarsest granularity for some nodes is more beneficial for overall abstraction preciseness. For example, the control-flow operators (e.g., `Loop`) benefit from concrete execution to determine the exact control flow in the dynamic graph, and the indexing operators (e.g., `Slice`) and shaping operators (e.g., `Reshape`) benefit

from explicit indexers and shapes to precisely infer the output range. Hence, we propose to use the finest granularity for some nodes (namely fine-grained requiring operators) while the coarsest granularity for other nodes during static analysis.

To benefit from the finest granularity abstraction for required nodes, typically, all of their preceding nodes also need the finest granularity. Otherwise, the over-approximated intervals from preceding nodes will be propagated to the required nodes, making the finest abstraction for the required nodes useless. To solve this problem, in RANUM, we back-propagate "fine-grained" labels from these fine-grained requiring nodes to initial nodes by topologically sorting the graph with *inverted* edges, and then apply the finest granularity abstractions on all labeled nodes. In practice, we find that this strategy eliminates the control-flow ambiguity and indexing ambiguity with little loss of efficiency[14]. As a result, RANUM supports all dynamic graphs (which are not supported by DEBAR) that comprise 39.2% of the benchmarks proposed by Yan et al. [426].

Furthermore, when preceding nodes use finer-grain abstraction granularity, the subsequent nodes should preserve such fine granularity to preserve the analysis preciseness. Principally, the choice of abstraction granularity should satisfy both tightness (bearing no precision loss compared to elementwise interval abstraction) and minimality (using the minimum number of partitions for high efficiency). To realize these principles, we dynamically determine a node's abstraction granularity based on the granularity of preceding nodes. The abstraction design for some operators is non-trivial. Omitted details (formulation, illustration, and proofs) about the static analysis framework are in Appendix I.3.

In summary, the whole static analysis process consists of three steps. (1) Determine the tensor partition granularity of all initial nodes by our technique of backward fine-grained node labeling. (2) Sort all nodes in the graph in the topological order. (3) Apply corresponding abstraction computation algorithms for each node based on the preceding node's abstractions. The key insight behind the design of our static analysis framework is the strategic granularity selection for tensor abstraction, maintaining both high efficiency (by selecting the coarse granularity for data-intensive nodes) and high precision (by selecting the fine granularity for some critical nodes, such as nodes with control-flow, indexing, and shaping operators).

[14]Theoretically, using the finest granularity for tensor partitioning cannot fully eliminate the ambiguity, since interval abstraction is intrinsically an over-approximation. Nevertheless, in our evaluation (Section 12.3), we find that this technique eliminates control-flow and indexing ambiguities on all 63 programs in the benchmarks.

12.2.2 Two-Step Test Generation for Feasibility Confirmation

RANUM generates failure-exhibiting system tests for the given DNN to confirm the feasibility of potential numerical defects. Here, we take the DNN architecture as the input. From the static analysis framework, we obtain a list of nodes that have potential numerical defects. For each node n_0 within the list, we apply our technique of two-step test generation to produce a failure-exhibiting system test $t_{\mathsf{sys}} = \langle \boldsymbol{x}_{\mathsf{train}}, \boldsymbol{x}_{\mathsf{infer}} \rangle$ as the output. According to Section 12.1.2, the test should satisfy that after the architecture is trained with $\boldsymbol{x}_{\mathsf{train}}$, entering $\boldsymbol{x}_{\mathsf{infer}}$ in the inference phase results in a numerical failure.

We propose the novel technique of two-step test generation: first, generate failure-exhibiting unit test $\langle \boldsymbol{w}_{\mathsf{infer}}, \boldsymbol{x}_{\mathsf{infer}} \rangle$; then, generate training example $\boldsymbol{x}_{\mathsf{train}}$ that leads model weights to be close to $\boldsymbol{w}_{\mathsf{infer}}$ after training.

Step a: Unit Test Generation. As sketched in Section 12.1.2, we strengthen the state-of-the-art unit test generation approach, GRIST [426], by combining it with random initialization to complete this step. Specifically, GRIST leverages the gradients of the defective node's input with respect to the inference input and weights to iteratively update the inference input and weights to generate failure-exhibiting unit tests. However, GRIST always conducts updates from the existing inference input and weights, suffering from local minima problem [252]. Instead, motivated by DNN adversarial attack literature [252, 371], a sufficient number of random starts help find global minima effectively. Hence, in RANUM, we first conduct uniform sampling 100 times for both the inference input and weights to trigger the numerical failure. If no failure is triggered, we use the sample that induces the smallest loss as the start point for gradient optimization. As Section 12.3.1 shows, this strategy substantially boosts the efficiency, achieving 17.32X speedup.

Step b: Training Example Generation. For this step, RANUM takes the following inputs: (1) the DNN architecture, (2) the failure-exhibiting unit test $t_{\mathsf{unit}} = \langle \boldsymbol{w}_{\mathsf{infer}}, \boldsymbol{x}_{\mathsf{infer}} \rangle$, and (3) the randomly initialized weights $\boldsymbol{w}_0$. Our goal is to generate a legal training example $\boldsymbol{x}_{\mathsf{train}}$, such that the model trained with $\boldsymbol{x}_{\mathsf{train}}$ will contain weights close to $\boldsymbol{w}_{\mathsf{infer}}$.

DNNs are typically trained with gradient-descent-based algorithms such as stochastic gradient descent (SGD). In SGD, in each step t, we sample a mini-batch of samples from the training dataset to compute their gradients on model weights and use these gradients to update the weights. We focus on one-step SGD training with a single training example, since generating a single one-step training example to exhibit a failure is more desirable for debugging because, in one-step training, the model weights are updated strictly following the

direction of the gradients. Therefore, developers can inspect inappropriate weights, easily trace back to nodes with inappropriate gradients, and then fix these nodes. In contrast, in multi-step training, from inappropriate weights, developers cannot trace back to inappropriate gradients because weights are updated iteratively and interactions between gradients and weights are complex (even theoretically intractable [228]).

In this one-step training case, after training, the model weights $\boldsymbol{w}_{\mathsf{infer}}$ satisfy

$$\boldsymbol{w}_{\mathsf{infer}} = \boldsymbol{w}_0 - \gamma \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{x}_{\mathsf{train}}; \boldsymbol{w}_0), \tag{12.2}$$

where $\gamma \in \mathbb{R}_+$ is a predefined learning rate, and $\mathcal{L}$ is the predefined loss function in the DNN architecture. Hence, our goal becomes finding $\boldsymbol{x}_{\mathsf{train}}$ that satisfies

$$\nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{x}_{\mathsf{train}}; \boldsymbol{w}_0) = (\boldsymbol{w}_0 - \boldsymbol{w}_{\mathsf{infer}})/\gamma. \tag{12.3}$$

The DLG attack [466] is a technique for generating input data that induce specific weight gradients. The attack is originally designed for recovering training samples from monitored gradient updates. Since the right-hand side (RHS) of Equation (12.3) is known, our goal here is also to generate input example $\boldsymbol{x}_{\mathsf{train}}$ that induces specific weight gradients. Therefore, we leverage the DLG attack to generate training example $\boldsymbol{x}_{\mathsf{train}}$.

Extending DLG Attack with Straight-Through Estimator. Directly using DLG attack suffers from an optimization challenge in our scenario. Specifically, in DLG attack, suppose that the target weight gradients are $\Delta \boldsymbol{w}_{\mathsf{targ}}$, we use gradient descent over the squared error $\|\nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{x}; \boldsymbol{w}_0) - \Delta \boldsymbol{w}_{\mathsf{targ}}\|_2^2$ to generate $\boldsymbol{x}$. In this process, we need meaningful gradient information of this squared error loss to perform the optimization. However, the gradient of this loss involves second-order derivatives of $\mathcal{L}(\boldsymbol{x}; \boldsymbol{w}_0)$, which could be zero. For example, DNNs with `ReLU` as activation function are piecewise linear and have zero second-order derivatives almost everywhere [327]. This optimization challenge is partly addressed in DLG attack by replacing `ReLU` with `Sigmoid`, but it changes the DNN architecture (i.e., the system under test) and hence is unsuitable.

We leverage the straight-through estimator to mitigate the optimization challenge. Specifically, for a certain operator, such as `ReLU`, we do not change its forward computation but change its backward gradient computation to provide second-order derivatives within the DLG attack process. For example, for `ReLU`, in backward computation we use the gradient of Softplus function, namely $1 - \frac{1}{1+\exp(\boldsymbol{x})}$, because Softplus is an approximation of `ReLU` [129] with non-zero second-order derivatives. Note that we modify the computed gradients only

within the DLG attack process. After such $\boldsymbol{x}_{\mathsf{train}}$ is generated by the attack, we evaluate whether it triggers a numerical failure using the original architecture and gradients in Equation (12.2).

Hyperparameters. For the unit test generation, we use the Adam optimizer where the learning rate is 1 and the maximum iteration number is 100. For the training example generation, we target for training example under learning rate $\gamma = 1$ and the approach has similar performance under other learning rates. We follow the convention in the DLG attack, where we use the L-BFGS method as the optimizer for gradient-based minimization. We terminate the method and return "`failed`" if either the running time exceeds 1800 s (universal execution time limit for all experimental evaluations), or a failure-exhibiting example training is not found after 300 iterations of L-BFGS optimization.

12.2.3 Abstraction Optimization for Fix Suggestion

In this task, we aim to generate the precondition fix given imposing locations. The inputs are the DNN architecture, the node n_0 with numerical defects, and a node set $\mathcal{V}_{\mathsf{fix}}$ to impose the fix. We would like to generate interval preconditions for $\mathcal{V}_{\mathsf{fix}}$ node inputs so that after these preconditions are imposed, the defect on n_0 is fixed.

Formally, our task is to find $\langle l_n, u_n \rangle$ for each $n \in \mathcal{V}_{\mathsf{fix}}$ (l_n and u_n are scalars so the same interval bound applied to all elements of n's tensor), such that for any $\boldsymbol{x}, \boldsymbol{w}$ satisfying $f_n^{\mathsf{in}}(\boldsymbol{x}; \boldsymbol{w}) \in [l_n, u_n]$, $\forall n \in \mathcal{V}_{\mathsf{fix}}$, for the defective node n_0, we have $f_{n_0}^{\mathsf{in}}(\boldsymbol{x}; \boldsymbol{w}) \notin \mathcal{I}_{n_0,\mathsf{invalid}}$, where the full list of invalid input ranges $\mathcal{I}_{n_0,\mathsf{invalid}}$ is in Appendix I.2. There is an infinite number of possible $\langle l_n, u_n \rangle$ interval candidates since l_n and u_n are floating numbers. Hence, we need an effective technique to find a valid solution from the exceedingly large search space that incurs a relatively small model utility loss. To achieve so, we formulate a surrogate optimization problem for this task.

$$\underset{l_n, u_n : n \in \mathcal{V}_{\mathsf{fix}}}{\text{maximize}} \quad s \quad \text{s.t.} \quad u_n \geqslant l_n + s(u_n^{\mathsf{valid}} - l_n^{\mathsf{valid}}), \forall n \in \mathcal{V}_{\mathsf{fix}}, \tag{12.4}$$

$$l_n^{\mathsf{valid}} \leqslant l_n \leqslant u_n \leqslant u_n^{\mathsf{valid}}, \forall n \in \mathcal{V}_{\mathsf{fix}}, \tag{12.5}$$

$$\mathcal{L}_{n_0}^{\mathsf{precond}}(\{l_n, u_n\}_{n \in \mathcal{V}_{\mathsf{fix}}}) < 0. \tag{12.6}$$

Here, l_n^{valid} and u_n^{valid} are the valid ranges (of the node's input n), which are fixed and determined by the valid ranges of input and weights. $\mathcal{L}_{n_0}^{\mathsf{precond}}$ is the node-specific precondition generation loss that is the distance between the furthest endpoint of defective node n_0's interval abstraction and n_0's valid input range. Hence, when $\mathcal{L}_{n_0}^{\mathsf{precond}}(\{l_n, u_n\}_{n \in \mathcal{V}_{\mathsf{fix}}})$ becomes

Algorithm 12.1: Abstraction Optimization (Section 12.2.3)

Input: DNN architecture $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$, defective node $n_0 \in \mathcal{V}$, nodes to impose fix $\mathcal{V}_{\mathsf{fix}} \subseteq \mathcal{V}$

1: $s \leftarrow 1, \gamma_s \leftarrow 0.9, \gamma_c \leftarrow 0.1, \mathsf{minstep} \leftarrow 0.1, \mathsf{maxiter} \leftarrow 1000$
2: $c_n \leftarrow (l_n^{\mathsf{valid}} + u_n^{\mathsf{valid}})/2, l_n \leftarrow l_n^{\mathsf{valid}}, u_n \leftarrow u_n^{\mathsf{valid}}, \forall n \in \mathcal{V}_{\mathsf{fix}}$
3: **for** $i = 1$ to $\mathsf{maxiter}$ **do**
4: **for** $n \in \mathcal{V}_{\mathsf{fix}}$ **do**
5: $\mathsf{loss} \leftarrow \mathcal{L}_{n_0}^{\mathsf{precond}}(\{l_{n'}, u_{n'}\}_{n' \in \mathcal{V}_{\mathsf{fix}}})$
6: $c_n \leftarrow c_n - \gamma_c \max\{|c_n|, \mathsf{minstep}\}\mathrm{sgn}(\nabla_{c_n} \mathsf{loss})$
7: $(l_n, u_n) \leftarrow (c_n - \frac{s(u_n^{\mathsf{valid}} - l_n^{\mathsf{valid}})}{2}, c_n + \frac{s(u_n^{\mathsf{valid}} - l_n^{\mathsf{valid}})}{2})$
8: $(l_n, u_n) \leftarrow (\max\{l_n, l_n^{\mathsf{valid}}\}, \min\{u_n, u_n^{\mathsf{valid}}\})$
9: **end for**
10: **if** $\mathcal{L}_{n_0}^{\mathsf{precond}}(\{l_n, u_n\}_{n \in \mathcal{V}_{\mathsf{fix}}}) < 0$ **then**
11: **return** $\{l_n, u_n\}_{n \in \mathcal{V}_{\mathsf{fix}}}$ {Find precondition fix}
12: **end if**
13: $s \leftarrow \gamma_s \cdot s$
14: **end for**
15: **return** "failed" {Failed to find precondition fix}

negative, the solution $\{l_n, u_n\}_{n \in \mathcal{V}_{\mathsf{fix}}}$ is a valid precondition. The optimization variables are the precondition interval endpoints l_n and u_n and the objective is the relative span of these intervals. The larger the span is, the looser the precondition constraints are, and the less hurt they are for the model's utility. Equation (12.4) enforces the interval span requirement. Equation (12.5) assures that the precondition interval is in the valid range. Equation (12.6) guarantees the validity of the precondition as a fix.

For any $\{l_n, u_n\}_{n \in \mathcal{V}_{\mathsf{fix}}}$, thanks to RANUM's static analysis framework, we can compute induced intervals of defective node n_0, and thus compute the loss value $\mathcal{L}_{n_0}^{\mathsf{precond}}$.

As shown in Algorithm 12.1, we propose the technique of **abstraction optimization** to effectively and approximately solve this optimization. Our technique works iteratively. In the first iteration, we set span $s = 1$, and in the subsequent iterations, we reduce the span s exponentially as shown in Line 13 where hyperparameter $\gamma_s = 0.9$. Inside each iteration, for each node to impose precondition $n \in \mathcal{V}_{\mathsf{fix}}$, we use the interval center $c_n = (l_n + u_n)/2$ as the optimizable variable and compute the *sign* of its gradient: $\mathrm{sgn}(\nabla_{c_n} \mathsf{loss})$. We use this gradient sign to update each c_n toward reducing the loss value in Line 6. Then, we use c_n and the span s to recover the actual interval in Line 7 and clip l_n and u_n by the valid range $[l_n^{\mathsf{valid}}, u_n^{\mathsf{valid}}]$ in Line 8. At the end of this iteration, for updated l_n and u_n, we compute $\mathcal{L}_{n_0}^{\mathsf{precond}}(\{l_n, u_n\}_{n \in \mathcal{V}_{\mathsf{fix}}})$ to check whether the precondition is a fix. If so, we terminate; otherwise, we proceed to the next iteration. We note that *if the algorithm finds a precondition, the precondition is guaranteed to be a valid fix* by the soundness nature of our static analysis framework and the definition of $\mathcal{L}_{n_0}^{\mathsf{precond}}$. When no feasible precondition is found within $\mathsf{maxiter} = 1000$

iterations, we terminate the algorithm and report "failed to find the fix".

Remark 12.1. The key ingredient in the technique is the gradient-sign-based update rule (shown in Line 6), which is much more effective than normal gradient descent for two reasons. (1) Our update rule can get rid of gradient explosion and vanishing problems. For early optimization iterations, the span s is large and interval bounds are generally coarse, resulting in too large or too small gradient magnitude. For example, the input range for `Log` could be $[1, 10^{10}]$ where gradient can be 10^{-10}, resulting in almost negligible gradient updates. In contrast, our update rule leverages the gradient sign, which always points to the correct gradient direction. The update step size in our rule is the maximum of current magnitude $|c_n|$ and $\mathsf{minstep}$ to avoid stagnation. (2) Our update rule mitigates the gradient magnitude discrepancy of different c_n. At different locations, the nodes in DNNs can have diverse value magnitudes that are not aligned with their gradient magnitudes, making gradient optimization challenging. Therefore, we use this update rule to solve the challenge, where the update magnitude depends on the value magnitude ($|c_n|$) instead of gradient magnitude ($\nabla_{c_n}\mathsf{loss}$). We empirically compare our technique with standard gradient descent in Section 12.3.3.

12.2.4 Implementation

We have implemented a tool for RANUM in roughly 10k lines of Python code based on PyTorch. Our tool leverages existing modules (`tf2onnx` for PyTorch and `torch.onnx.export` for Tensorflow and Keras) to extract ONNX-format DNN architectures from DL programs to take as the input, and automatically generates detection results, system tests, and defect fixes (given imposing locations). The ONNX format is supported by popular DL libraries such as PyTorch [289], Tensorflow [1], Keras [72], and GCP-CROWN (in Chapter 3). Our tool uses PyTorch [79] to provide the tensor computation and auto-differentiation functionality, which has high performance on both CPU and GPU environments.

12.3 EXPERIMENTAL EVALUATION

We conduct a systematic evaluation to answer the following research questions.

RQ1 For tasks already supported by existing state-of-the-art (SOTA) tools (tasks ① and ②a), how much more effective and efficient is RANUM compared to these SOTA tools?

RQ2 For feasibility confirmation via *generating failure-exhibiting system tests* (task ②), how much more effectively and efficiently can RANUM confirm potential numerical defects compared to baseline approaches?

RQ3 For *suggesting fixes* (task ③), how much more efficient and effective is RANUM in terms of guarding against numerical failures compared to baseline approaches and developers' fixes, respectively?

For RQ1, we compare RANUM with all SOTA tools. For RQ2 and RQ3, RANUM is the first approach to the best of our knowledge, so we compare RANUM with baseline approaches (constructed by leaving our novel techniques out of RANUM) and developers' fixes. We conduct the evaluation on the GRIST benchmarks [426], being the largest dataset of real-world DNN numerical defects to our knowledge. The benchmarks contain 63 real-world DL programs with numerical defects collected from previous studies and GitHub. Each program contains a DNN architecture, and each architecture has one or more numerical defects. There are 79 real numerical defects in total.

We perform our evaluation on a Linux workstation with a 24-core Xeon E5-2650 CPU running at 2.20 GHz. Throughout the evaluation, we stop the execution after reaching 30 min limit by following the evaluation setup by the most recent related work [426].

12.3.1 RQ1: Comparison with SOTA Tools

For two tasks, existing tools can provide automatic support: potential-defect detection (task ①) where the SOTA tool is DEBAR [460], and failure-exhibiting unit test generation (task ②a) where the SOTA tool is GRIST [426]. We compare RANUM with these tools on their supported tasks, respectively.

Comparison with DEBAR. RANUM successfully detects all 79 true defects and DEBAR detects only 48 true defects according to both our evaluation and the literature [426]. Hence, RANUM detects 64.58% more true defects than DEBAR. In terms of efficiency, DEBAR and RANUM have similar running time, and both finish in 3 s per case.

We manually inspect the cases where DEBAR fails but RANUM succeeds. They correspond to DL programs written with the PyTorch library, which generates dynamic computational graphs that DEBAR cannot handle. In contrast, RANUM provides effective static analysis support for dynamic computational graphs thanks to our backward fine-grained node labeling technique (Section 12.2.1) that is capable of disambiguating the control flow within dynamic graphs.

Comparison with GRIST. Results are shown in Table 12.1. Since both RANUM and GRIST have a randomness component where RANUM uses random initialization and GRIST relies on DNN's randomly initialized weights, we repeat both approaches for 10 runs, record the total number of times where a failure-exhibiting unit test is generated, and the average execution time per run. RANUM succeeds in *all* cases and *all* repeated runs, and GRIST fails to generate such unit test in 24 out of 790 runs (i.e., 96.96% success rate). RANUM has 6.66 s average execution time and is 17.32X faster than GRIST.

The superior effectiveness and efficiency of RANUM are largely due to the existence of random initialization as introduced in Section 12.2.2. We observe that since GRIST always takes initial model weights and inference input as the starting point to update from, the generated unit test is just slightly changed from initial weights and input, being hard to expose the target numerical defect. In contrast, RANUM uses random initialization to explore a much larger space and combines gradient-based optimization to locate the failure-exhibiting instances from the large space. We also evaluate the pure random strategy that uses only random initialization without gradient-based optimization, and such strategy fails in 30 runs, being inferior to both RANUM and GRIST, implying that both random initialization and gradient-based optimization are important. Among all the 79 cases, RANUM is slower than GRIST on only one case (28a). For this case, we find the default inference input loaded by the DNN program (used by GRIST) is not far from a failure-exhibiting one, but a randomly sampled inference input (used by RANUM) is usually far from that. Hence, RANUM takes more iterations to find a failure-exhibiting inference input by the nature of gradient-based optimization.

12.3.2 RQ2: Feasibility Confirmation via System Test Generation

In task ②, RANUM confirms the feasibility of potential numerical defects by generating failure-exhibiting system tests.

Baseline. Since RANUM is the first approach for this task, we do not compare with existing literature and propose one random-based approach (named "Random" hereinafter) as the baseline. In "Random", we first generate a failure-exhibiting unit test with random sampling. If there is any sample that triggers a failure, we stop and keep the inference example part as the desired $\boldsymbol{x}_{\mathsf{infer}}$. Then, we generate $\boldsymbol{x}_{\mathsf{train}}$ again by random sampling. If any sample, when used for training, could induce model weights $\boldsymbol{w}$ that cause a numerical failure when using $\boldsymbol{x}_{\mathsf{infer}}$ as the inference input, we keep that sample as $\boldsymbol{x}_{\mathsf{train}}$ and terminate. If and only if both $\boldsymbol{x}_{\mathsf{infer}}$ and $\boldsymbol{x}_{\mathsf{train}}$ are found, we count this run as a "success" one for

Table 12.1: (RQ1) Results of task ②a (failure-exhibiting **unit** test generation) with RANUM and GRIST [426]. C is the number of runs where numerical failures are triggered in 10 repeated runs, T is the average execution time per run, and ⇑T is the average time improvement achieved by RANUM compared to GRIST.

Case	RANUM			GRIST		Case	RANUM			GRIST		Case	RANUM			GRIST		Case	RANUM			GRIST	
ID	C	T	⇑T	C	T	ID	C	T	⇑T	C	T	ID	C	T	⇑T	C	T	ID	C	T	⇑T	C	T
1	10	9.01	1.20 X	10	10.77	16b	10	0.21	20.85 X	10	4.42	32	10	0.06	27.93 X	10	1.77	47	10	0.06	32.51 X	10	1.87
2a	10	0.02	9.75 X	10	0.24	16c	10	0.25	17.54 X	10	4.43	33	10	0.06	33.63 X	10	1.91	48a	10	0.38	3.06 X	10	1.17
2b	10	0.03	614.68 X	10	16.54	17	10	439.19	$+\infty$	0	-	34	10	0.06	33.95 X	10	1.90	48b	10	0.15	7.12 X	10	1.10
3	10	0.02	432.11 X	10	8.67	18	10	0.02	1040.46 X	10	22.17	35a	10	0.44	61.76 X	10	27.33	49a	10	0.49	41.09 X	10	20.07
4	10	0.01	1.00 X	10	0.01	19	10	0.16	689.66 X	10	107.78	35b	10	0.45	819.02 X	10	364.86	49b	10	0.50	612.22 X	10	307.24
5	10	0.05	6.48 X	10	0.34	20	10	0.16	3237.27 X	10	511.06	36a	10	0.44	41.80 X	10	18.58	50	10	0.16	781.02 X	10	126.80
6	10	0.84	5.20 X	10	4.38	21	10	0.16	259.73 X	10	42.09	36b	10	0.46	783.16 X	10	362.41	51	10	1.88	671.55 X	3	1263.04
7	10	0.87	4.54 X	10	3.96	22	10	0.94	1518.43 X	10	1433.12	37	10	0.06	38.34 X	10	2.39	52	10	0.15	336.37 X	10	50.59
8	10	0.86	4.63 X	10	3.99	23	10	0.01	157.72 X	10	1.88	38	10	0.06	34.50 X	10	1.94	53	10	0.05	36.64 X	10	1.92
9a	10	0.20	11.03 X	10	2.22	24	10	0.81	40.72 X	10	33.05	39a	10	0.43	42.79 X	10	18.30	54	10	0.05	36.76 X	10	1.83
9b	10	0.14	14.46 X	10	2.09	25	10	0.04	1271.88 X	10	44.70	39b	10	0.43	843.66 X	10	362.22	55	10	0.82	44.63 X	10	36.63
10	10	0.17	228.42 X	10	39.64	26	10	0.05	37.96 X	10	2.00	40	10	0.04	1995.27 X	10	85.97	56	10	0.06	35.04 X	10	1.93
11a	10	0.15	27.58 X	10	4.26	27	10	0.01	185.61 X	10	1.91	41	10	0.04	1967.23 X	10	86.36	57	10	0.01	177.45 X	10	1.88
11b	10	0.13	34.75 X	10	4.38	28a	10	24.37	-13.30 X	10	1.83	42	10	0.05	1934.84 X	10	87.89	58	10	0.83	12.01 X	10	9.95
11c	10	0.11	4499.86 X	10	516.13	28b	10	24.17	7.28 X	10	176.02	43a	10	0.48	35.63 X	10	16.96	59	10	0.02	105.40 X	10	1.94
12	10	0.26	135.94 X	10	34.69	28c	10	0.12	8.69 X	10	1.02	43b	10	0.45	4008.93 X	10	1800.00	60	10	0.15	221.97 X	10	34.19
13	10	0.01	1.10 X	10	0.01	28d	10	0.12	1518.28 X	10	176.02	44	10	0.27	579.29 X	10	155.38	61	10	0.35	53.29 X	10	18.78
14	10	0.80	107.96 X	10	86.23	29	10	0.89	16.83 X	10	14.98	45a	10	0.16	417.25 X	10	68.08	62	10	1.85	72.19 X	10	133.62
15	10	1.71	5.95 X	10	10.18	30	10	0.16	222.12 X	10	35.61	45b	10	0.88	14.69 X	10	12.98	63	10	2.06	117.12 X	10	240.68
16a	10	0.12	34.24 X	10	4.02	31	10	3.13	2.41 X	3	7.54	46	10	0.01	168.39 X	10	1.88	**Tot: 79**	**790**	**6.66**	**17.32 X**	**766**	**115.30**

Table 12.2: (RQ2) Results of task ② (failure-exhibiting **system** test generation) with RANUM and Random (baseline). C is the total number of runs where numerical failures are triggered in 10 repeated runs. T is the average execution time per run.

Case	RANUM		Random		Case	RANUM		Random		Case	RANUM		Random		Case	RANUM		Random		Case	RANUM		Random	
ID	C	T	C	T	ID	C	T	C	T	ID	C	T	C	T	ID	C	T	C	T	ID	C	T	C	T
1	0	9.01	0	1806.13	13	10	0.01	10	0.01	27	10	0.01	10	0.01	38	1	0.13	0	1800.65	49b	10	0.50	8	364.09
2a	10	0.03	10	0.06	14	10	12.60	10	0.50	28a	0	24.37	0	1920.29	39a	10	0.43	1	1623.72	50	10	4.89	10	0.16
2b	10	0.03	10	0.06	15	0	1.71	0	2107.24	28b	0	24.17	0	1911.26	39b	10	0.43	8	364.10	51	10	49.12	10	2.10
3	10	0.02	10	0.05	16a	10	0.12	10	0.75	28c	10	0.12	10	0.53	40	10	0.06	10	0.02	52	10	4.87	10	0.15
4	10	0.01	10	0.01	16b	10	0.21	0	1834.44	28d	10	0.12	10	0.48	41	10	0.06	10	0.02	53	10	0.07	10	0.03
5	10	0.05	10	0.06	16c	10	0.25	0	1831.67	29	10	0.89	10	11.96	42	10	0.06	10	0.02	54	10	0.07	10	0.02
6	10	0.84	10	12.42	17	10	549.98	10	235.11	30	10	4.88	10	0.14	43a	10	0.48	1	1623.71	55	10	0.82	10	12.79
7	10	0.87	10	12.51	18	10	0.02	10	0.05	31	10	14.62	10	9.31	43b	10	0.45	9	184.14	56	10	0.07	10	0.03
8	10	0.86	10	12.37	19	10	4.88	10	0.16	32	10	0.08	10	0.03	44	10	0.27	10	1.36	57	10	0.01	8	360.01
9a	10	0.20	7	541.25	20	10	4.88	10	0.14	33	10	0.07	10	0.02	45a	10	4.89	10	0.15	58	10	0.83	10	12.28
9b	10	0.14	10	1.39	21	10	4.89	10	0.14	34	10	0.42	10	0.20	45b	10	0.88	10	12.27	59	10	0.02	10	0.05
10	10	4.90	10	0.16	22	10	500.10	0	1801.60	35a	10	0.44	10	4.01	46	10	0.01	10	0.01	60	10	4.88	10	0.16
11a	10	0.15	10	0.72	23	10	0.01	10	0.01	35b	10	0.45	10	4.22	47	10	0.08	10	0.03	61	10	9.84	10	0.93
11b	10	0.13	10	0.76	24	10	0.81	10	12.52	36a	10	0.44	1	1623.76	48a	10	9.89	10	0.90	62	10	48.86	10	2.73
11c	10	0.11	10	0.74	25	10	0.04	10	0.15	36b	10	0.46	3	1263.79	48b	10	4.88	10	0.15	63	10	49.06	10	2.15
12	10	0.26	10	0.72	26	10	0.07	10	0.03	37	2	0.07	2	1440.50	49a	10	0.49	1	1623.88	**Tot: 79**	**733**	**17.31 (19.30X)**	**649**	**334.14**

"Random".

For each defect, due to the randomness of the model's initial weights, we repeat both RANUM and "Random" for 10 runs. Both approaches use the same set of random seeds.

Evaluation Result. Results are in Table 12.2. We observe that RANUM succeeds in $733/(79 \times 10) = 92.78\%$ runs and the baseline "Random" succeeds in $649/(79 \times 10) = 82.15\%$ runs. Moreover, RANUM spends only 17.31 s time on average per run, which is a 19.30X speedup compared to "Random". We also observe that RANUM is more reliable across repeated runs. There are only 6 cases with unsuccessful repeated runs in RANUM, but there are 19 such cases in "Random". Hence, RANUM is substantially more effective, efficient, and reliable for generating system tests than the baseline.

Ablation Study. RANUM uses the two-step test generation technique to produce failure-exhibiting system tests: it first generates failure-exhibiting unit tests with gradient back-propagation, then generates failure-exhibiting training examples via the extended DLG attack. To isolate the impact of RANUM at each step, we replace either step with random sampling: "Random + RANUM", which first generates failure-exhibiting unit tests via random sampling, then generates training example via RANUM; "RANUM + Random", which first generates failure-exhibiting unit tests via RANUM, then generates training example via random sampling. We follow the same evaluation protocol as above. We find that "RANUM + Random" takes 9.38X running time than RANUM and fails for 68 runs (RANUM only fails for 57 runs); and "Random + RANUM" fails for 113 runs (roughly 2X failed runs compared to RANUM). This study implies that RANUM's technique helps to improve effectiveness and efficiency at both steps of failure-exhibiting system test generation compared to the pure random baseline. The improvement for the first step is mainly from the effectiveness perspective, and the improvement for the second step is mainly from the efficiency perspective.

Discussion. The high effectiveness of RANUM mainly comes from the advantage of gradient-guided search compared with random search. As described in Section 12.2.2, RANUM leverages both first-order gradients (in step a) and second-order derivatives (in step b) to guide the search of system tests. In contrast, "Random" uses random sampling hoping that failure-exhibiting training examples can emerge after sufficient sampling. Hence, when such training examples are relatively rare in the whole valid input space, "Random" is less effective. The ablation study above shows that RANUM improves over "Random" in both steps inside RANUM.

Failing-case Analysis. We study all six defects where RANUM may fail and have the following findings. (1) For four defects (Case IDs 1, 15, 37, and 38), the architecture is challenging for gradient-based optimization, e.g., due to the `Min`/`Max`/`Softmax` operators that provide little or no gradient information. We leave it as future work to solve these cases, likely in need of dynamically detecting operators with vanishing gradients and reconstructing the gradient flow. (2) Two defects (Case IDs 28a and 28b) correspond to those caused by `Div` operators where only a close-to-zero divisor can trigger a numerical failure. Hence, for operators with narrow invalid ranges, RANUM may fail to generate failure-exhibiting system tests.

Table 12.3: (RQ3) Results of task ③ (fix suggestion) under three imposing location specifications with RANUM and two baselines (RANUM-E and GD). # is the number of fixes found. "Time (s)" is the total running time for all 79 cases.

Imposing Locations	RANUM		RANUM-E		GD	
	#	Time (s)	#	Time (s)	#	Time (s)
Weight + Input	**79**	**54.23**	78	540.13	57	188.63
Weight	**72**	**58.47**	71	581.86	43	219.28
Input	**37**	**924.74**	**37**	3977.30	29	952.19

12.3.3 RQ3: Fix Suggestion

In task ③, RANUM suggests fixes for numerical defects. We compare RANUM with fixes generated by baseline approaches and developers' fixes.

Comparison between RANUM and Baselines

RANUM is the first approach for this task, and we propose two baseline approaches to compare with. (1) RANUM-E: this approach changes the abstraction domain of RANUM from interval with tensor partitioning to standard interval. To some degree, RANUM-E represents the methodology of conventional static analysis tools that use standard interval domain for abstraction and search of effective fixes. (2) GD: this approach uses standard gradient descent for optimization instead of the abstraction optimization technique in RANUM.

Evaluation Protocol. We evaluate whether each approach can generate fixes that eliminate *all* numerical defects for the DNN architecture under analysis given imposing locations. We consider three types of locations: on both weight and input nodes, on only weight nodes, and on only input nodes. In practice, model providers can impose fixes on weight nodes by clipping weights after a model is trained; and users can impose fixes on input nodes by clipping their inputs before loading them into the model. Since all approaches are deterministic, for each case we run only once. We say that the fix eliminates all numerical defects if and only if (1) the RANUM static analysis framework cannot detect any defects from the fixed architecture; and (2) 1,000 random samples cannot trigger any numerical failures after imposing the fix.

Evaluation Result. We report the statistics, including the number of successful cases among all the 79 cases and the total running time, in Table 12.3. From the table, we observe that on all the three imposing location settings, RANUM always succeeds in most cases and spends much less time. For example, when fixes can be imposed on both weights

and input nodes, RANUM succeeds on *all* cases with a total running time 54.23 s. In contrast, RANUM-E requires $> 10\times$ time, and GD succeeds in only 72.15% cases. Hence, RANUM is substantially more effective and efficient for suggesting fixes compared to baseline approaches.

Since RANUM is based on iterative refinement (see Algorithm 12.1), we study the number of iterations needed for finding the fix. When fixes can be imposed on both weight and input nodes, where RANUM succeeds on all the 79 cases, the average number of iterations is 29.80, the standard deviation is 14.33, the maximum is 53, and the minimum is 2. Hence, when RANUM can find the fix, the number of iterations is small, coinciding with the small total running time 54.23 s.

Discussion. The two baseline approaches can be viewed as ablated versions of RANUM. Comparing RANUM and GD, we conclude that the technique of abstraction optimization substantially improves the effectiveness and also improves the efficiency. Comparing RANUM and RANUM-E, we conclude that the interval abstraction with tensor partitioning as the abstraction domain substantially improves the efficiency and also improves the effectiveness.

From Table 12.3, it is much easier to find the fix when imposing locations are weight nodes compared to input nodes. Since model providers can impose fixes on weights and users impose on inputs, this finding implies that fixing numerical defects on the providers' side may be more effective than on the users' side.

Comparison between RANUM and Developers' Fixes

We conduct an empirical study to compare the fixes generated by RANUM and by the developers.

Evaluation Protocol. We manually locate GitHub repositories from which the GRIST benchmarks are constructed. Among the 79 cases, we find the repositories for 53 cases on GitHub and we study these cases. We locate the developers' fixes of the numerical defects by looking at issues and follow-up pull requests. Since RANUM suggests different fixes for different imposing locations, for each case we first determine the imposing locations from the developer's fix, and then compare with RANUM's fix for these locations.

RANUM fixes are on the computational graph and developers' fixes are in the source code, so we determine to conduct code-centered comparison: RANUM fixes are considered feasible only when the fixes can be easily implemented by code (within 10 lines of code)

given that developers' fixes are typically small, usually in 3-5 lines of code. In particular, our comparison is based on two criteria: (1) which fix is sound on any valid input; (2) if both are sound, which fix hurts less to model performance and utility (based on the span of imposed precondition, the larger span the less hurt).

Results. We categorize the comparison results as below.

A *(30 cases) Better than developers' fixes or no available developer's fix.* Developers either propose no fixes or use heuristic fixes, such as reducing the learning rate or using the mean value to reduce the variance. These fixes may work in practice but are unsound, i.e., cannot rigorously guarantee the elimination of the numerical defect for any training or inference data. In contrast, RANUM generates better fixes since these fixes rigorously eliminate the defect.

B *(7 cases) Equivalent to developers' fixes.* Developers and RANUM suggest equivalent or highly similar fixes.

C *(13 cases) No need to fix.* For these cases, there is no need to fix the numerical defect in the given architecture. There are mainly three reasons. (1) The DNN is used in the whole project with fixed weights or test inputs. As a result, although the architecture contains defects, no system failure can be caused. (2) The architecture is injected a defect as a test case for automatic tools, such as a test architecture in the `TensorFuzz` [278] repository. (3) The defect can be hardly exposed in practice. For example, the defect is in a `Div` operator where the divisor needs to be very close to zero to trigger a divide-by-zero failure, but such situation hardly happens in practice since the divisor is randomly initialized.

D *(3 cases) Inferior than developers' fixes or RANUM-generated fixes are impractical.* In two cases, RANUM-generated fixes are inferior to developers' fixes. Human developers observe that the defective operator is `Log`, and its input is non-negative. So they propose to add 10^{-6} to the input of `Log` as the fix. In contrast, RANUM can generate only a clipping-based fix, e.g., clipping the input if it is less than 10^{-6}. When the input is small, RANUM's fix interrupts the gradient flow from output to input while the human's fix maintains it. As a result, the human's fix does less hurt to the model's trainability and is better than RANUM's fix. In another case, the RANUM-generated fix imposes a small span for some model weights (less than 0.1 for each component of that weight node). Such a small weight span strongly limits the model's expressiveness and utility. We leave it as the future work to solve these limitations.

From the comparison results, we can conclude that for the 40 cases where numerical defects are needed to be fixed (excluding case C), RANUM suggests equivalent or better fixes than human developers in 37 cases. Therefore, RANUM is comparably effective as human developers in terms of suggesting numerical-defect fixes, and is much more efficient since RANUM is an automatic approach.

Guidelines for Users. We discuss two practical questions for RANUM users. (1) *Does RANUM hurt model utility, e.g., inference accuracy?* If no training or test data ever exposes a numerical defect, RANUM does not confirm a defect and hence no fix is generated and there is no hurt to the utility. If RANUM confirms numerical defects, whether the fix affects the utility depends on the precondition-imposing locations. If imposing locations can be freely selected, RANUM tends to impose the fix right before the vulnerable operator, and hence the fix does not reduce inference performance. The reason is that the fix changes (by clipping) the input only when the input falls in the invalid range of the vulnerable operator. In practice, if the imposing locations cannot be freely selected and need to follow developers' requirements, our preceding empirical study shows that, in only 3 out of 40 cases, compared with developers' fixes, our fixes incur larger hurt to the inference or training performance of the architecture. (2) *Should we always apply RANUM to fix any architecture?* We can always apply RANUM to fix any architecture since RANUM fixes do not visibly alter the utility in most cases. Nonetheless, in deployment, we recommend first using RANUM to confirm defect feasibility. If there is no such failure-exhibiting system test, we may not need to fix the architecture; otherwise, we use RANUM to generate fixes.

12.4 THREATS TO VALIDITY

An external threat to validity is the evaluation subjects, which are the GRIST benchmarks [426], used to evaluate RANUM and baseline approaches. Although the subjects contain 63 real-world DL programs and are the largest to our knowledge, they still may not be representative enough. Another external threat is the approach randomness. To mitigate this threat, we repeat all randomized approaches for 10 runs. To mitigate bias in the empirical study, the author worked together with another researcher to independently conduct the study and discuss all cases to reach a consensus. A major internal threat to validity comes from the approach implementation. We reduce this threat by conducting code review and log checking extensively. A researcher independently checked the correctness of code implementation. We also verify the soundness of our static analysis framework with over 50 carefully designed unit tests.

12.5 RELATED WORK

Understanding and Detecting Defects in DL systems. Discovering and mitigating defects and failures in DL systems is an important research topic [186, 295, 457]. Following the taxonomy in previous work [157], DL defects are at four levels from bottom to top. (1) Platform-level defects. Defects can exist in real-world DL compilers and libraries. Approaches exist for understanding, detecting, and testing against these defects [333, 361, 395, 421]. (2) Architecture-level defects. *Our work focuses on numerical defects, being one type of architecture-level defects.* Automatic detection and localization approaches [235, 397] exist for other architecture-level defects such as suboptimal structure, activation function, and initialization and shape mismatch [140]. (3) Model-level defects. Once a model is trained, its defects can be viewed as violations of desired properties, such as all trustworthy properties discussed in previous chapters. (4) Interface-level defects. DL systems, when deployed as services, expose interaction interfaces to users where defects may exist, as shown by empirical studies on real-world systems [157, 382, 383].

Testing and Debugging for DNNs. A rich body of work exists for testing and debugging DNN defects [451]. Some representatives are DeepXplore [291] and DeepGauge [248]. Recent work enables automatic model debugging and repair via consistency checking [420], log checking [455], spectrum analysis [302], or analyzer-guided synthesis [350]. The RANUM approach contains testing (task ②) and debugging (task ③) component for numerical defects, and it also contains certification component (task ① for certifying numerical reliability and task ③ for generating certified fix).

Detecting and Exposing Numerical Defects in DNNs. Despite the widespread existence of numerical defects in real-world DL systems [157, 186, 457], only a few automatic approaches exist for detecting and exposing these defects. To the best of our knowledge, DEBAR [460] and GRIST [426] are the only two approaches. We discuss and compare RANUM with both approaches extensively in Sections 12.2 and 12.3.

12.6 SUMMARY

In this chapter, we have presented a novel automatic approach named RANUM for reliability assurance of DNNs against numerical defects. RANUM supports detection of potential numerical defects, confirmation of potential-defect feasibility, and suggestion of defect fixes. RANUM includes multiple novel extensions and optimizations upon existing tools, and in-

cludes three novel techniques. Our extensive evaluation on real-world DNN architectures has demonstrated high effectiveness and efficiency of RANUM compared to both the state-of-the-art approaches and developers' fixes.

Data Availability. All artifacts including the tool source code and experiment logs are available and actively maintained at `https://github.com/llylly/RANUM`.

CHAPTER 13: DISCUSSION FOR PART IV

In this part, we have proposed two certified approaches, CertFair and RANUM, for two trustworthy properties beyond robustness, distributional fairness and numerical reliability, respectively. Unlike robustness, these two trustworthy properties are defined uniquely — the distributional fairness property is defined at the distribution level for some statistics (expected loss), rather than at the instance level for a boolean predicate; the numerical reliability property is defined for the whole valid input space, rather than for some local perturbations.

As we can see, even though these two properties are so different from robustness, the certified approaches can be extended to certify these two properties successfully, demonstrating the flexibility and extensibility of methodologies in certified approaches. In CertFair, we extend the distribution overlap characterization for certifying robustness of randomized smoothing to bring certification of expected loss, and then propose a subpopulation decomposition technique to decompose fairness certification problem into optimizing linear combinations of these expected losses. In the future, we believe that this type of extension for distribution overlap characterization can further solve more certification problems involving distribution measures and statistics. In RANUM, we extend the white-box bounding technique to certify a practical trustworthy threat in DL systems — numerical reliability, and further propose a fixing technique based on optimizing the bounds. Before RANUM, the bounding techniques are limited to certifying robustness or closely-related properties, and RANUM shows their great potential in certifying other trustworthy properties and their connection to program analysis. In the future, we believe that this type of extension of bounding techniques can further solve more certification problems involving output uncertainties (e.g., for content legality of generative models) and neuro-symbolic systems.

Part V

Outlook and Conclusion

CHAPTER 14: OUTLOOK AND CONCLUSION

This thesis has developed a systematic methodology for enhancing and extending certified approaches for deep learning systems. Starting from a taxonomy and characterization of trustworthy properties and certified approaches in Part I, the thesis first investigates the robustness certification against ℓ_p-bounded perturbations in depth in Part II, by enhancing both black-box and white-box certified robustness with certification and certified training approaches respectively. Then, the thesis focuses on broadening the scope of certified approaches. Part III illustrates how to achieve certified robustness beyond ℓ_p-bounded perturbations. Inspired by real-world scenarios, robustness against semantic transformations and robustness in reinforcement learning are studied and state-of-the-art certified approaches are proposed under these robustness notions. Next, Part IV further extends the certified approaches beyond robustness, by developing scalable and tight certified approaches for distributional fairness and numerical reliability.

In summary, through extensive theoretical analysis and empirical evaluations, this thesis proves that the certified trustworthiness of large-scale deep learning systems is feasible by developing a series of scalable and tight certified approaches to provide and enhance the certified trustworthiness of large-scale deep learning systems (with millions of parameters) to the state-of-the-art level. The author believes that the thesis represents an important step in certified machine learning, a topic of interest to machine learning, computer security, software engineering, and programming languages.

Challenges. Despite the remarkable progress, certified approaches for DL systems are still far from perfect. As sketched in Chapter 6, certification approaches are not scalable and tight enough. As a result, certified training approaches need to strongly regularize the DL models to achieve a decent level of certifiability, often resulting in a significant loss of utility, i.e., accuracy. Taking robustness against ℓ_p-bounded perturbations as an example, on MNIST, a small-scale dataset, the certified robust accuracy against ℓ_∞-bounded adversary with 0.3 radius has reached over 94%. This is remarkable since the limit is 0.5 radius where any input image can be perturbed to indistinguishable half-gray. However, on more challenging CIFAR-10 and ImageNet datasets, certified robust accuracy is still low. On CIFAR-10,

against ℓ_∞ adversary with radius 8/255, the certified robust accuracy is only 40.39%. On the large-scale ImageNet dataset, the certified robust accuracy is only 42.2% under ℓ_2 radius 2.0. Hence, on large-scale datasets, usually, there is a gap between the certified level of trustworthiness and desired level of trustworthiness.

Another critical challenge is to deal with the complex nature of the real world. When DL systems are deployed in the real world, it is always challenging to foresee all possible security and trustworthy threats. Even if we can certify the degree of trustworthiness when the attacker falls into some predefined range, e.g., ℓ_p-bounded perturbations, there is no guarantee whether the attacker will always stick to such range or such attack vector. Hence, all certified approaches cannot provide a full certificate of trustworthiness, but just a relatively strong one. It is always an important research topic to discover, identify, monitor, and formalize trustworthy issues so that certification approaches can be developed to mitigate them. On the other hand, more recently, the development of generative models and language models adds another layer of real-world complexity. For these models, even though threats can be identified, they are hard to certify due to their unbounded and discrete nature. For example, the space of the generative and language model's output is almost infinite and it is an open problem how to make sure any generated content is ethical, non-toxic, unbiased, and privacy-preserving.

Future Directions. The author outlines a few future directions toward solving the above challenges and beyond.

1. **Tight and Scalable Certification for Trustworthiness.** As discussed above, tightness and scalability are core goals of trustworthiness certification approaches. The author proposes two principles to guide the design toward tighter and more scalable certification: (1) *Integration of domain-specific or task-specific knowledge*: Pure data-driven learning may be intrinsically limited toward human-level trustworthiness for complex tasks [45]. Hence, domain-specific or task-specific knowledge can be leveraged. As a future step, the author believes that automated knowledge collection, knowledge distillation, knowledge integration, and knowledge-based data generation, for both certification and certified training approaches, could be the key to large-scale certified trustworthiness. (2) *Efficient abstraction of knowledge-enhanced DL system*: Certification approaches can be viewed as constructing abstractions for the DL system to characterize its behavior on the threat model set. The abstraction of existing certification is usually too generic, which handles the model in a way agnostic of its training or constructing method. In the next-generation approach for DL certification,

knowledge and reasoning components will be integrated into the abstraction to derive a much tighter and more scalable certification.

2. **Generic Certified Trustworthiness.** Besides certification for existing trustworthy properties, certification is also in need against more foreseen or unforeseen trustworthy threats. In this aspect, the author proposes a roadmap containing three stages: *Stage 1: Discover, define, and certify more practical trustworthy properties.* Toward trustworthy AI, we need to identify more trustworthy properties that incur profound social impacts. After properties are defined, we need to define them formally and propose corresponding certification approaches. To this end, one of the future research questions is "Can we formally construct a DL-based system, such as a DL-based autonomous vehicle or robotic agent, that is guaranteed to be universally safe when deployed in the physical world?" *Stage 2: Certify multiple trustworthy properties at the same time.* Existing certified methods mainly provide certified trustworthiness for a single property. Can we achieve certified trustworthiness under multiple properties at the same time? If so, what would be the efficient method? If not, is there any inherent trade-off between different properties? What else (e.g., more data, more structured knowledge, more human supervision, or better algorithms) do we need to achieve so? *Stage 3: Develop certifiably trustworthy DL systems holding multiple foreseen and unforeseen properties.* Instead of defining trustworthy properties and developing methods to certify them, is it possible to achieve "meta-trustworthiness"? In other words, can we develop DL systems in a property-agnostic way to achieve human-level trustworthiness under all existing notions (robustness, fairness, privacy, etc.) and possible future notions?

3. **Deployment of Certified Trustworthy Systems and Study of Social Impacts.** There is no free lunch—the certified methods for DL systems come at a cost. Costs include inference overhead, training overhead, and normal performance degradation. These negative costs are understudied but impose great barriers to deploying certified DL systems in the real world. To deploy certified trustworthy systems in practice, we need to mitigate these practical challenges. We also need to understand the practical implications and social impacts of certifiably trustworthy systems—if certifiably trustworthy DL systems are deployed, to what extent they can benefit social good, equality, democracy, and inclusion.

4. **Interdescriplinary Research.** Lastly, the author believes that the ultimate goal of certified AI trustworthiness is to achieve fully controllable AI, i.e., to learn while

perfectly conforming to human specifications and safety constraints. Hence, research in other disciplines chasing controllable AI, such as robotics, control theory, and reinforcement learning, may bring many potential opportunities to better certified AI trustworthiness.

As a closing remark, AI revolution is at the corner. Without certified trustworthiness, human is not ready for AI revolution. It is our researcher's responsibility to equip AI with certification, so human can embrace the AI revolution. This thesis is just a milestone, but never the end of the author's quest for fully certified AI trustworthiness.

APPENDIX A: ROBUSTNESS AGAINST SEMANTIC TRANSFORMATIONS ON POINT CLOUD MODELS

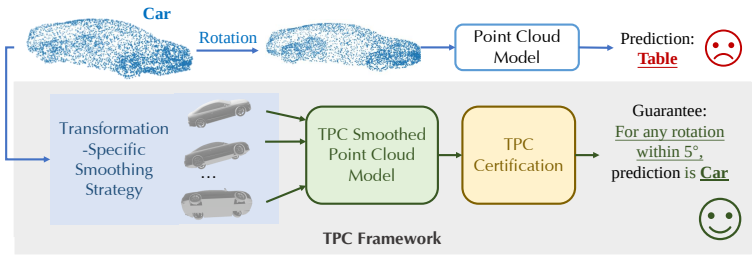


Figure A.1: Overview of TPC framework. TPC includes smoothing and certification strategies to provide certified robustness for point cloud models against semantic transformations. Besides the rotation as shown in the figure, TPC provides strong robustness certification for a wide range of other semantic transformations.

Besides image models, in computer vision, deep neural networks that take point clouds data as inputs (point cloud models) are also widely used [301, 391, 465] especially in autonomous driving [61, 63, 217]. For instance, modern autonomous driving systems are equipped with LiDAR sensors that generate point cloud inputs to feed into point cloud models [50]. Despite their successes, point cloud models are shown to be vulnerable to adversarial attacks that mislead the model's prediction by adding stealthy perturbations to point coordinates or applying semantic transformations (e.g., rotation, shearing, tapering) [50, 51, 412, 417]. Specifically, semantic transformation based attacks can be easily operated on point cloud models by simply manipulating sensor positions or orientations [50, 51]. These attacks may lead to severe consequences such as forcing an autonomous driving vehicle to steer toward the cliff [291]. A wide range of empirical defenses against these attacks has been studied [12, 301, 355, 356, 357, 467], while defenses with robustness guarantees are less explored [120, 241] and provides loose and less scalable certification.

In this appendix, we extend the TSS framework in Chapter 7 to point cloud models, yielding TPC that provides *tight* and *scalable probabilistic* robustness guarantees for point cloud models against a wide range of semantic transformation attacks. Following the methodology in TSS, we first categorize common semantic transformations into three categories: additive (e.g., shearing), composable (e.g., rotation), and indirectly composable (e.g., tapering). For each category, our framework proposes novel *smoothing* and *robustness certification* strategies. With TPC, for each common semantic transformation or composition, we prove the corresponding robustness conditions that yield efficient and tight robustness certification.

For example, regarding general rotation based attacks, we first prove that it is a type

of *composable* transformations; we then propose a corresponding smoothing strategy and certify the robustness of the smoothed model with a novel sampling-based algorithm, which is shown to have sound and tight input-dependent sampling error bound. TPC achieves 69.2% certified robust accuracy for any rotation within 10°. To the best of our knowledge, no prior work can provide robustness certification for rotations within such large angles.

In addition to our theoretical analysis for the certification against different types of semantic transformations, we conduct extensive experiments to evaluate TPC. Compared with existing baselines, our TPC achieves *substantially* higher certified robust accuracy. For example, for any twisting along z-axis within $\pm 20^\circ$, we achieve the probabilistic certified accuracy of 83.8% with a high confidence level 99.9%, while the existing baseline [241] provides a deterministic certified accuracy of 20.3%. Furthermore, compared with prior works, we show that TPC can: (1) certify a more general class of semantic transformations; (2) certify large-size point clouds; and (3) certify under large perturbation magnitudes. We also show that TPC can certify the robustness for multiple tasks on 3D point clouds, including classification and part segmentation.

We illustrate our TPC framework in Figure A.1, and we summarize the main technical **contributions** as follows.

- We propose a general robustness certification framework TPC for point cloud models. We categorize common semantic transformations of point clouds into three categories: additive, composable, and indirectly composable, and provide general smoothing and certification strategies for each.
- We concretize our framework TPC to provide transformation-specific smoothing and certification for various realistic common semantic transformations for point clouds, including rotation, shearing, twisting, and tapering as well as their compositions.
- We conduct extensive experiments and show that TPC (1) achieves significantly higher certified robust accuracy than baselines, (2) provides certification for large-size point clouds and large perturbation magnitudes, and (3) provides efficient and effective certification for different tasks such as classification and part segmentation.

A.1 RELATED WORK

Our TPC aims to generalize the model robustness certification to point cloud models against a more generic family of practical attacks – semantic transformation attacks. The

semantic transformation attacks have been shown feasible for both image classification models and point cloud models [50, 145, 412], and certified robustness against such attacks is mainly studied for 2D image classification models [22, 118, 223]. For point cloud models, some work considers point *addition* and *removal* attacks [412] and provides robustness certification against such attacks [236]. The randomized smoothing technique is applied to certify point cloud models on segmentation tasks by [120]. However, their certification only covers points edition with bounded ℓ_2 norm and rotations along a fixed axis. For general semantic transformation attacks, to the best of our knowledge, the only work that can provide robustness certification against them is DeepG3D [241], which is based on linear bound relaxations. In this work, we derive novel randomized smoothing techniques on point clouds models to provide probabilistic robustness certification against semantic transformations. In Appendix A.5, we conduct extensive experiments to show that our framework is more general and provides significantly higher certified robust accuracy than DeepG3D under different settings.

A.2 SEMANTIC TRANSFORMATION ATTACKS ON POINT CLOUD MODELS

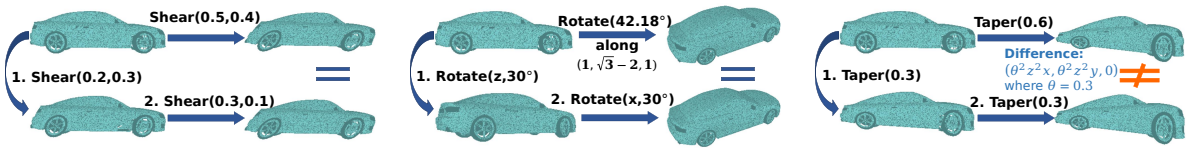
We denote the space of point cloud inputs as $\mathcal{X} = \mathbb{R}^{N\times 3}$ where N is the number of points the point cloud has. A point cloud with N points is denoted by $\boldsymbol{x} = \{\boldsymbol{p}_i\}_{i=1}^N$ with $\boldsymbol{p}_i \in \mathbb{R}^3$. Unless otherwise noted, we assume all point cloud inputs are normalized to be within a unit ball, i.e., $\|\boldsymbol{p}_i\|_2 \leqslant 1$. We mainly consider classification tasks on the point clouds level. Such classification task is defined with a set of labels $\mathcal{Y} = [C] = \{1, \ldots, C\}$ and a classifier is defined by a deterministic function $h : \mathcal{X} \to \mathcal{Y}$. More extensions are in Appendix A.5.2.

A.2.1 Semantic Transformations

Semantic transformations on point cloud models are defined as functions $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ where $\mathcal{Z}$ is the parameter space for transformations. The semantic transformations discussed in this paper may change the three-dimensional coordinate of each point (usually in a pointwise manner) but do not increase or decrease the number of points. In Appendix A.3, we will further categorize different semantic transformations based on their intrinsic properties.

A.2.2 Threat Model and Certification Goal

We consider semantic transformation attacks that an adversary can apply arbitrary semantic transformations to the point cloud data according to a parameter $z \in \mathcal{Z}$. The



a Additive. b Composable. c Indirectly composable.

Figure A.2: Illustration of different types of transformations. (a) additive transformations (e.g., shearing), (b) composable transformations (e.g., rotation), and (c) indirectly composable transformations (e.g., tapering).

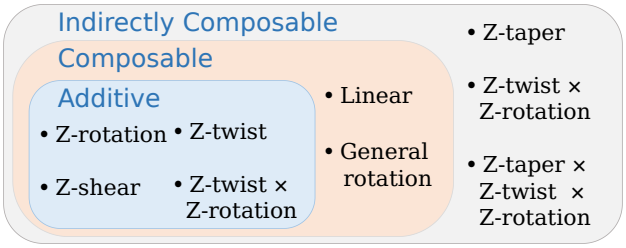


Figure A.3: Taxonomy of common 3D semantic transformations for point clouds.

adversary then performs evasion attacks to a classifier h with the transformed point cloud $\phi(\boldsymbol{x}, z)$. The attack is successful if h predicts different labels on x and $\phi(\boldsymbol{x}, z)$[15].

The main goal of this paper is to certify the robustness of point cloud classifiers against all semantic attacks within a certain transformation parameter space. Formally, our **certification goal** is to find a subset $\mathcal{Z}_{\text{robust}} \subseteq Z$ for a classifier $h : \mathcal{X} \to \mathcal{Y}$, such that

$$h(\boldsymbol{x}) = h(\phi(\boldsymbol{x}, z)), \forall z \in \mathcal{Z}_{\text{robust}}. \tag{A.1}$$

A.3 TRANSFORMATION SPECIFIC SMOOTHING FOR POINT CLOUD MODELS

In this section, we first introduce the proposed randomized smoothing techniques for general semantic transformations. Next, we categorize the semantic transformations into three types: composable, additive, and indirectly composable transformations. We then derive the smoothing-based certification strategies for each type.

[15]Without loss of generality, we consider untargeted attacks here, and the targeted attack can be derived similarly.

A.3.1 Transformation Specific Smoothed Classifier

We apply transformation-specific smoothing to an arbitrary base classifier $h : \mathcal{X} \to \mathcal{Y}$ to construct a smoothed classifier. Specifically, the smoothed classifier g predicts the class with the highest conditional probability when the input x is perturbed by some random transformations.

Definition A.1 (Transformation Specific Smoothed Classifier)**.** Let $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ be a semantic transformation. Let ϵ be a random variable in the parameter space $\mathcal{Z}$. Suppose we have a base classifier that learns a conditional probability distribution, $h(\boldsymbol{x}) = \arg\max_{y \in \mathcal{Y}} p(y|\boldsymbol{x})$. Applying transformation specific smoothing to the base classifier h yields a smoothed classifier $g : \mathcal{X} \to \mathcal{Z}$, which predicts

$$g(\boldsymbol{x}; \epsilon) = \arg\max_{y \in \mathcal{Y}} q(y|\boldsymbol{x}, \epsilon) = \arg\max_{y \in \mathcal{Y}} \mathbb{E}_{\epsilon}(p(y|\phi(\boldsymbol{x}, \epsilon))). \tag{A.2}$$

We recall the theorem proved in Chapter 7, which provides a generic certification bound for the transformation-specific smoothed classifier based on the Neyman-Pearson lemma [276].

Next, we will categorize the semantic transformations into different categories based on their intrinsic properties as shown in Figure A.3, and we will then discuss the certification principles for each specific category.

A.3.2 Composable Transformations

A set of semantic transformations is called composable if it is closed under composition.

Definition A.2. A set of semantic transformations defined by $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ is called **composable** if for any $\alpha \in \mathcal{Z}$ there exists an injective and continuously differentiable function $\gamma_\alpha : \mathcal{Z} \to \mathcal{Z}$ with non-vanishing Jacobian, such that

$$\phi(\phi(\boldsymbol{x}, \alpha), \beta) = \phi(\boldsymbol{x}, \gamma_\alpha(\beta)), \ \forall \boldsymbol{x} \in \mathcal{X}, \beta \in \mathcal{Z}. \tag{A.3}$$

The definition of composable transformations is the same as resolvable transformations in Chapter 7, but applied to point cloud data. Common semantic transformations for point cloud data that are composable include rotation, shearing along a fixed axis, and twisting along a fixed axis. For example, according to Euler's rotation theorem, we can always find another rotation $\gamma_\alpha(\beta) \in \mathcal{Z}$ for any two rotations $\alpha, \beta \in \mathcal{Z}$. Therefore, rotations belong to the composable transformations as shown in Figure A.2 (b).

In general, composable transformations can be certified against by Corollary 7.1 stated in Chapter 7. For a classifier $g(\boldsymbol{x}; \epsilon_0)$ smoothed by the composable transformation, we can

simply replace the random variable ϵ_1 by $\gamma_\alpha(\epsilon)$ in Corollary 7.1 to derive a robustness certification condition. However, some composable transformations with complicated $\gamma_\alpha(\beta)$ function result in intractable distribution for ϵ_1, causing difficulties for the certification. Therefore, we focus on a subset of composable transformations, called *additive transformations*, for which it is straight-forward to certify by applying Corollary 7.1.

A.3.3 Additive Transformations

We are particularly interested in a subset of composable transformations that the function $\gamma_\alpha : \mathcal{Z} \to \mathcal{Z}$ defined in Definition A.2 satisfies $\gamma_\alpha(\beta) = \alpha + \beta$ as shown in Figure A.2 (a) where the one step rotation above is equivalent to the two step transformations below.

Definition A.3. A set of semantic transformations $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ is called **additive** if

$$\phi(\phi(\boldsymbol{x}, \alpha), \beta) = \phi(\boldsymbol{x}, \alpha + \beta), \ \forall \ \boldsymbol{x} \in \mathcal{X}, \alpha, \beta \in \mathcal{Z}. \tag{A.4}$$

An additive transformation must be composable, but the reverse direction does not hold. For instance, the set of general rotations from the SO(3) group is composable, but not additive. Rotating 10° along the x axis first and then 10° along the y axis does not equal rotating 20° along the xy axis. Thus, general rotations cannot be categorized as an additive transformation. However, rotating along any fixed axis is additive. Based on this observation, we discuss z-rotation (i.e., rotation along z axis) and general rotations separately in Appendix A.4. All additive transformations can be certified following the same protocol derived from Corollary 7.1.

A.3.4 Indirectly Composable Transformations

As shown in Appendix A.3.2, composable transformations can be certified following Corollary 7.1. However, some semantic transformations of point clouds do not have such closure property under composition and thus do not fall in this category as shown in Figure A.2 (c). For example, the tapering transformation which we will discuss in Appendix A.4 is not composable and cannot be certified directly using Corollary 7.1. This kind of transformation is therefore categorized as a more general class called indirectly composable transformations.

Definition A.4. A set of transformations $\phi : \mathcal{X} \times \mathcal{Z}_\phi \to \mathcal{X}$ is **indirectly composable** if there is a set of composable transformations $\psi : \mathcal{X} \times \mathcal{Z}_\psi \to \mathcal{X}$, such that for any $x \in \mathcal{X}$, there exists a function $\delta_x : \mathcal{Z}_\phi \times \mathcal{Z}_\phi \to \mathcal{Z}_\psi$ with

$$\psi(\phi(\boldsymbol{x}, \alpha), \delta_{\boldsymbol{x}}(\alpha, \beta)) = \phi(\boldsymbol{x}, \beta), \ \forall \alpha, \beta \in \mathcal{Z}_\phi. \tag{A.5}$$

The definition is the same as differentially resolvable transformation in Chapter 7, but is applied to point cloud data. This definition involves more kinds of transformations, since we can choose the transformation ψ as $\psi(\boldsymbol{x}, \delta) = \boldsymbol{x} + \delta$ and let $\delta_{\boldsymbol{x}}(\alpha, \beta) = \phi(\boldsymbol{x}, \beta) - \phi(\boldsymbol{x}, \alpha)$. This specific assignment of ψ leads to a useful theorem Corollary 7.2, which we use to certify against some more complicated transformations, such as tapering in Appendix A.4. The theorem states that the overall robustness can be guaranteed if we draw multiple samples within the parameter space and certify the neighboring distribution of each sampled parameter separately.

A.4 CERTIFYING POINT CLOUD MODELS AGAINST SPECIFIC SEMANTIC TRANSFORMATIONS

In this section, we certify the point cloud models against several specific semantic transformations that are commonly seen for point cloud data, including rotation, shearing, twisting, and tapering. We do not analyze scaling and translation, since the point cloud models are usually inherently invariant to them due to the standard pre-processing pipeline [301]. For each transformation, we specify a corresponding certification protocol based on the categorization they belong to introduced in Appendix A.3.

A.4.1 Rotation, Shearing, and Twisting along a Fixed Axis

Rotation, shearing, and twisting are all common 3D transformations that are performed pointwise on point clouds. Without loss of generality, we consider performing these transformations along the z-axis.

Specifically, we define **z-shear** transformation as $\phi_{Sz} : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ where $\mathcal{X} = \mathbb{R}^{N \times 3}$ is the space of the point clouds with N points and $\mathcal{Z} = \mathbb{R}^2$ is the parameter space. For any $z = (\theta_1, \theta_2)$, z-shear acting on a point cloud $x \in \mathcal{X}$ with $\boldsymbol{x} = \{\boldsymbol{p}_i\}_{i=1}^N$ yields ($\boldsymbol{p}_i = (x_i, y_i, z_i)^T$)

$$\phi_{Sz}(\boldsymbol{p}_i, z) = (x_i + \theta_1 z_i, y_i + \theta_2 z_i, z_i). \tag{A.6}$$

Z-twist transformation $\phi_{Tz} : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ is defined similarly but with parameter space

$\mathcal{Z} = \mathbb{R}$. For any $\theta \in Z$ and $\boldsymbol{p}_i = (x_i, y_i, z_i)^T$,

$$\phi_{Tz}(\boldsymbol{p}_i, \theta) = \begin{pmatrix} x_i \cos(\theta z_i) - y_i \sin(\theta z_i) \\ x_i \sin(\theta z_i) + y_i \cos(\theta z_i) \\ z_i \end{pmatrix}. \tag{A.7}$$

Note that z-rotation, z-shear and z-twist are all *additive* transformations. Hence, we present the following corollary based on Corollary 7.1, which certifies the robustness of point clouds models with bounded ℓ_2 norm for the transformation parameters.

Corollary A.1. Suppose a classifier $g : \mathcal{X} \to \mathcal{Y}$ is smoothed by a transformation $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ with $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$. Assume its class probability satisfies condition (7.4). If the transformation is z-rotation, z-shear, or z-twist ($\phi = \phi_{Sz}, \phi_{Tz}$ or ϕ_{Rot-z}), then it is guaranteed that $g(\phi(\boldsymbol{x}, \alpha); \epsilon) = g(\boldsymbol{x}; \epsilon)$, if the following condition holds:

$$\|\alpha\|_2 \leqslant \frac{\sigma}{2}\Big(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\Big), \alpha \in \mathcal{Z}. \tag{A.8}$$

A.4.2 Tapering along a Fixed Axis

Tapering a point keeps the coordinate of a specific axis k, but scales the coordinates of other axes proportional to k's coordinate. For clarity, we define **z-taper** transformation $\phi_{TP} : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ as tapering along the z-axis, with its parameter space defined by $\mathcal{Z} = \mathbb{R}$. For any point cloud $x = \{\boldsymbol{p}_i\}_{i=1}^N \in \mathcal{X}$ ($\boldsymbol{p}_i = (x_i, y_i, z_i)$) and for any $\theta \in \mathcal{Z}$,

$$\phi_{TP}(\boldsymbol{p}_i, \theta) = \big(x_i(1 + \theta z_i), y_i(1 + \theta z_i), z_i\big). \tag{A.9}$$

However, z-taper is not a composable transformation, since the composition of two z-taper transformations contains terms with z_i^2 component. Therefore, we propose a specific certification protocol for z-taper based on Corollary 7.2. To achieve this goal, we specify a sampling strategy in the parameter space $\mathcal{Z}$ and bound the interpolation error of the sampled z-taper transformations.

Theorem A.1. Let $\phi_{TP} : \mathcal{X} \times \mathbb{R} \to \mathcal{X}$ be a z-taper transformation. Let $g : \mathcal{X} \to \mathcal{Y}$ be a ϵ-smoothed classifier with random noises $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_{3\times N})$, which predicts $g(\boldsymbol{x}; \epsilon) = \arg\max_y q(y|\boldsymbol{x}; \epsilon) = \arg\max \mathbb{E}_\epsilon p(y|\boldsymbol{x} + \epsilon)$. Let $\{\theta_j\}_{j=0}^M$ be a set of transformation parameters and $\theta_j = (\frac{2j}{M} - 1)R$. Suppose for any i,

$$q(y_A|\phi_{TP}(\boldsymbol{x}, \theta_j); \epsilon) \geqslant p_A^{(j)} > p_B^{(j)} \geqslant \max_{y \neq y_A} q(y|\phi_{TP}(\boldsymbol{x}, \theta_j); \epsilon) \tag{A.10}$$

Then it is guaranteed that $\forall\theta \in [-R, R]$: $y_A = \arg\max_y q(y|\phi_{TP}(\boldsymbol{x}, \theta); \epsilon)$ if for all $j = 1, \ldots, M$,

$$\frac{\sigma}{2}\left(\Phi^{-1}\left(p_A^{(j)}\right) - \Phi^{-1}\left(p_B^{(j)}\right)\right) \geqslant \frac{R\sqrt{N}}{2M}. \tag{A.11}$$

Detailed proof for Theorem A.1 can be found in [75].

A.4.3 General Rotation

Rotation is one of the most common transformations for point cloud data. Therefore, we hope the classifier is robust not only against rotation attacks along a fixed axis, but also those along arbitrary axes. In this section, we first define general rotation and show its universality for rotations as well as their composition; and then provide a concrete certification protocol for smoothing and certifying the robustness against this type of transformation.

We define **general rotation** transformations as $\phi_R : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ where $\mathcal{Z} = S^2 \times \mathbb{R}^+$ is the parameter space of rotations. For a rotation $z \in \mathcal{Z}$, its rotation axis is defined by a unit vector $k \in S^2$ and its rotation angle is $\theta \in \mathbb{R}^+$. For any 3D point $\boldsymbol{p}_i \in \mathbb{R}^3$,

$$\phi_R(\boldsymbol{p}_i, z) = \mathrm{Rot}(k, \theta)\boldsymbol{p}_i, \ z = (k, \theta). \tag{A.12}$$

where $\mathrm{Rot}(k, \theta)$ is the rotation matrix that rotates by θ along axis k. General rotations are *composable* transformations since the composition of any two 3D rotations can be expressed by another 3D rotation.

However, certifying against the general rotation is more challenging, since the general rotation is not additive and the expression of their composition is extremely complicated. In particular, if we smooth a base classifier with a random variable ϵ_0, a semantic attack with parameter $\alpha \in \mathcal{Z}$ results in $\phi_R(\phi_R(\boldsymbol{x}, \alpha), \epsilon_0) = \gamma_\alpha(\epsilon_0)$, which is a bizarre distribution in the parameter space. Therefore, we cannot directly apply Corollary 7.1 to certify general rotation.

On the other hand, as Corollary 7.2 shows, if we uniformly sample many parameters in a subspace of $\mathcal{Z} = S^2 \times \mathbb{R}^+$ and certify robustness in the neighborhood of each sample, we are able to certify a large and continuous subspace $\mathcal{Z}_{\text{robust}} \subseteq \mathcal{Z}$. As a result, we propose a sampling-based certification strategy, together with a tight bound for the interpolation error of general rotation transformations, which we summarize in the following theorem.

Theorem A.2. Let $\phi_R : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ be a general rotation transformation. Let $g : \mathcal{X} \to \mathcal{Y}$ be a classifier smoothed by random noises $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_{3\times N})$, which predicts $g(\boldsymbol{x}; \epsilon) = \arg\max_y q(y|\boldsymbol{x}; \epsilon) = \arg\max_y \mathbb{E}(p(y|\boldsymbol{x} + \epsilon))$. Let $\{z_j\}_{j=1}^M$ be a set of transformation parame-

ters with $z_j = (k_j, \theta_j), k_j \in S^2, \theta_j \in \mathbb{R}^+$ such that

$$\forall k \in S^2, \theta \in [0, R], \exists k_j, \theta_j \text{ s.t. } \langle k, k_j \rangle \leqslant \epsilon, |\theta - \theta_j| \leqslant \delta \tag{A.13}$$

Suppose for any j, the smoothed classifier g has class probabilities that satisfy

$$q(y_A | \phi_R(x, z_j); \epsilon) \geqslant p_A^{(j)} > p_B^{(j)} \geqslant \max_{y \neq y_A} q(y | \phi_R(x, z_j); \epsilon). \tag{A.14}$$

Then it is guaranteed that for any z with rotation angle $\theta < R$: $y_A = \arg\max_y q(y | \phi_R(\boldsymbol{x}, z); \epsilon)$ if $\forall j$,

$$\frac{\sigma}{2} \left(\Phi^{-1} \left(p_A^{(j)} \right) - \Phi^{-1} \left(p_B^{(j)} \right) \right) \geqslant \pi \sqrt{\frac{\delta^2}{4} + \frac{\epsilon^2 R^2}{8}} \|\boldsymbol{x}\|_2. \tag{A.15}$$

We present a proof sketch here and leave the details in [75]. Notice that the interpolation error between two transformations on a point cloud $\boldsymbol{x} = \{\boldsymbol{p}_i\}_{i=1}^N$ can be calculated by $\|\phi(\boldsymbol{x}, z_j) - \phi(\boldsymbol{x}, z)\|_2 = \|\phi(\boldsymbol{x}, z') - x\|_2 \leqslant \theta' (\sum_i^N \|\boldsymbol{p}_i\|_2^2)^{1/2}$, where $z' = (k', \theta')$ is the composition of the rotation with parameter z and the reverse rotation z_j^{-1}. Combined with the generic theorem for indirectly composable transformations (Corollary 7.2), bounding θ' using Equation (A.13) yields Theorem A.2.

A.4.4 Linear Transformations

Here, we consider a broader class of semantic transformations that contains all linear transformations applied to a 3D point. Formally, a **linear** transformation $\phi_L : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ has a parameter space of $\mathcal{Z} = \mathbb{R}^{3 \times 3}$. For any point cloud $\boldsymbol{x} = \{\boldsymbol{p}_i\}_{i=1}^N \in \mathcal{X}$ and for any $\mathbf{A} \in \mathcal{Z}$,

$$\phi_L(\boldsymbol{p}_i, \mathbf{A}) = (\mathbf{I} + \mathbf{A}) p_i. \tag{A.16}$$

Equation (A.16) describes any linear transformation with a bounded perturbation $\mathbf{A}$ from the identity transformation $\mathbf{I}$. A natural threat model is considered by [309] that the perturbation matrix $\mathbf{A}$ has a bounded Frobenius norm $\|\mathbf{A}\|_F \leqslant \epsilon$, for which we present a certification protocol in this paper. Linear transformations are composable because their compositions are also linear, but the fact that they are not additive prohibits a direct usage of Corollary 7.1. Nevertheless, these transformations can still be certified with a more complicated protocol, if Gaussian smoothing is applied.

Theorem A.3. Suppose a classifier g is smoothed by random linear transformations $\phi_L : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ where $\mathcal{Z} = \mathbb{R}^{3 \times 3}$, with a Gaussian random variable $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_9)$. If the class

probability satisfies condition (7.4), then it is guaranteed that $g(\phi((\boldsymbol{x}, \alpha); \epsilon) = g(\boldsymbol{x}; \epsilon)$ for all $\|\alpha\|_F \leqslant R$, where

$$R = \frac{\sigma\Big(\Phi^{-1}(\tilde{p}_A) - \Phi^{-1}(1 - \tilde{p}_A)\Big)}{2 + \sigma\Big(\Phi^{-1}(\tilde{p}_A) - \Phi^{-1}(1 - \tilde{p}_A)\Big)}. \tag{A.17}$$

$\tilde{p}_A$ is a function of p_A as explained in [75, Lemma B.2]

The theorem is proved in [75, Appendix B.4]

A.4.5 Compositions of Different Transformations

In addition to certifying against a single transformation, we also provide certification protocols for composite transformations, including z-twist ∘ z-rotation, z-taper ∘ z-rotation and z-twist ∘ z-taper ∘ z-rotation.

Notice that z-twist ∘ z-rotation is an additive function:

$$\phi_{Tz}(\phi_{Rot-z}(\phi_{Tz}(\phi_{Rot-z}(\boldsymbol{x}, \theta_1), \alpha_1), \theta_2), \alpha_2) = \phi_{Tz}(\phi_{Rot-z}(\boldsymbol{x}, \theta_1 + \theta_2), \alpha_1 + \alpha_2). \tag{A.18}$$

Therefore, we directly apply Corollary 7.1 to certify z-twist ∘ z-rotation transformation. The concrete corollary is stated as below.

Corollary A.2. Suppose a classifier $g : \mathcal{X} \to \mathcal{Y}$ is smoothed by random transformations z-twist ∘ z-rotation $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ where the parameter space $\mathcal{Z} = \mathcal{Z}_{Twist} \times \mathcal{Z}_{Rot-z} = \mathbb{R}^2$. The random variable for smoothing is $\epsilon \sim \mathcal{N}(0, \mathrm{diag}(\sigma_1^2, \sigma_2^2))$. If the class probability of g satisfies condition (7.4), then it is guaranteed that $g(\phi(\boldsymbol{x}, \alpha); \epsilon) = g(\boldsymbol{x}; \epsilon)$ for all $(\alpha_1, \alpha_2) \in \mathcal{Z}$, if the following condition holds:

$$\sqrt{\left(\frac{\alpha_1}{\sigma_1}\right)^2 + \left(\frac{\alpha_2}{\sigma_2}\right)^2} \leqslant \frac{\sigma}{2}\Big(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\Big). \tag{A.19}$$

Another composite transformation z-taper ∘ z-rotation first rotates the point cloud along z-axis, and then taper along z-axis. As z-taper is not composable with itself, this composite transformation is also not composable. Similar to z-taper, we certify the composite transformation z-taper ∘ z-rotation by upper-bounding the interpolation error.

Theorem A.4. We denote z-taper ∘ z-rotation by $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}, \phi = \phi_{TP} \circ \phi_{Rot-z}$ with a parameter space of $\mathcal{Z} = \mathcal{Z}_{TP} \times \mathcal{Z}_{Rot-z} = \mathbb{R}^2$. Let $g : \mathcal{X} \to \mathcal{Y}$ be a classifier smoothed by random noises $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_{3\times N})$.

For a subspace in the parameter space, $S = [-\varphi, \varphi] \times [-\theta, \theta] \subseteq \mathcal{Z}$, we uniformly sample $\varphi\theta M^2$ parameters $\{z_{jk}\}$ in S. That is, $z_{jk} = (\varphi_j, \theta_k)$ where $\varphi_j = \frac{2j}{M} - \varphi$ and $\theta_k = \frac{2k}{M} - \theta$.

Suppose for any j, k the smoothed classifier g has class probability that satisfy

$$q(y_A|\phi(\boldsymbol{x}, z_{jk}); \epsilon) \geqslant p_A^{(jk)} > p_B^{(jk)} \geqslant \max_{y \neq y_A} q(y|\phi(\boldsymbol{x}, z_{jk}); \epsilon), \tag{A.20}$$

then it is guaranteed that $y_A = \arg\max_y q(y|\phi(\boldsymbol{x}, z); \epsilon)$ if $\forall j, k$ and $\forall z \in S$,

$$\frac{\sigma}{2}\left(\Phi^{-1}\left(p_A^{(jk)}\right) - \Phi^{-1}\left(p_B^{(jk)}\right)\right) \geqslant \frac{\sqrt{N(4\varphi^2 + 8\varphi + 5)}}{2M}. \tag{A.21}$$

We also consider the composition of three transformations: z-twist ∘ z-taper ∘ z-rotation.

Theorem A.5. We define the composite transformation z-twist ∘ z-taper ∘ z-rotation by $\phi : \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$, with input space $\mathcal{X} = \mathbb{R}^{3\times N}$ and parameter space $\mathcal{Z} = \mathcal{Z}_{Twist} \times \mathcal{Z}_{Taper} \times \mathcal{Z}_{Rot-z} = \mathbb{R}^3$. Let $g : \mathcal{X} \to \mathcal{Y}$ be a classifier smoothed by random noises $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_{3\times N})$, which predicts $g(\boldsymbol{x}; \epsilon) = \arg\max_y q(y|\boldsymbol{x}; \epsilon) = \arg\max_y \mathbb{E}(p(y|\boldsymbol{x} + \epsilon))$.

Let $\{z_{jkl} \in \mathcal{Z} : z = (\varphi_j, \alpha_k, \theta_l)\}$ be a set of parameters with $\varphi_j = \frac{2j}{M} - \varphi$, $\alpha_k = \frac{2k}{M} - \alpha$ and $\theta_l = \frac{2l}{M} - \theta$. Therefore $(\varphi_j, \alpha_k, \theta_l)$ distribute uniformly in the subspace $\mathcal{Z}_{robust} = [-\varphi, \varphi] \times [-\alpha, \alpha] \times [-\theta, \theta] \subseteq \mathcal{Z}$. Suppose for any j, k, l, the smoothed classifier g has class probability that satisfy

$$q(y_A|\phi(\boldsymbol{x}, z_{jkl}); \epsilon) \geqslant p_A^{(jkl)} > p_B^{(jkl)} \geqslant \max_{y \neq y_A} q(y|\phi(\boldsymbol{x}, z_{jkl}); \epsilon), \tag{A.22}$$

then it is guaranteed that for any $z \in \mathcal{Z}_{robust}$: $y_A = \arg\max_y q(y|\phi(\boldsymbol{x}, z); \epsilon)$, if for any i, j, k,

$$\frac{\sigma}{2}\Big(\Phi^{-1}\Big(p_A^{(jkl)}\Big) - \Phi^{-1}\Big(p_B^{(jkl)}\Big)\Big) \geqslant \frac{\sqrt{N(1 + \frac{27}{4}(1+\alpha)^2)}}{2M} \tag{A.23}$$

Both Theorem A.4 and Theorem A.5 are based on our proposed approach of sampling parameters in the parameter space and certifying the neighboring distributions of the samples separately by bounding the interpolation error. These two theorems are rigorously proved in [75, Appendix B.5 and B.6].

A.5 EXPERIMENTS

We conduct extensive experiments on different 3D semantic transformations and models to evaluate the certified robustness derived from our TPC framework. We show that TPC significantly outperforms the state-of-the-art in terms of the certified robustness against a range of semantic transformations, and the results also lead to some interesting findings.

A.5.1 Experimental setup

Dataset. We perform experiments on the ModelNet40 dataset [411], which includes different 3D objects of 40 categories. We follow the standard pre-processing pipeline that places the point clouds in the center and scales them into a unit sphere.

We also conduct experiments for part segmentation tasks, for which the ShapeNet dataset [58] is used for evaluation. It contains 16681 meshes from 16 categories and also 50 predefined part labels. The experiment results are presented in Appendix A.5.2.

Models. We run our experiments for point cloud classification on PointNet models [301] with different point cloud sizes. We apply data augmentation training for each transformation combined with consistency regularization to train base classifiers. We then employ our TPC framework to smooth these models and derive robustness certification bounds against various transformations. TPC does not depend on specific model selection and can be directly applied to certify other point cloud model architectures. We present certification results for other architectures (e.g., CurveNet [415]) in [75].

Evaluation Metrics. To evaluate the robustness of point clouds classification, we pick a fixed random subset of the ModelNet40 test dataset. We report the **certified accuracy** defined by the fraction of point clouds that are classified both *correctly* and *consistently* within certain transformation space. The baseline we compare with [241] only presents **certified ratio**, which is the fraction of test samples classified *consistently*. We believe that the *certified accuracy* is a more rigorous metric for evaluation based on existing standard certification protocols in the image domain [77]. We thus calculate the certified accuracy for baselines based on the results reported in the paper [241] for comparison. Besides, we also report the certified ratio comparison in Table A.2 and the full version of this appendix [75]. We remark that TPC provides a probabilistic certification for point cloud models and we use a high confidence level of 99.9% in all experiments; while the baseline DeepG3D [241] yields a deterministic robustness certification.

For the part segmentation task, we evaluate our method using a fixed random subset of the ShapeNet test dataset. As the part segmentation task requires assigning a part category to each point in a point cloud, we report the **point-wise certified accuracy** defined as the fraction of points that are classified *correctly* and *consistently*. Note that other common metrics such as IoU can be easily derived based on our bound as well. We will focus on the point-wise certified accuracy for the convenience of comparison with baseline [241].

Table A.1: Comparison of certified accuracy achieved by our transformation-specific smoothing framework TPC and the baseline, DeepG3D [241]. "-" denotes the settings where the baselines cannot scale up to.

Transformation	Attack radius	Certified Accuracy (%)	
		TPC	DeepG3D
ZYX-rotation	2°	**81.4**	61.6
	5°	**69.2**	49.6
General rotation	5°	**78.5**	-
	10°	**69.2**	-
	15°	**55.5**	-
Z-rotation	20°	**84.2**	81.8
	60°	**83.8**	81.0
	180°	**81.3**	-
Z-shear	0.03	**83.4**	59.8
	0.1	**82.2**	-
	0.2	**77.7**	-
Z-twist	20°	**83.8**	20.3
	60°	**80.1**	-
	180°	**64.3**	-
Z-taper	0.1	**78.1**	69.0
	0.2	**76.5**	23.9
	0.5	**66.0**	-
Linear	0.1	**74.0**	-
	0.2	**59.9**	-
Z-twist ∘ Z-rotation	20°, 1°	**78.9**	13.8
	20°, 5°	**78.5**	-
	50°, 5°	**76.9**	-
Z-taper ∘ Z-rotation	0.1, 1°	**76.1**	58.2
	0.2, 1°	**72.9**	17.5
Z-twist ∘ Z-taper ∘ Z-rotation	10°, 0.1, 1°	**68.8**	17.5
	20°, 0.2, 1°	**63.1**	4.6

A.5.2 Main Results

In this section, we present our main experimental results. Concretely, we show that: (1) the certified accuracy of TPC under a range of semantic transformations is significantly higher than the baseline, and TPC is able to certify under some transformation space where the baseline cannot be applied; (2) the certified accuracy of TPC always outperforms the baseline for different point cloud sizes, and more interestingly, the certified accuracy of TPC increases with the increasing of point cloud size while that of the baseline decreases due to relaxation; (3) TPC is also capable of certifying against ℓ_2 or ℓ_∞ norm bounded 3D perturbations for different point clouds sizes; (4) on the part segmentation task, TPC still outperforms the baseline against different semantic transformations and is able to certify some transformation parameter space that the baseline is not applicable.

Table A.2: Comparison of certified ratio as well as certified accuracy for z-rotation transformations. "-" denotes the settings which the baselines cannot scale up to.

Radius	Certified Ratio (%)		Certified Accuracy (%)	
	TPC	DeepG3D	TPC	DeepG3D
20°	**99.0**	96.7	**84.2**	81.8
60°	**98.1**	95.7	**83.8**	81.0
180°	**95.2**	-	**81.3**	-

Comparison of Certified Accuracy

Table A.1 shows the certified accuracy we achieved for different transformations compared with prior works. We train a PointNet model with 64 points, which is consistent with the baseline. For transformations characterized by one parameter, such as z-rotation, z-twist, and z-taper, we report the certified accuracy against attacks in $\pm\theta$. For z-shear with a parameter space of $\mathbb{R}^2$, we report the certified accuracy against attacks in a certain ℓ_2 parameter radius. For the linear transformation with a parameter space of $\mathbb{R}^{3\times3}$, we report the certified accuracy against attacks in a certain Frobenius norm radius.

The highlighted results in Table A.1 demonstrate that our framework TPC significantly outperforms the state of the art in every known semantic transformation. For example, we improve the certified accuracy from 59.8% to 83.4% for z-shear in ±0.03 and from 20.3% to 83.8% for z-twist in $\pm20°$.

Besides, we also report the certified accuracy for larger attack radius for which the baseline cannot certify (cells with "-"). For instance, we achieve 81.3% certified accuracy on z-rotation within $\pm180°$, which is essentially every possible z-rotation transformation.

The general rotation transformations we define in Appendix A.4.3 includes rotations along any axis with bounded angles. Lorenz et al. [241] consider *ZYX-rotation*, the composition of three rotations within $\pm\theta$ (Euler angles) along x, y, z axes instead, which results in a different geometric shape for the certified parameter space. However, the parameter space restricted by $S^2 \times [0, 2\theta]$ of general rotation strictly contains the space defined by $\pm\theta$ for three Euler angles. The derived results for ZYX-rotation are also shown in Table A.1 for comparison.

Comparison of Certified Ratio. Aside from the certified accuracy, we also consider the certified ratio as another metric according to the baseline. This metric measures the tightness of certification bounds but fails to take the classification accuracy into account which is important. Therefore, we mainly present the comparison based on the certified ratio for z-rotations in Table A.2 only for comparison and leave the full comparison in [75].

Table A.3: Certification of z-rotation for different point cloud sizes. The certified accuracy achieved by our TPC increases as the size of the point cloud model increases.

(a) $\theta = \pm 3^\circ$ compared with DeepG3D [241]

Points	16	32	64	128	256	512	1024
TPC	**83.2**	**83.8**	**86.6**	**87.4**	**89.4**	**89.8**	**90.5**
DeepG3D	75.4	78.4	79.1	69.4	57.5	42.8	32.3

(b) Certified accuracy of TPC under $\theta = \pm 180^\circ$

Points	16	32	64	128	256	512	1024
TPC	73.6	79.3	81.3	81.8	83.0	84.6	83.8

Certification on Point Clouds with Different Sizes

Here we show that our certification framework naturally scales up to larger point cloud models. A basic principle of our TPC framework is that deriving the certification bound for a smoothed classifier only depends on the predicted class probability. In other words, it does not rely on specific model architectures.

The relaxation-based verifiers [241, 343] have worse certification guarantees for larger point clouds due to the precision loss during relaxation, especially for pooling layers that are heavily used in point cloud model architectures. For example, the DeepG3D verifier guarantees 79.1% certified accuracy for a 64-point model on z-rotation with $\pm 3^\circ$ (without splitting); but the certification drops to 32.3% for a 1024-point model [241]. In contrast, using our TPC framework, the certified accuracy tends to **increase** with a larger number of points in point clouds. This is because larger PointNet models predict more accurately and yield higher class probability after smoothing. We compare our TPC framework with the baseline in terms of certified accuracy for different point cloud sizes in Table A.3(a). The baseline DeepG3D only presents results for z-rotations in $\theta = \pm 3^\circ$, which cannot fully illustrate the capability of our method. Therefore, we also report our experimental results for z-rotations in $\theta = \pm 180^\circ$ in Table A.3(b). It shows that our method can scale up to larger point cloud models to accommodate real-world scenarios.

Certification against ℓ_p-bounded 3D-Perturbations

In addition to semantic transformations, we also provide robustness certification for point cloud models against ℓ_p perturbations. Defenses have been proposed for point cloud models against ℓ_2 norm bounded perturbations [120], which is a special case of our TPC framework regarding an additive transformation $\phi(x, z) = x + z$.

We cannot directly certify against perturbations with bounded ℓ_∞ norm. However, a certification bound similar to the baseline [241] can still be derived, using the loose inequality $\|\theta\|_\infty \leqslant \sqrt{3N}\|\theta\|_2$. Here, we exhibit the certified accuracy for bounded ℓ_2 norm in Table A.4 as a benchmark; and omit more details for ℓ_∞ norm to [75]. We show that under the ℓ_2 norm bounded 3D-perturbations, TPC certifies even better as the point clouds sizes increase and

Table A.4: Certified accuracy of TPC for point cloud models under ℓ_2 attacks. The certified accuracy increases as the size of point cloud models increases.

Attack	Radius	Certified Accuracy (%)		
		16	64	256
ℓ_2	0.05	74.1	82.2	84.2
ℓ_2	0.1	61.9	70.8	77.3

Table A.5: Comparison of point-wise certified accuracy for the *part segmentation* task. "-" denotes the settings that the baseline does not consider or cannot scale up to.

Transformation	Radius	Certified Accuracy (%)	
		TPC	DeepG3D
Z-rotation	5°	**87.8**	85.7
Z-rotation	10°	**86.1**	84.8
Z-rotation	180°	**70.8**	-
Z-shear	0.2	**86.1**	-
Z-twist	180°	**74.5**	-

achieve a high certified accuracy of 77.3% for perturbations with ℓ_2 norm bounded by 0.1.

Certification for Part Segmentation

Part segmentation is a common 3D recognition task in which a model is in charge of assigning each point or face of a 3D mesh to one of the predefined categories. As our TPC framework is independent of concrete model architectures, it can be naturally extended to handle this task.

We evaluate our method using the ShapeNet part dataset [58]. We train a segmentation version PointNet [301] with 64 points, which predicts a part category for each point in the point cloud. The certified accuracy reported in Table A.5 denotes the percentage of points guaranteed to be labeled correctly. We can see that for the part segmentation task, TPC consistently outperforms the baseline against different semantic transformations. The baseline only reports the result for z-rotations in $\pm 5^\circ$ and $\pm 10^\circ$, while we present robustness guarantees for any z-rotation ($\pm 180^\circ$) as well as other transformations including shearing and twisting.

A.6 SUMMARY

In this appendix, by extending from TSS, we propose a unified certification framework TPC for point cloud models against a diverse range of semantic transformations. Our the-

oretical and empirical analysis show that TPC is more scalable and able to provide much tighter certification under different settings and tasks.

APPENDIX B: APPENDIX FOR CHAPTER 2

B.1 EXTENSIONS AND PROOFS IN Section 2.4

In this appendix, we provide formal proofs and theoretical extensions for the results in Section 2.4.

B.1.1 Proof of Theorem 2.1

Proof of Theorem 2.1. We let D_c denote the decision region of F_0 for class c, i.e., $D_c := \{\boldsymbol{x} : F_0(\boldsymbol{x}) = c\}$. Since $\mathcal{Q}$ is supported on the decision region shifted by $\boldsymbol{x}_0$, $f_0^{\mathcal{Q}}(\boldsymbol{x}_0)_c = 1$. Thus, from $f_0^{\mathcal{Q}}(\boldsymbol{x}_0)_c$, we know $(\mathrm{supp}(\mathcal{Q}) + \boldsymbol{x}_0)\backslash S \subseteq D_c$, where S is some set with zero measure under $\mathcal{Q} + \boldsymbol{x}_0$. Since $0 < q(\boldsymbol{x})/p(\boldsymbol{x}) < +\infty$, S also has zero measure under $\mathcal{P} + \boldsymbol{x}_0$. On the other hand, by $0 < q(\boldsymbol{x})/p(\boldsymbol{x}) < +\infty$, we can determine the probability mass of $\mathrm{supp}(\mathcal{Q})$ on $\mathcal{P}$, i.e., $\Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[\boldsymbol{\epsilon} \in \mathrm{supp}(\mathcal{Q})]$. Then, we observe that

$$
\begin{aligned}
& f_0^{\mathcal{P}}(\boldsymbol{x}_0)_c = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = c] \\
= & \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[\boldsymbol{\epsilon} \in \mathbb{R}^d\backslash\mathrm{supp}(\mathcal{Q})] \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = c \,|\, \boldsymbol{\epsilon} \in \mathbb{R}^d\backslash\mathrm{supp}(\mathcal{Q})] \\
& + \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[\boldsymbol{\epsilon} \in \mathrm{supp}(\mathcal{Q})] \cdot \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = c \,|\, \boldsymbol{\epsilon} \in \mathrm{supp}(\mathcal{Q})] \qquad \text{(B.1)} \\
= & \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[\boldsymbol{\epsilon} \in \mathbb{R}^d\backslash\mathrm{supp}(\mathcal{Q})] \cdot \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = c \,|\, \boldsymbol{\epsilon} \in \mathbb{R}^d\backslash\mathrm{supp}(\mathcal{Q})] \\
& + \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[\boldsymbol{\epsilon} \in \mathrm{supp}(\mathcal{Q})].
\end{aligned}
$$

By the definition of $\mathrm{supp}(\mathcal{Q})$, we observe that $\Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = c \,|\, \boldsymbol{\epsilon} \in \mathbb{R}^d\backslash\mathrm{supp}(\mathcal{Q})] = 0$. As a result, we will find that $f_0^{\mathcal{P}}(\boldsymbol{x}_0)_c = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[\boldsymbol{\epsilon} \in \mathrm{supp}(\mathcal{Q})]$. Then the DSRS certification method can know $\Big(\big(\mathbb{R}^d\backslash\mathrm{supp}(\mathcal{Q})\big) + \boldsymbol{x}_0\Big) \cap D_c$ has zero measure under $\mathcal{P} + \boldsymbol{x}_0$, i.e., In summary, the certification method can determine that $\mathrm{supp}(\mathcal{Q}) + \boldsymbol{x}_0$ differs from D_c on some set Δ with zero measure under $\mathcal{P} + \boldsymbol{x}_0$. Because $\mathcal{P}$ has positive density everywhere, Δ also has zero measure under $\mathcal{P} + \boldsymbol{x}_0 + \boldsymbol{\delta}$ for arbitrary $\boldsymbol{\delta} \in \mathbb{R}^d$. Thus, for arbitrary $\boldsymbol{\delta} \in \mathbb{R}^d$, the certification method can compute out

$$
f_0^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta})_c = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[F_0(\boldsymbol{x}_0 + \boldsymbol{\delta} + \boldsymbol{\epsilon}) = c] = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}+\boldsymbol{\delta}}[\boldsymbol{x}_0 + \boldsymbol{\epsilon} \in D_c] = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}+\boldsymbol{\delta}}[\mathrm{supp}(\mathcal{Q})]. \qquad \text{(B.2)}
$$

Under the binary classification setting, it implies that for any $\boldsymbol{\delta} \in \mathbb{R}^d$, the $f_0^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta})$ can be uniquely determined by DSRS certification method. Since the smoothed classifier's decision

at any $\boldsymbol{x}_0 + \boldsymbol{\delta}$ is uniquely determiend by $f_0^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta})$ (see Equation (2.2)), the certification method can exactly know $\widetilde{F}_0^{\mathcal{P}}(\boldsymbol{x} + \boldsymbol{\delta})$ for any $\boldsymbol{\delta}$ and thus determine tightest possible certified robust radius r_{tight}. QED.

B.1.2 Extending Theorem 2.1 to Multiclass Setting

For the multiclass setting, we define a variant of DSRS as follows.

Definition B.1 ($r_{\mathsf{DSRS}}^{\mathsf{multi}}$)**.** Given $P_A \in [0,\, 1]$ and $Q_A^{\mathsf{multi}} \in \mathbb{R}^{C-1}$,

$$\begin{aligned} &r_{\mathsf{DSRS}}^{\mathsf{multi}} := \max r \quad \text{s.t.} \quad \forall F : \mathbb{R} \to [C], f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0} = P_A, f^{\mathcal{Q}_c}(\boldsymbol{x}_0)_c = (Q_A^{\mathsf{multi}})_c, c \in [C-1] \\ \Rightarrow &\forall \boldsymbol{x}, \|\boldsymbol{x} - \boldsymbol{x}_0\|_p < r, \widetilde{F}^{\mathcal{P}}(\boldsymbol{x}) = y_0. \end{aligned} \tag{B.3}$$

In the above definition, $r_{\mathsf{DSRS}}^{\mathsf{multi}}$ is the tightest possible certified radius with prediction probability Q_A^{multi}, where each component of Q_A^{multi}, namely $(Q_A^{\mathsf{multi}})_c$, corresponds to the prediction probability for label c under additional smoothing distribution $\mathcal{Q}_c$. Note that there are $(C-1)$ additional smoothing distributions $\{Q_c\}_{c=1}^{C-1}$ in this generalization for multiclass setting.

With this DSRS generation, the following corollary extends the tightness analysis in Theorem 2.1 from binary to the multiclass setting.

Corollary B.1. Suppose the original smoothing distribution $\mathcal{P}$ has positive density everywhere, i.e., $p(\cdot) > 0$. For multiclass classification with base classifier F_0, at point $\boldsymbol{x}_0 \in \mathbb{R}^d$, for each class $c \in [C-1]$, let $\mathcal{Q}_c$ be a distribution that satisfies: (1) its support is the decision region of c shifted by $\boldsymbol{x}_0$: $\mathrm{supp}(\mathcal{Q}_c) = \{\boldsymbol{x} - \boldsymbol{x}_0 : F_0(\boldsymbol{x}) = c\}$; (2) for any $\boldsymbol{x} \in \mathrm{supp}(\mathcal{Q})$, $0 < q_c(\boldsymbol{x})/p(\boldsymbol{x}) < +\infty$. Then, plugging $P_A = f_0^{\mathcal{P}}(\boldsymbol{x}_0)_c$ and Q_A^{multi} where $(Q_A^{\mathsf{multi}})_c = f_0^{\mathcal{Q}_c}(\boldsymbol{x}_0)_c$ for $c \in [C-1]$ into Definition B.1, we have $r_{\mathsf{DSRS}}^{\mathsf{multi}} = r_{\mathsf{tight}}$ under any ℓ_p ($p \geqslant 1$).

Proof of Corollary B.1. Similar as the proof of Theorem 2.1, the certification method can observe that for any $c \in [C-1]$, $f_0^{\mathcal{Q}_c}(\boldsymbol{x}_0)_c = 1$. Thus, the method knows $\mathrm{supp}(\mathcal{Q}_c) + \boldsymbol{x}_0 \approx D_c$ for arbitrary $c \in [C-1]$. Here, the "$\approx$" means that the difference between the two sets has zero measure under $\mathcal{P} + \boldsymbol{x}_0$. Thus, for arbitrary $\boldsymbol{\delta} \in \mathbb{R}^d$ the certification method can precisely compute out $f^{\mathcal{P}}(\boldsymbol{x}_0)_c$ for any $c \in [C-1]$. Since $f^{\mathcal{P}}(\boldsymbol{x}_0) \in \Delta^C$, we also know $f^{\mathcal{P}}(\boldsymbol{x}_0)_C$ and the smoothed classifier's prediction on $\boldsymbol{x}_0 + \boldsymbol{\delta}$ can be uniquely determined. Then, following the same argument in Theorem 2.1's proof, we can determine the tightest possible certified robust radius $r_{\mathsf{DSRS}}^{\mathsf{multi}}$. QED.

Remark B.1. To achieve the tightest possible certified radius r_{tight}, for binary classification, we only need one extra scalar as the additional information (Q_A), while for multiclass classification, we need $(C-1)$ extra scalars as the additional information ($Q_A^{\mathsf{multi}} \in \mathbb{R}^{C-1}$). Following the convention as discussed in Section 2.2, we are interested in tight certification under the binary classification for sampling efficiency concerns. Therefore, we focus on using only one extra scalar (Q_A) to additional information in DSRS. In both Theorem 2.1 and Corollary B.1, we only need finite quantities to achieve tight certification for *any* smoothed classifier. In contrast, other existing work requires infinite quantities to achieve such optimal tightness [267, Section 3.1].

B.1.3 Proof of Theorem 2.2

The proof of Theorem 2.2 is a bit complicated, which relies on several propositions and lemmas along with theoretical results in Section 2.5. At high level, based on the standard Gaussian distribution's property (Proposition B.1), we find $Q_A = 1$ under concentration property (Lemma B.1). With $Q_A = 1$, we derive a lower bound of r_{DSRS} in Lemma B.2. We then use: (1) the concentration of beta distribution $\mathrm{Beta}(\frac{d-1}{2}, \frac{d-1}{2})$ (see Lemma B.3) for large d; (2) the relative concentration of gamma $\Gamma(d/2, 1)$ distribution around mean for large d (see Proposition B.2 and resulting Fact B.1); and (3) the misalignment of gamma distribution $\Gamma(d/2-k, 1)$'s mean and median for small $(d/2-k)$ (see Proposition B.3) to lower bound the quantity in Lemma B.2 and show it is large or equal to 0.5. Then, using the conclusion in Section 2.5 we conclude that $r_{\mathsf{DSRS}} \geqslant 0.02\sigma\sqrt{d}$.

Proposition B.1. If random vector $\boldsymbol{\epsilon} \in \mathbb{R}^d$ follows standard Gaussian distribution $\mathcal{N}(\sigma)$, then

$$\Pr[\|\boldsymbol{\epsilon}\|_2 \leqslant T] = \Gamma\mathrm{CDF}_{d/2}\left(\frac{T^2}{2\sigma^2}\right), \tag{B.4}$$

where $\Gamma\mathrm{CDF}_{d/2}$ is the CDF of gamma distribution $\Gamma(d/2, 1)$.

Proof of Proposition B.1. According to [244, Eqn. 5.19.4], he volume of a d-dimensional ball, i.e., d-ball, with radius r is $V_d(r) = \frac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} r^d$. Thus,

$$\mathrm{Vol}(\{\boldsymbol{\epsilon} : \|\boldsymbol{\epsilon}\|_2 = r\}) = V_d(r)' = \frac{d\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} r^{d-1}. \tag{B.5}$$

$$\Pr[\|\boldsymbol{\epsilon}\|_2 \leqslant T]$$

$$= \int_0^T \frac{1}{(2\pi\sigma^2)^{d/2}} \cdot \exp\left(-\frac{r^2}{2\sigma^2}\right) \cdot \mathrm{Vol}(\{\boldsymbol{\epsilon} : \|\boldsymbol{\epsilon}\|_2 = r\})\,\mathrm{d}r \tag{B.6}$$

$$= \int_0^T \frac{1}{(2\pi\sigma^2)^{d/2}} \cdot \exp\left(-\frac{r^2}{2\sigma^2}\right) \cdot \frac{d\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} r^{d-1}\,\mathrm{d}r \tag{B.7}$$

$$= \int_0^{T^2} \frac{1}{(2\sigma^2)^{d/2}\Gamma(\frac{d}{2})} \exp\left(-\frac{r}{2\sigma^2}\right) r^{d/2-1}\,\mathrm{d}r \tag{B.8}$$

$$= \frac{1}{\Gamma(\frac{d}{2})} \int_0^{\frac{T^2}{2\sigma^2}} \exp(-r) r^{d/2-1}\,\mathrm{d}r = \Gamma\mathrm{CDF}_{d/2}\left(\frac{T^2}{2\sigma^2}\right). \tag{B.9}$$

QED.

With Proposition B.1, now we can show that $Q_A = 1$ under the condition of Theorem 2.2 as stated in the following lemma.

Lemma B.1. Suppose F_0 satisfies $(\sigma, P_{\mathsf{con}})$-concentration property at input point $\boldsymbol{x}_0 \in \mathbb{R}^d$, with additional smoothing distribution $\mathcal{Q} = \mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$ where $T^2 = 2\sigma^2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}})$ and $d/2 - 15 \leqslant k < d/2$, we have

$$Q_A = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = y_0] = 1. \tag{B.10}$$

Proof of Lemma B.1. According to Definition 2.3, for T' that satisfies

$$\Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}(\sigma)}[\|\boldsymbol{\epsilon}\|_2 \leqslant T'] = P_{\mathsf{con}} \tag{B.11}$$

we have

$$\Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}(\sigma)}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = y_0 \,|\, \|\boldsymbol{\epsilon}\|_2 \leqslant T'] = 1. \tag{B.12}$$

With Equation (B.11), from Proposition B.1, we have

$$T'^2 = 2\sigma^2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}}). \tag{B.13}$$

Thus, Equation (B.12) implies

$$\Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}(\sigma)}[F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = y_0 \,|\, \|\boldsymbol{\epsilon}\|_2 \leqslant \sqrt{2\sigma^2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}})}] = 1. \tag{B.14}$$

Notice that $\mathcal{N}(\sigma)$ has finite and positive density anywhere within $\{\boldsymbol{\epsilon} : \|\boldsymbol{\epsilon}\|_2 \leqslant T'\}$. Thus, $F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = y_0$ for any $\boldsymbol{\epsilon}$ with $\|\boldsymbol{\epsilon}\|_2 \leqslant T'$ unless a zero-measure set.

Now, we consider $\mathcal{Q}$. $\mathcal{Q} = \mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$ where $T = T'$, and $\mathcal{N}_{\mathsf{trunc}}$ has finite and positive

density anywhere within $\{\boldsymbol{\epsilon} : \|\boldsymbol{\epsilon}\|_2 \leqslant T'\}\backslash\{\mathbf{0}\}$. Thus,

$$Q_A = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k,T,\sigma)}[F_0(\boldsymbol{x}_0+\boldsymbol{\epsilon}) = y_0] \tag{B.15}$$

$$= \Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}^{\mathsf{g}}(k,\sigma)}[F_0(\boldsymbol{x}_0+\boldsymbol{\epsilon}) = y_0 \,|\, \|\boldsymbol{\epsilon}\|_2 \leqslant T] \tag{B.16}$$

$$\overset{(*)}{=} \Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}^{\mathsf{g}}(k,\sigma)}[F_0(\boldsymbol{x}_0+\boldsymbol{\epsilon}) = y_0 \,|\, \|\boldsymbol{\epsilon}\|_2 \leqslant T, \boldsymbol{\epsilon}\neq\mathbf{0}] = 1. \tag{B.17}$$

In the above equations, $(*)$ is because

$$\Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}^{\mathsf{g}}(k,\sigma)}[\boldsymbol{\epsilon}=\mathbf{0} \,|\, \|\boldsymbol{\epsilon}\|_2 \leqslant T] \leqslant \lim_{r\to 0} \Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}^{\mathsf{g}}(k,\sigma)}[\|\boldsymbol{\epsilon}\|_2 \leqslant r \,|\, \|\boldsymbol{\epsilon}\|_2 \leqslant T] \tag{B.18}$$

$$= \lim_{r\to 0} C\int_0^r x^{-2k}\exp\left(-\frac{x^2}{2\sigma'^2}\right) dx^{d-1}\,\mathrm{d}x \tag{B.19}$$

$$= C\lim_{r\to 0}\int_0^r dx^{d-2k-1}\exp\left(-\frac{x^2}{2\sigma'^2}\right)\mathrm{d}x = 0 \tag{B.20}$$

where C is a constant, and the last equality is due to $d/2 > k \Rightarrow d-2k-1 \geqslant 0$. QED.

With $Q_A = 1$, we can have a lower bound of r_{DSRS} as stated in the following lemma.

Lemma B.2. Under the same condition as in Lemma B.1, we let $\mathrm{BetaCDF}_{\frac{d-1}{2}}$ be the CDF of distributon $\mathrm{Beta}(\frac{d-1}{2}, \frac{d-1}{2})$, let

$$\begin{aligned} r_0 &= \max u \\ \text{s.t.} \mathop{\mathbb{E}}_{t\sim\Gamma(\frac{d}{2}-k)} &\mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2-(\sigma'\sqrt{2t}-u)^2}{4u\sigma'\sqrt{2t}}\right) \geqslant 0.5, \end{aligned} \tag{B.21}$$

and let r_{DSRS} be the tightest possible certified radius in DSRS under ℓ_2 when smoothing distribution $\mathcal{P} = \mathcal{N}^{\mathsf{g}}(k,\sigma)$, then

$$r_{\mathsf{DSRS}} \geqslant r_0. \tag{B.22}$$

Proof of Lemma B.2. The proof shares the same core methodology as DSRS computational method introduced in Section 2.5. Basically, according to Equation (2.10), for any radius r, let $\boldsymbol{\delta} = (r, 0, \ldots, 0)^{\intercal}$, if $\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) > 0.5$, then $r_{\mathsf{DSRS}} \geqslant r$, where $Q_A = 1$ according to Lemma B.1, and by the $(\sigma, P_{\mathsf{con}})$-concentration property

$$P_A \geqslant \Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}^{\mathsf{g}}(k,\sigma)}[\|\boldsymbol{\epsilon}\|_2 \leqslant T]. \tag{B.23}$$

Therefore, to prove the lemma, we only need to show that when

$$\mathop{\mathbb{E}}_{t\sim\Gamma(\frac{d}{2}-k)} \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2-(\sigma'\sqrt{2t}-u)^2}{4u\sigma'\sqrt{2t}}\right) \geqslant 0.5, \tag{B.24}$$

for any $\boldsymbol{\delta} = (u, 0, \ldots, 0)^\intercal$, $\mathsf{C}_{\boldsymbol{\delta}}(P_A, Q_A) > 0.5$.

By definition (Equation (2.9)),

$$\begin{aligned}
&\mathsf{C}_{\boldsymbol{\delta}}(P_A, Q_A)\\
=&\min_f \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon}+\boldsymbol{\delta})] \quad \text{s.t.}\ \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon})] = P_A,\ \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[f(\boldsymbol{\epsilon})] = Q_A, 0\leqslant f(\boldsymbol{\epsilon})\leqslant 1 \quad \forall \epsilon\in\mathbb{R}^d\\
\geqslant&\min_f \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon}+\boldsymbol{\delta})] \quad \text{s.t.}\ \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[f(\boldsymbol{\epsilon})] = 1, 0\leqslant f(\boldsymbol{\epsilon})\leqslant 1 \quad \forall \epsilon\in\mathbb{R}^d\\
=&\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon}+\boldsymbol{\delta})] \quad \text{where} \quad f(\boldsymbol{\epsilon}) = \begin{cases} 1, & \|\boldsymbol{\epsilon}\|_2\leqslant T\\ 0. & \|\boldsymbol{\epsilon}\|_2 > T\end{cases}\\
=:&V.
\end{aligned} \tag{B.25}$$

We now compute V:

$$\begin{aligned}
V =& \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[\|\boldsymbol{\epsilon}+\boldsymbol{\delta}\|_2\leqslant T] && \text{(B.26)}\\
=&\int_{\mathbb{R}^d} p(\boldsymbol{x})\mathbb{I}[\|\boldsymbol{x}+\boldsymbol{\delta}\|_2\leqslant T]\,\mathrm{d}\boldsymbol{x}\\
\overset{(1)}{=}&\int_0^\infty y\,\mathrm{d}y\int_{\substack{p(\boldsymbol{x})=y\\ \mathbb{I}[\|\boldsymbol{x}+\boldsymbol{\delta}\|_2\leqslant T}} \frac{\mathrm{d}\boldsymbol{x}}{\|\nabla p(\boldsymbol{x})\|_2}\\
\overset{(2)}{=}&\int_0^\infty y\,\mathrm{d}y\int_{\substack{p(\boldsymbol{x})=y\\ \mathbb{I}[\|\boldsymbol{x}+\boldsymbol{\delta}\|_2\leqslant T}} -\frac{\mathrm{d}\boldsymbol{x}}{r_p'(r_p^{-1}(y))}\\
\overset{(3)}{=}&\int_0^\infty y\,\mathrm{d}y\frac{2\pi^{d/2}}{\Gamma(\frac{d}{2})}r_p^{-1}(y)^{d-1}\cdot\left(-\frac{1}{r_p'(r_p^{-1}(y))}\right)\cdot\Pr[\|\boldsymbol{x}+\boldsymbol{\delta}\|_2\leqslant T\,|\,p(\boldsymbol{x})=y]\\
\overset{(4)}{=}&\int_0^\infty r_p(t)\,\mathrm{d}t\frac{2\pi^{d/2}}{\Gamma(\frac{d}{2})}t^{d-1}\Pr[\|\boldsymbol{x}+\boldsymbol{\delta}\|_2\leqslant T\,|\,\|\boldsymbol{x}\|_2=t]\\
\overset{(5)}{=}&\int_0^\infty \frac{1}{(2\sigma'^2)^{\frac{d}{2}-k}\pi^{\frac{d}{2}}}\cdot\frac{\Gamma(\frac{d}{2})}{\Gamma(\frac{d}{2}-k)}t^{-2k}\exp\left(-\frac{t^2}{2\sigma'^2}\right)\frac{2\pi^{d/2}}{\Gamma(\frac{d}{2})}t^{d-1}\Pr[\|\boldsymbol{x}+\boldsymbol{\delta}\|_2\leqslant T\,|\,\|\boldsymbol{x}\|_2=t]\,\mathrm{d}t\\
=&\frac{1}{\Gamma(\frac{d}{2}-k)}\int_0^\infty t^{d/2-k-1}\exp(-t)\Pr[\|\boldsymbol{x}+\boldsymbol{\delta}\|_2\leqslant T\,|\,\|\boldsymbol{x}\|_2=\sigma'\sqrt{2t}]\,\mathrm{d}t && \text{(B.27)}\\
=&\mathop{\mathbb{E}}_{t\sim\Gamma(\frac{d}{2}-k)}\Pr[\|\boldsymbol{x}+\boldsymbol{\delta}\|_2\leqslant T\,|\,\|\boldsymbol{x}\|_2=\sigma'\sqrt{2t}]. && \text{(B.28)}
\end{aligned}$$

In above equations, (1) follows from level-set sliced integration extended from [427] and $p(\boldsymbol{x})$

is the density of distribution $\mathcal{P} = \mathcal{N}^{\mathrm{g}}(k, \sigma)$ at point $\boldsymbol{x}$; in (2) we define $r_p(\|\boldsymbol{x}\|_2) := p(\boldsymbol{x})$ noting that $\mathcal{P}$ is ℓ_2 symmetric and all $\boldsymbol{x}$ with same ℓ_2 length having the same $p(\boldsymbol{x})$, and we have $\|\nabla p(\boldsymbol{x})\|_2 = -r_p'(r_p^{-1}(y))$ since $y = r_p(\|\boldsymbol{x}\|_2)$ and r_p is monotonically decreasing; (3) uses

$$\mathrm{Vol}(\{\boldsymbol{x} : p(\boldsymbol{x}) = y\}) = \mathrm{Vol}(\{\boldsymbol{x} : \|\boldsymbol{x}\|_2 = r_p^{-1}(y)\}) = \frac{2\pi^{d/2}}{\Gamma(\frac{d}{2})} r_p^{-1}(y)^{d-1}; \tag{B.29}$$

(4) changes the integration variable from y to $t = r_p^{-1}(y)$; and (5) injects the concrete expression of r_p.

Now, we inspect $\Pr[\|\boldsymbol{x} + \boldsymbol{\delta}\|_2 \leqslant T \,|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}]$: When $\|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}$, $\sum_{i=1}^d x_i^2 = 2t\sigma'^2$. Meanwhile, $\|\boldsymbol{x} + \boldsymbol{\delta}\|_2 \leqslant T$ means $(x_1 + u)^2 + \sum_{i=2}^d x_i^2 \leqslant T^2$. Thus, when $\|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}$,

$$\|\boldsymbol{x} + \boldsymbol{\delta}\|_2 \leqslant T \iff \frac{x_1}{\sigma'\sqrt{2t}} \leqslant \frac{T^2 - u^2 - 2t\sigma'^2}{2u\sigma'\sqrt{2t}}. \tag{B.30}$$

According to [427, Lemma I.23], for $\boldsymbol{x}$ uniformly sampled from sphere with radius $\sigma'\sqrt{2t}$, the component coordinate $\dfrac{1 + \frac{x_1}{\sigma'\sqrt{2t}}}{2} \sim \mathrm{Beta}(\frac{d-1}{2}, \frac{d-1}{2})$. Thus,

$$\Pr[\|\boldsymbol{x} + \boldsymbol{\delta}\|_2 \leqslant T \,|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}] = \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{1 + \frac{T^2 - u^2 - 2t\sigma'^2}{2u\sigma'\sqrt{2t}}}{2}\right) \tag{B.31}$$

$$= \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2 - (\sigma'\sqrt{2t} - u)^2}{4u\sigma'\sqrt{2t}}\right). \tag{B.32}$$

Finally, we get

$$V = \mathop{\mathbb{E}}_{t \sim \Gamma(\frac{d}{2} - k, 1)} \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2 - (\sigma'\sqrt{2t} - u)^2}{4u\sigma'\sqrt{2t}}\right). \tag{B.33}$$

In other words, when

$$\mathop{\mathbb{E}}_{t \sim \Gamma(\frac{d}{2} - k, 1)} \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2 - (\sigma'\sqrt{2t} - u)^2}{4u\sigma'\sqrt{2t}}\right) \geqslant 0.5, \tag{B.34}$$

we have $V \geqslant 0.5$, and thus $\mathbf{C}_\delta(P_A, Q_A) \geqslant V \geqslant 0.5$, $r_{\mathsf{DSRS}} \geqslant u$, which concludes the proof. QED.

We require the following property of BetaCDF.

Lemma B.3. There exists $d_0 \in \mathbb{N}_+$, for any $d \geqslant d_0$,

$$\mathrm{BetaCDF}_{\frac{d-1}{2}}(0.6) \geqslant 0.999. \tag{B.35}$$

Proof of Lemma B.3. We let $\mathrm{v} \sim \mathrm{Beta}(\frac{d-1}{2}, \frac{d-2}{2})$, then

$$\mathbb{E}[\mathrm{v}] = 1/2, \tag{B.36}$$

$$\mathrm{Var}[\mathrm{v}] = \frac{(\frac{d-1}{2})^2}{(\frac{d-1}{2}+\frac{d-1}{2})^2(\frac{d-1}{2}+\frac{d-1}{2}+1)} = \frac{1}{4d}. \tag{B.37}$$

Now, applying Chebyshev's inequality, we have

$$\Pr[|\mathrm{v} - 0.5| \geqslant 0.1] \leqslant \frac{1}{0.04d}. \tag{B.38}$$

Therefore,

$$\mathrm{BetaCDF}_{\frac{d-1}{2}}(0.6) = \Pr[\mathrm{v} < 0.6] = 1 - \Pr[\mathrm{v} \geqslant 0.6] \tag{B.39}$$

$$\geqslant 1 - \Pr[|\mathrm{v} - 0.5| \geqslant 0.1] \geqslant 1 - \frac{1}{0.04d}. \tag{B.40}$$

Thus, when $d \geqslant 25000$, $\mathrm{BetaCDF}_{\frac{d-1}{2}}(0.6) \geqslant 0.999$. QED.

We also require the following properties of the gamma distribution.

Proposition B.2. For any $P_{\mathsf{con}} \in (0,1)$, there exists $d_0 \in \mathbb{N}_+$, for any $d \geqslant d_0$,

$$\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}}) \geqslant 0.99 \cdot \frac{d}{2}. \tag{B.41}$$

Proof of Proposition B.2. We let $\mathrm{v} \sim \Gamma(d/2)$, then

$$\mathbb{E}[\mathrm{v}] = d/2, \mathrm{Var}[\mathrm{v}] = d/2. \tag{B.42}$$

We now apply Chebyshev's inequality and get

$$\Pr[\mathrm{v} < 0.99 \cdot d/2] \leqslant \Pr[|\mathrm{v} - d/2| > 0.01 \cdot d/2] \leqslant \frac{20000}{d}. \tag{B.43}$$

Thus, for any $P_{\mathsf{con}} \in (0,1)$, when $d \geqslant \dfrac{20000}{P_{\mathsf{con}}}$,

$$\Gamma\mathrm{CDF}_{d/2}\left(0.99 \cdot \frac{d}{2}\right) \leqslant \frac{20000}{d} \leqslant P_{\mathsf{con}}, \tag{B.44}$$

i.e., $\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}}) \geqslant 0.99 \cdot \frac{d}{2}$. QED.

Table B.1: Table for proof of Proposition B.3.

$\frac{d}{2}-k$	$\Gamma\mathrm{CDF}_{d/2-k}(0.98(d/2-k))$	$\frac{d}{2}-k$	$\Gamma\mathrm{CDF}_{d/2-k}(0.98(d/2-k))$
0.5	0.6778	8.0	0.5245
1.0	0.6247	8.5	0.5224
1.5	0.5990	9.0	0.5204
2.0	0.5831	9.5	0.5186
2.5	0.5718	10.0	0.5168
3.0	0.5632	10.5	0.5152
3.5	0.5564	11.0	0.5136
4.0	0.5507	11.5	0.5121
4.5	0.5459	12.0	0.5107
5.0	0.5418	12.5	0.5093
5.5	0.5381	13.0	0.5080
6.0	0.5349	13.5	0.5068
6.5	0.5319	14.0	0.5056
7.0	0.5292	14.5	0.5044
7.5	0.5268	15.0	0.5033

Proposition B.3. When $d/2-15 \leqslant k < d/2$, $k, d \in \mathbb{N}_+$,

$$\Pr_{t\sim\Gamma(d/2-k)}\left[t \leqslant 0.98\left(\frac{d}{2}-k\right)\right] \geqslant \frac{0.5}{0.999}. \tag{B.45}$$

Proof of Proposition B.3. We prove the proposition by enumeration. Notice that $d/2-k \in \{0.5, 1.0, \cdots, 14.5, 15.0\}$, we enumerate $\Gamma\mathrm{CDF}_{d/2-k}(0.98(d/2-k))$ for each $(d/2-k)$ and get Table B.1.

On the other hand, $\frac{0.5}{0.999} \leqslant 0.5001$, which concludes the proof. QED.

Now we are ready to prove the main theorem.

Proof of Theorem 2.2. According to Lemma B.2, we only need to show that for $u = 0.02\sigma\sqrt{d}$, Equation (B.21) holds. For sufficiently large d, indeed,

$$\begin{aligned}
&\mathop{\mathbb{E}}_{t\sim\Gamma(d/2-k)} \mathrm{BetaCDF}_{(d-1)/2}\left(\frac{T^2-(\sigma'\sqrt{2t}-u)^2}{4u\sigma'\sqrt{2t}}\right)\\
\overset{\text{Lemma B.3}}{\geqslant}\ & 0.999 \mathop{\mathbb{E}}_{t\sim\Gamma(d/2-k)} \mathbb{I}\left[\frac{T^2-(\sigma'\sqrt{2t}-u)^2}{4u\sigma'\sqrt{2t}} \geqslant 0.6\right] &\text{(B.46)}\\
\overset{(*)}{\geqslant}\ & 0.999 \mathop{\mathbb{E}}_{t\sim\Gamma(d/2-k)} \mathbb{I}\left[t \leqslant 0.98\left(\frac{d}{2}-k\right)\right] &\text{(B.47)}
\end{aligned}$$

$$\overset{\text{Proposition B.3}}{\geqslant} 0.999 \cdot \frac{0.5}{0.999} = 0.5. \tag{B.48}$$

Thus, from Lemma B.2 we have $r_{\mathsf{DSRS}} \geqslant u = 0.02\sqrt{d}$.

The inequality $(*)$ follows from Fact B.1. QED.

Fact B.1. Under the condition of Theorem 2.2, for sufficiently large d,

$$t \leqslant 0.98\left(\frac{d}{2} - k\right) \Rightarrow \frac{T^2 - (\sigma'\sqrt{2t} - u)^2}{4u\sigma'\sqrt{2t}} \geqslant 0.6. \tag{B.49}$$

Proof of Fact B.1.

$$\begin{aligned} &\frac{T^2 - (\sigma'\sqrt{2t} - u)^2}{4u\sigma'\sqrt{2t}} \geqslant 0.6 \\ \iff & T^2 - (\sigma'\sqrt{2t} - u)^2 \geqslant 2.4u\sigma'\sqrt{2t} \\ \overset{x := \sigma'\sqrt{2t}}{\iff} & T^2 - (x - u)^2 \geqslant 2.4ux \\ \iff & x^2 + 0.4ux + u^2 - T^2 \leqslant 0. \end{aligned} \tag{B.50}$$

From Proposition B.2, we have

$$u^2 - T^2 = 0.0004d\sigma^2 - 2\sigma^2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}}) \leqslant 0.0004d\sigma^2 - 0.99d\sigma^2 < 0. \tag{B.51}$$

Thus,

$$\begin{aligned} & x^2 + 0.4ux + u^2 - T^2 \leqslant 0 \\ \iff & x \leqslant \frac{-0.4u + \sqrt{0.16u^2 - 4(u^2 - T^2)}}{2} = -0.2u + \sqrt{T^2 - 0.96u^2} \\ \iff & t \leqslant \frac{(-0.2u + \sqrt{T^2 - 0.96u^2})^2}{2\sigma'^2}. \end{aligned} \tag{B.52}$$

Again, from Proposition B.2,

$$\begin{aligned} & T^2 - 0.96u^2 \\ =& 2\sigma^2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}}) - 0.96 \times 0.0004d\sigma^2 \geqslant (0.99 - 0.96 \times 0.0004)d\sigma^2, \end{aligned} \tag{B.53}$$

and therefore

$$\begin{aligned} & (-0.2u + \sqrt{T^2 - 0.96u^2})^2 \\ \geqslant & d\sigma^2\left(-0.004 + \sqrt{0.99 - 0.96 \times 0.0004}\right)^2 \approx 0.9816d\sigma^2 \geqslant 0.98d\sigma^2. \end{aligned} \tag{B.54}$$

Then,

$$t \leqslant \frac{(-0.2u + \sqrt{T^2 - 0.96u^2})^2}{2\sigma'^2} \Longleftarrow t \leqslant \frac{0.98d\sigma^2}{2\sigma^2} \cdot \frac{d-2k}{d} \iff t \leqslant 0.98\left(\frac{d}{2} - k\right). \quad \text{(B.55)}$$

QED.

B.1.4 Theorem B.1

Theorem B.1. Let d be the input dimension and F_0 be the base classifier. For an input point $\boldsymbol{x}_0 \in \mathbb{R}^d$ with true class y_0, suppose F_0 satisfies $(\sigma, P_{\mathsf{con}})$-Concentration property and $\Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}(\sigma)}[F_0(\boldsymbol{x}_0+\boldsymbol{\epsilon}) = y_0] = P_{\mathsf{con}}$ where $P_{\mathsf{con}} < 1$. The smoothed classifier $\widetilde{F}_0^{\mathcal{P}'}$ is constructed from F_0 and smoothed by generalized Gaussian $\mathcal{P}' = \mathcal{N}^{\mathtt{g}}(k_0, \sigma)$ where k_0 is a constant independent of input dimension d. Then, for any constant $c > 0$, there exists d_0, such that when input dimension $d \geqslant d_0$, **any method cannot certify** ℓ_2 **radius** $c\sqrt{d}$, where $T = \sigma\sqrt{2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}})}$ and $\Gamma\mathrm{CDF}_{d/2}$ is the CDF of gamma distribution $\Gamma(d/2, 1)$.

This theorem suggests that, if we use generalized Gaussian whose k is a constant with respect to input dimension d or use standard Gaussian (whose $k = 0$ is a constant) for smoothing, we cannot achieve $\Omega(\sqrt{d})$ certified radius rate from DSRS and any other certification method.

The proof of Theorem B.1 is based on three lemmas listed below.

Lemma B.4. Given $k_0 \in \mathbb{N}$, for any $\epsilon > 0$, there exists d_0, such that when $d > d_0$,

$$\Pr_{\mathrm{t}\sim\Gamma\left(\frac{d}{2}-k_0,1\right)}\left[\mathrm{t} \leqslant (1-\epsilon)\left(\frac{d}{2} - k_0\right)\right] \leqslant \frac{0.48}{0.99}. \quad \text{(B.56)}$$

Lemma B.5. Given $P_{\mathsf{con}} \geqslant 0$, for any $\epsilon > 0$, there exists d_0, such that when $d > d_0$,

$$T := \sigma\sqrt{2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}})} \leqslant \sigma\sqrt{(1+\epsilon)d}. \quad \text{(B.57)}$$

Lemma B.6. For any $\epsilon > 0$, there exists d_0, such that when $d > d_0$,

$$\mathrm{BetaCDF}_{\frac{d-1}{2}}(0.5 - \epsilon) \leqslant 0.01. \quad \text{(B.58)}$$

Proofs of these lemmas are based on Chebyshev's inequality.

Proof of Lemma B.4. For $\mathrm{t} \sim \Gamma(d/2 - k_0, 1)$, we have

$$\mathbb{E}[\mathrm{t}] = d/2 - k_0,\ \mathrm{Var}[\mathrm{t}] = d/2 - k_0. \tag{B.59}$$

By Chebyshev's inequality,

$$\Pr\left[\mathrm{t} \leqslant (1-\epsilon)\left(\frac{d}{2} - k_0\right)\right] \leqslant \Pr\left[|\mathrm{t} - \mathbb{E}[\mathrm{t}]| \geqslant \epsilon\left(\frac{d}{2} - k_0\right)\right] \leqslant \frac{1}{\epsilon^2(\frac{d}{2} - k_0)}. \tag{B.60}$$

Picking $d_0 = 2\left(\frac{0.99}{0.48\epsilon^2} + k_0\right)$ concludes the proof. QED.

Proof of Lemma B.5. We define random variable $\mathrm{v} \sim \Gamma(d/2, 1)$, so $\mathbb{E}[\mathrm{v}] = d/2, \mathrm{Var}[\mathrm{v}] = d/2$. By Chebyshev's inequality,

$$\begin{aligned}
&\Pr[\mathrm{v} \leqslant (1+\epsilon)d/2] \\
\geqslant &1 - \Pr[\mathrm{v} \geqslant (1+\epsilon)d/2] \\
\geqslant &1 - \Pr[|\mathrm{v} - \mathbb{E}[\mathrm{v}]| \geqslant \epsilon d/2] \\
\geqslant &1 - \frac{2}{d\epsilon^2}.
\end{aligned} \tag{B.61}$$

Let $d_0 = \frac{2}{\epsilon^2(1-P_{\mathsf{con}})}$. Thus, when $d > d_0$, $\Pr[\mathrm{v} \leqslant (1+\epsilon)d/2] \geqslant 1 - \frac{2}{d_0\epsilon^2} = P_{\mathsf{con}}$, which implies that $\Gamma\mathrm{CDF}_{d/2}((1+\epsilon)d/2) \geqslant P_{\mathsf{con}}$ and $\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}}) \leqslant (1+\epsilon)d/2$ and concludes the proof. QED.

Proof of Lemma B.6. We define random variable $\mathrm{v} \sim \mathrm{Beta}(\frac{d-1}{2}, \frac{d-1}{2})$, and we have $\mathbb{E}[\mathrm{v}] = 1/2, \mathrm{Var}[\mathrm{v}] = \frac{1}{4d}$. By Chebyshev's inequality,

$$\Pr[\mathrm{v} \leqslant 0.5 - \epsilon] \leqslant \Pr[|\mathrm{v} - \mathbb{E}[\mathrm{v}]| \geqslant \epsilon] \leqslant \frac{1}{4d\epsilon^2}. \tag{B.62}$$

Let $d_0 = \frac{25}{d\epsilon^2}$, when $d > d_0$, $\Pr[\mathrm{v} \leqslant 0.5 - \epsilon] \leqslant 0.01$ and hence $\mathrm{BetaCDF}_{\frac{d-1}{2}}(0.5 - \epsilon) \leqslant 0.01$. QED.

Now we are ready to prove the main theorem.

Proof of Theorem B.1. According to the above three lemmas, we pick d_0, such that when $d > d_0$, the followings hold simultaneously.

$$\Pr_{\mathrm{t} \sim \Gamma\left(\frac{d}{2} - k_0, 1\right)}\left[\mathrm{t} \leqslant \left(1 - \frac{c^2}{8\sigma^2}\right)\left(\frac{d}{2} - k_0\right)\right] \leqslant \frac{0.48}{0.99}, \tag{B.63}$$

$$T = \sigma\sqrt{2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}})} \leqslant \sigma\sqrt{\left(1+\frac{c^2}{8\sigma^2}\right)d}, \tag{B.64}$$

$$\mathrm{BetaCDF}_{\frac{d-1}{2}}\left(0.5-\frac{c}{8\sigma}\right) \leqslant 0.01. \tag{B.65}$$

We define vector $\boldsymbol{\delta} = (c\sqrt{d}, 0, 0, \ldots, 0)^\intercal$. Since F_0 satisfies $(\sigma, P_{\mathsf{con}})$-concentration property and $\Pr_{\boldsymbol{\epsilon}\sim\mathcal{N}(\sigma)}[F_0(\boldsymbol{x}_0+\boldsymbol{\epsilon}) = y_0] = P_{\mathsf{con}}$, up to a set of zero measure, the region $\{\boldsymbol{\epsilon} : F_0(\boldsymbol{x}_0+\boldsymbol{\epsilon}) = y_0\}$ and region $\{\boldsymbol{\epsilon} : \|\boldsymbol{\epsilon}\|_2 \leqslant T\}$ coincide.

We now show that $\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{P}'=\mathcal{N}^{\mathrm{g}}(k_0,\sigma)}[F_0(\boldsymbol{x}_0+\boldsymbol{\delta}+\boldsymbol{\epsilon}) = y_0] < 0.5$ when $c \leqslant \sigma\sqrt{8/7}$.

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}'}[F_0(\boldsymbol{x}_0+\boldsymbol{\delta}+\boldsymbol{\epsilon}) = y_0] \\
=&\Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}'}[\|\boldsymbol{\delta}+\boldsymbol{\epsilon}\|_2 \leqslant T] \\
=&\mathop{\mathbb{E}}_{\mathrm{t}\sim\Gamma(\frac{d}{2}-k_0,1)} \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2-(\sigma'\sqrt{2\mathrm{t}}-c\sqrt{d})^2}{4\sigma'\sqrt{2\mathrm{t}}\cdot c\sqrt{d}}\right) && \text{(from Equation (B.33))} \\
\leqslant&0.99\mathop{\mathbb{E}}_{\mathrm{t}\sim\Gamma(\frac{d}{2}-k_0,1)} \mathbb{I}\left[\frac{T^2-(\sigma'\sqrt{2\mathrm{t}}-c\sqrt{d})^2}{4\sigma'\sqrt{2\mathrm{t}}\cdot c\sqrt{d}} \geqslant 0.5-\frac{c}{8\sigma}\right]+0.01 && \text{(by Equation (B.65) and} \\
& && \mathrm{BetaCDF}_{\cdot}(\cdot) \leqslant 1) \\
=&0.99\Pr_{\mathrm{t}\sim\Gamma(\frac{d}{2}-k_0,1)}\left[\frac{T^2-(\sigma'\sqrt{2\mathrm{t}}-c\sqrt{d})^2}{4\sigma'\sqrt{2\mathrm{t}}\cdot c\sqrt{d}} \geqslant 0.5-\frac{c}{8\sigma}\right]+0.01. && \text{(B.66)}
\end{aligned}$$

Since

$$\begin{aligned}
&\frac{T^2-(\sigma'\sqrt{2\mathrm{t}}-c\sqrt{d})^2}{4\sigma'\sqrt{2\mathrm{t}}\cdot c\sqrt{d}} \geqslant 0.5-\frac{c}{8\sigma} \\
\iff& T^2-2\mathrm{t}\sigma^2\frac{d}{d-2k_0}-dc^2+\frac{c^2d}{2}\sqrt{\frac{2\mathrm{t}}{d-2k_0}} \geqslant 0 && \text{(B.67)} \\
\overset{\text{Equation (B.64)}}{\Longrightarrow}& \left(1+\frac{c^2}{8\sigma^2}\right)d\sigma^2-2\mathrm{t}\sigma^2\frac{d}{d-2k_0}-dc^2+\frac{c^2d}{2}\sqrt{\frac{2\mathrm{t}}{d-2k_0}} \geqslant 0. && \text{(B.68)}
\end{aligned}$$

We now inject $\mathrm{t} = 0$ and $\mathrm{t} = \left(1-\frac{c^2}{8\sigma^2}\right)\left(\frac{d}{2}-k_0\right)$ to the LHS of Equation (B.68).

- When $\mathrm{t} = 0$,

$$\text{LHS of Equation (B.68)} = \left(1+\frac{c^2}{8\sigma^2}\right)d\sigma^2-dc^2 = d\left(\sigma^2-\frac{7}{8}c^2\right) \geqslant 0. \tag{B.69}$$

- When $\mathrm{t} = \left(1 - \frac{c^2}{8\sigma^2}\right)\left(\frac{d}{2} - k_0\right)$,

$$\begin{aligned}
&\text{LHS of Equation (B.68)} \\
=&\left(1 + \frac{c^2}{8\sigma^2}\right) d\sigma^2 - \frac{2d\sigma^2}{d - 2k_0}\left(1 - \frac{c^2}{8\sigma^2}\right)\left(\frac{d}{2} - k_0\right) - dc^2 + \frac{dc^2}{2}\sqrt{1 - \frac{c^2}{8\sigma^2}} \\
=&d\sigma^2 + \frac{dc^2}{8} - d\sigma^2 + \frac{dc^2}{8} - dc^2 + \frac{dc^2}{2}\sqrt{1 - \frac{c^2}{8\sigma^2}} \\
\leqslant&\frac{dc^2}{4} - dc^2 + \frac{dc^2}{2} < 0.
\end{aligned} \tag{B.70}$$

Notice that the LHS of Equation (B.68) is a parabola with negative second-order coefficient. Thus,

$$\text{Equation (B.68)} \Longrightarrow \mathrm{t} \in \left[0, \left(1 - \frac{c^2}{8\sigma^2}\right)\left(\frac{d}{2} - k_0\right)\right] \tag{B.71}$$

and hence

$$\begin{aligned}
&\Pr_{\mathrm{t}\sim\Gamma(\frac{d}{2}-k_0,1)}\left[\frac{T^2 - (\sigma'\sqrt{2\mathrm{t}} - c\sqrt{d})^2}{4\sigma'\sqrt{2\mathrm{t}}\cdot c\sqrt{d}} \geqslant 0.5 - \frac{c}{8\sigma}\right] \\
\leqslant&\Pr_{\mathrm{t}\sim\Gamma(\frac{d}{2}-k_0,1)}\left[\mathrm{t} \leqslant \left(1 - \frac{c^2}{8\sigma^2}\right)\left(\frac{d}{2} - k_0\right)\right] \leqslant \frac{0.48}{0.99}. \qquad \text{(by Equation (B.63))}
\end{aligned} \tag{B.72}$$

Plugging this inequality to Equation (B.66), we get

$$\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}'}[F_0(\boldsymbol{x}_0 + \boldsymbol{\delta} + \boldsymbol{\epsilon}) = y_0] \leqslant 0.99 \cdot \frac{0.48}{0.99} + 0.01 = 0.49. \tag{B.73}$$

As a result, when $c \leqslant \sigma\sqrt{8/7}$, the smoothed classifier $\widetilde{F}_0^{\mathcal{P}'}$ is not robust given the perturbation $\boldsymbol{\delta} = (c\sqrt{d}, 0, 0, \cdots, 0)^\intercal$, since there may exist another $y' \neq y_0$ with $\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{P}'}[F_0(\boldsymbol{x}_0 + \boldsymbol{\delta} + \boldsymbol{\epsilon}) = y'] \geqslant 0.51$ so $\widetilde{F}_0^{\mathcal{P}'}(\boldsymbol{x}_0 + \boldsymbol{\delta}) = y' \neq y_0$.

When $c > \sigma\sqrt{8/7}$, within the ℓ_2 radius ball $c\sqrt{d}$, there exists perturbation vector $\boldsymbol{\delta} = (c'\sqrt{d}, 0, 0, \cdots, 0)^\intercal$ fooling smoothed classifier $\widetilde{F}_0^{\mathcal{P}'}$ where $c' = \sigma\sqrt{8/7}$. Hence, for any $c > 0$, there exists a perturbation within ℓ_2 ball with radius $c\sqrt{d}$, such that smoothed classifier $\widetilde{F}_0^{\mathcal{P}'}$ can be fooled, and then any robustness certification method cannot certify ℓ_2 radius $c\sqrt{d}$ since the smoothed classifier itself is not robust. QED.

B.2 PROOFS OF DSRS COMPUTATIONAL METHOD

B.2.1 Proof of Strong Duality (Theorem 2.3)

Proof of Theorem 2.3. We write down the Lagrangian dual function of Equation (2.9a):

$$\Lambda(f,\,\lambda_1,\,\lambda_2) := \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\delta}+\boldsymbol{\epsilon})] - \lambda_1\left(\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon})] - P_A\right) - \lambda_2\left(\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[f(\boldsymbol{\epsilon})] - Q_A\right). \tag{B.74}$$

Then, from $\mathbf{C}$'s expression (Equation (2.9)), we have

$$\begin{aligned}
&\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A)\\
=&\min_{f}\ \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon}+\boldsymbol{\delta})] \quad \text{s.t.} \quad 0\leqslant f(\boldsymbol{\epsilon})\leqslant 1\ \forall \boldsymbol{\epsilon}\in\mathbb{R}^d,\ \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon})] = P_A,\ \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[f(\boldsymbol{\epsilon})] = Q_A\\
=&\min_{f:\mathbb{R}^d\to[0,1]}\ \max_{\lambda_1,\lambda_2\in\mathbb{R}} \Lambda(f,\,\lambda_1,\,\lambda_2)\\
\overset{(i)}{\geqslant}&\max_{\lambda_1,\lambda_2\in\mathbb{R}}\ \min_{f:\mathbb{R}^d\to[0,1]} \Lambda(f,\,\lambda_1,\,\lambda_2)\\
\overset{(ii)}{=}&\max_{\lambda_1,\lambda_2\in\mathbb{R}}\ \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon}) < \lambda_1 p(\boldsymbol{\epsilon}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{\epsilon}+\boldsymbol{\delta})]\\
&\qquad - \lambda_1 \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] - \lambda_2 \Pr_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})]\\
&\qquad + \lambda_1 P_A + \lambda_2 Q_A.
\end{aligned} \tag{B.75}$$

In the above equation, (i) is from the min-max inequality. For completeness, we provide the proof as such: Define $g:\mathbb{R}^2\to\mathbb{R}$ such that $g(\lambda_1,\lambda_2) := \min_{f:\mathbb{R}^d\to[0,1]}\Lambda(f,\,\lambda_1,\lambda_2)$. As a result, for any $\lambda_1,\lambda_2\in\mathbb{R}$ and any $f:\mathbb{R}^d\to[0,1]$, $g(\lambda_1,\lambda_2)\leqslant\Lambda(f,\lambda_1,\lambda_2)$. So for any $f:\mathbb{R}^d\to[0,1]$, $\max_{\lambda_1,\lambda_2\in\mathbb{R}} g(\lambda_1,\lambda_2)\leqslant\max_{\lambda_1,\lambda_2\in\mathbb{R}}\Lambda(f,\lambda_1,\lambda_2)$, which implies

$$\max_{\lambda_1,\lambda_2\in\mathbb{R}} g(\lambda_1,\lambda_2) \leqslant \min_{f:\mathbb{R}\to[0,1]}\ \max_{\lambda_1,\lambda_2\in\mathbb{R}} \Lambda(f,\lambda_1,\lambda_2), \tag{B.76}$$

where LHS is the RHS of (i) and RHS is the LHS of (i).

In above equation, (ii) comes from a closed-form solution of f for $\Lambda(f,\lambda_1,\lambda_2)$ given $(\lambda_1,\lambda_2)\in\mathbb{R}^2$. Notice that we can rewrite $\Lambda(f,\lambda_1,\lambda_2)$ as an integral over $\mathbb{R}^d$:

$$\begin{aligned}
&\Lambda(f,\,\lambda_1,\,\lambda_2)\\
=&\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\delta}+\boldsymbol{\epsilon})] - \lambda_1\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon})] - \lambda_2\mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[f(\boldsymbol{\epsilon})] + \lambda_1 P_A + \lambda_2 Q_A\\
=&\int_{\mathbb{R}^d} f(\boldsymbol{x})\cdot\big(p(\boldsymbol{x}-\boldsymbol{\delta}) - \lambda_1 p(\boldsymbol{x}) - \lambda_2 q(\boldsymbol{x})\big)\,\mathrm{d}\boldsymbol{x} + \lambda_1 P_A + \lambda_2 Q_A.
\end{aligned} \tag{B.77}$$

We would like to minimize over $f : \mathbb{R}^d \to [0, 1]$ in Equation (B.77) and simple greedy solution reveals that we should choose

$$f(\boldsymbol{x}) = \begin{cases} 1, & p(\boldsymbol{x} - \boldsymbol{\delta}) - \lambda_1 p(\boldsymbol{x}) - \lambda_2 q(\boldsymbol{x}) < 0 \\ 0. & p(\boldsymbol{x} - \boldsymbol{\delta}) - \lambda_1 p(\boldsymbol{x}) - \lambda_2 q(\boldsymbol{x}) \geqslant 0 \end{cases} \tag{B.78}$$

We inject this f into Equation (B.77) and get

$$\begin{aligned} &\min_{f:\mathbb{R}^d \to [0,1]} \Lambda(f, \lambda_1, \lambda_2) \\ &= \Pr_{\boldsymbol{\epsilon} \sim \mathcal{P}}[p(\boldsymbol{\epsilon}) < \lambda_1 p(\boldsymbol{\epsilon} + \boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{\epsilon} + \boldsymbol{\delta})] \quad + \lambda_1 \left(P_A - \Pr_{\boldsymbol{\epsilon} \sim \mathcal{P}}[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] \right) \\ &\qquad\qquad\qquad\qquad\qquad\qquad + \lambda_2 \left(Q_A - \Pr_{\boldsymbol{\epsilon} \sim \mathcal{Q}}[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] \right) \end{aligned} \tag{B.79}$$

so (ii) holds.

On the other hand, we know that $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$ (defined by Equation (2.11)) is feasible by theorem statement. Denote $(\lambda_1^*, \lambda_2^*) \in \mathbb{R}^2$ to a feasible solution to $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$ and d^* to the objective value, then from the constraints of $(\mathbf{D})$ we know

$$\begin{aligned} \Pr_{\boldsymbol{\epsilon} \sim \mathcal{P}}[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1^* p(\boldsymbol{\epsilon}) + \lambda_2^* q(\boldsymbol{\epsilon})] &= P_A, \\ \Pr_{\boldsymbol{\epsilon} \sim \mathcal{Q}}[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1^* p(\boldsymbol{\epsilon}) + \lambda_2^* q(\boldsymbol{\epsilon})] &= Q_A, \\ \Pr_{\boldsymbol{\epsilon} \sim \mathcal{P}}[p(\boldsymbol{\epsilon}) < \lambda_1^* p(\boldsymbol{\epsilon} + \boldsymbol{\delta}) + \lambda_2^* q(\boldsymbol{\epsilon} + \boldsymbol{\delta})] &= d^*. \end{aligned} \tag{B.80}$$

Plugging in these equalities into Equation (B.75), we have

$$\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) \geqslant d^* - \lambda_1 P_A - \lambda_2 Q_A + \lambda_1 P_A + \lambda_2 Q_A = d^*. \tag{B.81}$$

At the same time, we define function $f^* : \mathbb{R}^d \to [0, 1]$ such that

$$f^*(\boldsymbol{x}) = \mathbb{I}[p(\boldsymbol{x} - \boldsymbol{\delta}) - \lambda_1^* p(\boldsymbol{x}) - \lambda_2^* q(\boldsymbol{x}) < 0]. \tag{B.82}$$

From Equation (B.80), f^* satisfies the constraints of $(\mathbf{C})$ (Equations (2.9b) and (2.9c)). Since $(\mathbf{C})$ minimizes over all possible functions $f : \mathbb{R}^d \to [0, 1]$, we have

$$\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) \leqslant \mathop{\mathbb{E}}_{\boldsymbol{\epsilon} \sim \mathcal{P}}[f^*(\boldsymbol{\epsilon} + \boldsymbol{\delta})] = d^*. \tag{B.83}$$

Combining Equations (B.81) and (B.83), $\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A) = d^*$ and hence strong duality holds.

QED.

B.2.2 Proofs of Propositions 2.2 and 2.3 and theorem 2.4

Proof of Proposition 2.2. Suppose f_1 is the optimal solution to $\mathbf{C}_{\boldsymbol{\delta}}(P_A^1, Q_A^1)$ and f_2 is the optimal solution to $\mathbf{C}_{\boldsymbol{\delta}}(P_A^2, Q_A^2)$. Due to the linearity of expectation, $(f_1 + f_2)/2$ satisfies all the constraints of Equation (2.9) for $P_A = (P_A^1 + P_A^2)/2$ and $Q_A = (Q_A^1 + Q_A^2)/2$, i.e., $(f_1 + f_2)/2$ is feasible for $P_A = (P_A^1 + P_A^2)/2$ and $Q_A = (Q_A^1 + Q_A^2)/2$ with objective value $\left(\mathbf{C}_{\boldsymbol{\delta}}(P_A^1, Q_A^1) + \mathbf{C}_{\boldsymbol{\delta}}(P_A^2, Q_A^2)\right)/2$. Thus, we have

$$\mathbf{C}_{\boldsymbol{\delta}}\left(\frac{P_A^1 + P_A^2}{2}, \frac{Q_A^1 + Q_A^2}{2}\right) \leqslant \frac{1}{2}\left(\mathbf{C}_{\boldsymbol{\delta}}(P_A^1, Q_A^1) + \mathbf{C}_{\boldsymbol{\delta}}(P_A^2, Q_A^2)\right) \tag{B.84}$$

since $\mathbf{C}$ is a minimization problem. By definition, $\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A)$ is convex. QED.

Remark B.2. Since $\mathbf{C}_{\boldsymbol{\delta}}(P_A, Q_A)$ is defined on a compact $\mathbb{R}^2$ subspace, the convexity implies continuity. The continuity property is used in the following proof of Theorem 2.4.

Proof of Proposition 2.3. Here, we only prove the monotonicity for functions $x \mapsto \min_y \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ and $x \mapsto \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(x, y)$. The same statement for $y \mapsto \min_x \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ and $y \mapsto \min_x \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ is then straightforward due to the symmetry.

For simplification, we define $\mathbf{C}'_{\boldsymbol{\delta}} : x \mapsto \min_y \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ and let $\tilde{\mathbf{C}}_{\boldsymbol{\delta}} : x \mapsto \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(x, y)$. We notice that both functions can be exactly mapped to the constrained optimization problem $(\mathbf{C}')$ which removes the second constraint in Equation (2.9b) in $(\mathbf{C})$:

$$\underset{f}{\text{minimize}} \quad \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\delta} + \boldsymbol{\epsilon})] \tag{B.85a}$$

$$\text{s.t.} \quad \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon})] = x,\ 0 \leqslant f(\boldsymbol{\epsilon}) \leqslant 1 \quad \forall \boldsymbol{\epsilon} \sim \mathbb{R}^d. \tag{B.85b}$$

$\mathbf{C}'_{\boldsymbol{\delta}}(x)$ is the optimal objective to $(\mathbf{C}')$ and $\bar{\mathbf{C}}_{\boldsymbol{\delta}}(x)$ is $\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[f^*(\boldsymbol{\epsilon})]$ where f^* is the optimal solution.

Either based on Neyman-Pearson lemma [1933] or strong duality, $(\mathbf{C}')$ is equivalent to $(\mathbf{D}')$ defined as such:

$$\mathop{\Pr}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon}) < \lambda p(\boldsymbol{\epsilon} + \boldsymbol{\delta})] \tag{B.86a}$$

$$\text{s.t.} \quad \mathop{\Pr}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda p(\boldsymbol{\epsilon})] = x. \tag{B.86b}$$

For a given x, we only need to find λ satisfying Equation (B.86b). Then,

$$\mathbf{C}'_{\boldsymbol{\delta}}(x) = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon}) < \lambda p(\boldsymbol{\epsilon}+\boldsymbol{\delta})], \tag{B.87}$$

$$\bar{\mathbf{C}}_{\boldsymbol{\delta}}(x) = \Pr_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda p(\boldsymbol{\epsilon})]. \tag{B.88}$$

Now the monotonicity (what we would like to prove) is apparent. For $x_1 < x_2$, from Equation (B.86b), we have $\lambda_1 < \lambda_2$, since the probability density function p is non-negative. Thus, we inject λ_1 and λ_2 into Equation (B.87) and Equation (B.88), and yield

$$\mathbf{C}'_{\boldsymbol{\delta}}(x_1) \leqslant \mathbf{C}'_{\boldsymbol{\delta}}(x_2),\ \bar{\mathbf{C}}_{\boldsymbol{\delta}}(x_1) \leqslant \bar{\mathbf{C}}_{\boldsymbol{\delta}}(x_2), \tag{B.89}$$

which concludes the proof. QED.

Proof of Theorem 2.4. We discuss the cases according to the branching statement in the algorithm (Algorithm 2.2).

If $\underline{q} > \underline{Q_A}$,

- if $\underline{q} \leqslant \overline{Q_A}$, by definition we have $\mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, \underline{q}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, y)$ for arbitrary y. According to Proposition 2.3, we also have $\mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, \underline{q}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(x,\, y)$ for arbitrary $x \geqslant \underline{P_A}$ and arbitrary y. Given that $(\underline{P_A},\, \underline{q}) \in [\underline{P_A},\, \overline{P_A}] \times [\underline{Q_A},\, \overline{Q_A}]$, $(\underline{P_A},\, \underline{q})$ solves Equation (2.12);

- if $\underline{q} > \overline{Q_A}$, by convexity, $\mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, \overline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, y)$ for $y \in [\underline{Q_A},\, \overline{Q_A}]$.

  We further show that $\mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, \overline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(x,\, \overline{Q_A})$ for $x \in [\underline{P_A},\, \overline{P_A}]$: assume that this is not true, by Proposition 2.2, the function $y \mapsto \arg\min_x \mathbf{C}_{\boldsymbol{\delta}}(x,\, y)$ has function value larger than $\underline{P_A}$ at $y = \overline{Q_A}$. Since $\mathbf{C}_{\boldsymbol{\delta}}(0,\, 0) = 0$ is the global minimum of $\mathbf{C}_{\boldsymbol{\delta}}$, the function value at $y = 0$ is $x = 0$. By Proposition 2.3, there exists $y_0 \in [0,\, \overline{Q_A}]$ such that $\underline{P_A} = \arg\min_x \mathbf{C}_{\boldsymbol{\delta}}(x,\, y_0)$. Then, we get

$$\mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, y_0) \overset{(i.)}{\leqslant} \mathbf{C}_{\boldsymbol{\delta}}(\arg\min_x \mathbf{C}_{\boldsymbol{\delta}}(x,\, \overline{Q_A}),\, \overline{Q_A}) \overset{(ii.)}{\leqslant} \mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, \overline{Q_A}), \tag{B.90}$$

  where $(i.)$ follows from Proposition 2.3 for $y \mapsto \arg\min_x \mathbf{C}_{\boldsymbol{\delta}}(x,\, y)$; $(ii.)$ is implied in the meaning of $\arg\min_x \mathbf{C}_{\boldsymbol{\delta}}(x,\, \overline{Q_A})$. Since $y_0 \in [0,\, \overline{Q_A}]$, Equation (B.90) implies that $\underline{q}$ should be in $[0,\, \overline{Q_A}]$ as well, which violates the branching condition. Thus, $\mathbf{C}_{\boldsymbol{\delta}}(\underline{P_A},\, \overline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(x,\, \overline{Q_A})$ for $x \in [\underline{P_A},\, \overline{P_A}]$.

  Using Proposition 2.3 for function $x \mapsto \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(x,\, y)$ in interval $[\underline{P_A},\, \overline{P_A}]$ together with Proposition 2.2, we get $\mathbf{C}_{\boldsymbol{\delta}}(x,\, \overline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(x,\, y)$ for $x \in [\underline{P_A},\, \overline{P_A}]$ and $y \in [\underline{Q_A},\, \overline{Q_A}]$. Thus, $(\underline{P_A},\, \overline{Q_A})$ solves Equation (2.12).

If $\underline{q} \leqslant \underline{Q_A}$,

- if $\underline{p} \leqslant \overline{P_A}$, by definition we have $\mathbf{C}_{\boldsymbol{\delta}}(\max\{\underline{p}, \underline{P_A}\}, \underline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(x, \underline{Q_A})$ for $x \in [\underline{P_A}, \overline{P_A}]$. According to Proposition 2.3 and condition $\underline{q} \leqslant \underline{Q_A}$, we further have $\mathbf{C}_{\boldsymbol{\delta}}(\max\{\underline{p}, \underline{P_A}\}, \underline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(\max\{\underline{p}, \underline{P_A}\}, y) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ for arbitrary $x \in [\underline{P_A}, \overline{P_A}]$ and $y \in [\underline{Q_A}, \overline{Q_A}]$. Given that $(\max\{\underline{p}, \underline{P_A}\}, \underline{Q_A}) \in [\underline{P_A}, \overline{P_A}] \times [\underline{Q_A}, \overline{Q_A}]$, $(\underline{p}, \underline{Q_A})$ solves Equation (2.12);

- if $\underline{p} > \overline{P_A}$, according to Proposition 2.2, $\mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, \underline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(x, \underline{Q_A})$ for $x \in [\underline{P_A}, \overline{P_A}]$.

  We further show that $\mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, \underline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, y)$ for $y \in [\underline{Q_A}, \overline{Q_A}]$: assume that this is not true, by Proposition 2.2, the function $x \mapsto \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ has function value larger than $\underline{Q_A}$ at $x = \overline{P_A}$. Since $\mathbf{C}_{\boldsymbol{\delta}}(0, 0)$ is the global minimum, by Proposition 2.3 on $x \mapsto \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(x, y)$, there exists $x_0 \in [0, \overline{P_A}]$ such that $\underline{Q_A} = \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(x_0, y)$. Then, we get

$$\mathbf{C}_{\boldsymbol{\delta}}(x_0, \underline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, \arg\min_y \mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, y)) \leqslant \quad \mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, \underline{Q_A}) \tag{B.91}$$

  following the similar deduction as in Equation (B.90). Since $x_0 \in [0, \overline{P_A}]$, Equation (B.91) implies that $\underline{p}$ should be in $[0, \overline{P_A}]$ as well, which violates the branching condition. Thus, $\mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, \underline{Q_A}) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, y)$ for $y \in [\underline{Q_A}, \overline{Q_A}]$.

  Using Proposition 2.3 for function $y \mapsto \arg\min_x \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ in interval $[\underline{Q_A}, \overline{Q_A}]$ together with Proposition 2.2, we get $\mathbf{C}_{\boldsymbol{\delta}}(\overline{P_A}, y) \leqslant \mathbf{C}_{\boldsymbol{\delta}}(x, y)$ for $y \in [\underline{Q_A}, \overline{Q_A}]$ and $x \in [\underline{P_A}, \overline{P_A}]$. Thus, $(\overline{P_A}, \underline{Q_A})$ solves Equation (2.12).

QED.

B.2.3 Proof of Theorem 2.5

Proof of Theorem 2.5. We first define $r_p(\|\boldsymbol{\epsilon}\|_2) = p(\boldsymbol{\epsilon})$ and $r_q(\|\boldsymbol{\epsilon}\|_2) = q(\boldsymbol{\epsilon})$, then easily seen the concrete expressions of r_p and r_q are:

$$r_p(t) = \frac{1}{(2\sigma'^2)^{d/2-k}\pi^{d/2}} \cdot \frac{\Gamma(d/2)}{\Gamma(d/2-k)}, \tag{B.92}$$

$$r_q(t) = \frac{\nu}{(2\sigma'^2)^{d/2-k}\pi^{d/2}} \cdot \frac{\Gamma(d/2)}{\Gamma(d/2-k)}, \tag{B.93}$$

where

$$\nu := \frac{\Gamma(d/2-k)}{\gamma(d/2-k, \frac{T^2}{2\sigma'^2})} > 1 \tag{B.94}$$

and γ is the lower incomplete Gamma function.

Now we use level-set integration similar as the proof in Lemma B.2 to get the expressions of P, Q, and R respectively. Since $\mathcal{P}$ and $\mathcal{Q}$ are ℓ_2-symmetric, without loss of generality, we let $\boldsymbol{\delta} = (r, 0, \dots, 0)^\intercal$.

(P).

Suppose $P_T = r_p(T)$.

$$
\begin{aligned}
&P(\lambda_1, \lambda_2) \\
=&\Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}=\mathcal{N}^{\mathrm{g}}(k,\sigma)}[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] && \text{(B.95)}\\
=&\int_{\mathbb{R}^d} \mathbb{I}[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 q(\boldsymbol{x})] p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} && \text{(B.96)}\\
=&\int_0^{P_T} y\,\mathrm{d}y \int_{\substack{p(\boldsymbol{x})=y\\ p(\boldsymbol{x}-\boldsymbol{\delta})<\lambda_1 p(\boldsymbol{x})}} \frac{\mathrm{d}\boldsymbol{x}}{\|\nabla p(\boldsymbol{x})\|_2} + \int_{P_T}^{\infty} y\,\mathrm{d}y \int_{\substack{p(\boldsymbol{x})=y\\ p(\boldsymbol{x}-\boldsymbol{\delta})<(\lambda_1+\lambda_2\nu) p(\boldsymbol{x})}} \frac{\mathrm{d}\boldsymbol{x}}{\|\nabla p(\boldsymbol{x})\|_2} && \text{(B.97)}\\
=&\int_0^{P_T} y\,\mathrm{d}y \frac{2\pi^{d/2}}{\Gamma(d/2)} r_p^{-1}(y)^{d-1} \left(-\frac{1}{r_p'(r_p^{-1}(y))}\right) \cdot \Pr[p(\boldsymbol{x}-\boldsymbol{\delta}) \leqslant \lambda_1 p(\boldsymbol{x}) \,|\, p(\boldsymbol{x}) = y] + && \text{(B.98)}\\
&\int_{P_T}^{\infty} y\,\mathrm{d}y \frac{2\pi^{d/2}}{\Gamma(d/2)} r_p^{-1}(y)^{d-1} \left(-\frac{1}{r_p'(r_p^{-1}(y))}\right) \cdot \Pr[p(\boldsymbol{x}-\boldsymbol{\delta}) < (\lambda_1 + \lambda_2\nu) p(\boldsymbol{x}) \,|\, p(\boldsymbol{x}) = y] && \text{(B.99)}\\
\overset{y=r_p(t)}{=}&\int_T^{\infty} r_p(t)\,\mathrm{d}t \frac{2\pi^{d/2}}{\Gamma(d/2)} t^{d-1} \cdot \Pr[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) \,|\, \|\boldsymbol{x}\|_2 = t] + && \text{(B.100)}\\
&\int_0^{T} r_p(t)\,\mathrm{d}t \frac{2\pi^{d/2}}{\Gamma(d/2)} t^{d-1} \cdot \Pr[p(\boldsymbol{x}-\boldsymbol{\delta}) < (\lambda_1 + \lambda_2\nu) p(\boldsymbol{x}) \,|\, \|\boldsymbol{x}\|_2 = t] && \text{(B.101)}\\
=&\frac{1}{\Gamma(d/2-k)} \int_{T^2/(2\sigma'^2)}^{\infty} t^{d/2-k-1} \exp(-t)\,\mathrm{d}t \cdot \Pr[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) \,|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}] + && \text{(B.102)}\\
&\frac{1}{\Gamma(d/2-k)} \int_0^{T^2/(2\sigma'^2)} t^{d/2-k-1} \exp(-t)\,\mathrm{d}t \cdot \Pr[p(\boldsymbol{x}-\boldsymbol{\delta}) < (\lambda_1 + \lambda_2\nu) p(\boldsymbol{x}) \,|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}] && \text{(B.103)}\\
=&\mathop{\mathbb{E}}_{t\sim\Gamma(d/2-k,1)} \begin{cases} u_3(t, \lambda_1), & t \geqslant T^2/(2\sigma'^2) \\ u_3(t, \lambda_1 + \lambda_2\nu). & t < T^2/(2\sigma'^2) \end{cases} && \text{(B.104)}
\end{aligned}
$$

Here, $u_3(t, \lambda) = \Pr[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda p(\boldsymbol{x}) \,|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}]$.

(Q).

Similarly,

$$
\begin{aligned}
&Q(\lambda_1, \lambda_2) \\
=&\Pr_{\boldsymbol{\epsilon} \sim \mathcal{Q} = \mathcal{N}^{\mathrm{g}}_{\mathsf{trunc}}(k,T,\sigma)} [p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] && \text{(B.105)} \\
=&\int_{\|\boldsymbol{x}\|_2 \leqslant T} \mathbb{I}[p(\boldsymbol{x} - \boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 q(\boldsymbol{x})] q(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} && \text{(B.106)} \\
=&\int_{\nu P_T}^{\infty} y \,\mathrm{d}y \int_{\substack{q(\boldsymbol{x}) = y \\ p(\boldsymbol{x}-\boldsymbol{\delta}) < (\lambda_1 + \lambda_2 \nu) p(\boldsymbol{x})}} \frac{\mathrm{d}\boldsymbol{x}}{\|\nabla q(\boldsymbol{x})\|_2} && \text{(B.107)} \\
\overset{y = r_q(t)}{=}&\int_0^T r_q(t) \,\mathrm{d}t \frac{2\pi^{d/2}}{\Gamma(d/2)} t^{d-1} \cdot \Pr[p(\boldsymbol{x} - \boldsymbol{\delta}) < (\lambda_1 + \lambda_2 \nu) p(\boldsymbol{x}) \,|\, \|\boldsymbol{x}\|_2 = t] && \text{(B.108)} \\
=&\nu \mathop{\mathbb{E}}_{t \sim \Gamma(d/2-k,1)} u_3(t, \lambda_1 + \lambda_2 \nu) \cdot \mathbb{I}\left[t \leqslant \frac{T^2}{2\sigma'^2} \right]. && \text{(B.109)}
\end{aligned}
$$

(R).

Now, for R:

$$
\begin{aligned}
&R(\lambda_1, \lambda_2) \\
=&\Pr_{\boldsymbol{\epsilon} \sim \mathcal{P} = \mathcal{N}^{\mathrm{g}}(k,\sigma)} [p(\boldsymbol{\epsilon}) < \lambda_1 p(\boldsymbol{\epsilon} + \boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{\epsilon} + \boldsymbol{\delta})] && \text{(B.110)} \\
=&\int_{\mathbb{R}^d} \mathbb{I}[p(\boldsymbol{x}) < \lambda_1 p(\boldsymbol{x} + \boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x} + \boldsymbol{\delta})] p(\boldsymbol{x}) \,\mathrm{d}\boldsymbol{x} && \text{(B.111)} \\
=&\mathop{\mathbb{E}}_{t \sim \Gamma(d/2-k,1)} u_4(t, \lambda_1, \lambda_2). && \text{(B.112)}
\end{aligned}
$$

Here, $u_4(t, \lambda_1, \lambda_2) = \Pr[\lambda_1 p(\boldsymbol{x} + \boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x} + \boldsymbol{\delta}) > r_p(\sigma' \sqrt{2t}) \,|\, \|\boldsymbol{x}\|_2 = \sigma' \sqrt{2t}]$.

Plugging Lemma B.7 into $P(\lambda_1, \lambda_2)$ and $Q(\lambda_1, \lambda_2)$, and then plugging Lemma B.8 into $R(\lambda_1, \lambda_2)$, we yield the desired expressions in theorem statement. QED.

Lemma B.7. Under the condition of Theorem 2.5, let $\boldsymbol{\delta} = (r, 0, \ldots, 0)^\intercal$,

$$
\begin{aligned}
u_3(t, \lambda) &:= \Pr[p(\boldsymbol{x} - \boldsymbol{\delta}) < \lambda p(\boldsymbol{x}) \,|\, \|\boldsymbol{x}\|_2 = \sigma' \sqrt{2t}] \\
&= \mathrm{BetaCDF}_{\frac{d-1}{2}} \left(\frac{(r + \sigma' \sqrt{2t})^2}{4 r \sigma' \sqrt{2t}} - \frac{2k\sigma'^2 W(\frac{t}{k} e^{\frac{t}{k}} \lambda^{-\frac{1}{k}})}{4 r \sigma' \sqrt{2t}} \right),
\end{aligned} \tag{B.113}
$$

where W is the principal branch of Lambert W function.

Proof of Lemma B.7. $p(\boldsymbol{x} - \boldsymbol{\delta}) < \lambda p(\boldsymbol{x})$ means that $r_p(\|\boldsymbol{x} - \boldsymbol{\delta}\|_2) < \lambda r_p(\|\boldsymbol{x}\|_2)$ and therefore

$\|\boldsymbol{x}-\boldsymbol{\delta}\|_2 > r_p^{-1}(\lambda r_p(\|\boldsymbol{x}\|_2))$. Given that $\|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}$, we have

$$\begin{cases} x_1^2 + \sum_{i=2}^{d} x_i^2 = 2t\sigma'^2, \\ (x_1 - r)^2 + \sum_{i=2}^{d} x_i^2 \geqslant r_p^{-1}(\lambda r_p(\sigma'\sqrt{2t}))^2. \end{cases} \tag{B.114}$$

This is equivalent to

$$x_1 \leqslant \frac{2t\sigma'^2 + r^2 - r_p^{-1}(\lambda r_p(\sigma'\sqrt{2t}))^2}{2r}. \tag{B.115}$$

From the expression of r_p (Equation (B.92)), we have

$$r_p^{-1}(\lambda r_p(\sigma'\sqrt{2t}))^2 = 2\sigma'^2 k W\left(\frac{t}{k} e^{\frac{t}{k}} \lambda^{-\frac{1}{k}}\right). \tag{B.116}$$

Thus, when $\|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}$ and $\boldsymbol{x}$ uniformly sampled from this sphere,

$$\begin{aligned} & p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda p(\boldsymbol{x}) \\ \iff & x_1 \leqslant \frac{2t\sigma'^2 + r^2 - 2\sigma'^2 k W(\frac{t}{k} e^{\frac{t}{k}} \lambda^{-\frac{1}{k}})}{2r} \\ \iff & \frac{1 + \frac{x_1}{\sigma'\sqrt{2t}}}{2} \leqslant \frac{(r + \sigma'\sqrt{2t})^2 - 2\sigma'^2 k W\left(\frac{t}{k} e^{\frac{t}{k}} \lambda^{-\frac{1}{k}}\right)}{4r\sigma'\sqrt{2t}}. \end{aligned} \tag{B.117}$$

According to [427, Lemma I.23], for $\boldsymbol{x}$ uniformly sampled from sphere with radius $\sigma'\sqrt{2t}$, the component coordinate $\dfrac{1 + \frac{x_1}{\sigma'\sqrt{2t}}}{2} \sim \text{Beta}(\frac{d-1}{2}, \frac{d-1}{2})$. Combining Equation (B.117) with this result concludes the proof. QED.

Lemma B.8. Under the condition of Theorem 2.5, let $\boldsymbol{\delta} = (r, 0, \ldots, 0)^\intercal$,

$$\begin{aligned} u_4(t, \lambda_1, \lambda_2) &:= \Pr[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \,|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}] \\ &= \begin{cases} u_1(t), & \lambda_1 \leqslant 0 \\ u_1(t) + u_2(t), & \lambda_1 > 0 \end{cases} \end{aligned} \tag{B.118}$$

where

$$
\begin{aligned}
u_1(t) =& \text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{\min\{T^2, 2\sigma'^2 kW(\frac{t}{k}e^{\frac{t}{k}}(\lambda_1+\nu\lambda_2)^{\frac{1}{k}})\}}{4r\sigma'\sqrt{2t}} - \frac{(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right),\\
u_2(t) =& \max\left\{0, \text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{2\sigma'^2 kW(\frac{t}{k}e^{\frac{t}{k}}\lambda_1^{\frac{1}{k}}) - (\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right) - \text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2-(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right)\right\},
\end{aligned}
\tag{B.119}
$$

Proof of Lemma B.8. Under the condition that $\|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}$, we separate two cases: $q(\boldsymbol{x}+\boldsymbol{\delta}) > 0$ and $q(\boldsymbol{x}+\boldsymbol{\delta}) = 0$, which corresponds to $\|\boldsymbol{x}+\boldsymbol{\delta}\|_2 \leqslant T$ and $\|\boldsymbol{x}+\boldsymbol{\delta}\|_2 > T$.

(1) $q(\boldsymbol{x}+\boldsymbol{\delta}) > 0$:

Notice that

$$
q(\boldsymbol{x}+\boldsymbol{\delta}) > 0 \iff \|\boldsymbol{x}+\boldsymbol{\delta}\|_2^2 \leqslant T^2 \iff x_1 \leqslant \frac{T^2 - 2t\sigma'^2 - r^2}{2r}. \tag{B.120}
$$

From Equation (B.93), $q(\boldsymbol{x}+\boldsymbol{\delta}) = \nu p(\boldsymbol{x}+\boldsymbol{\delta})$. Thus,

$$
\begin{aligned}
&\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t})\\
\iff& (\lambda_1+\nu\lambda_2)p(\boldsymbol{x}+\boldsymbol{\delta}) \geqslant r_p(\sigma'\sqrt{2t})\\
\iff& \|\boldsymbol{x}+\boldsymbol{\delta}\|_2^2 \leqslant r_p^{-1}\left(\frac{r_p(\sigma'\sqrt{2t})}{\lambda_1+\nu\lambda_2}\right)^2\\
\iff& x_1 \leqslant \frac{r_p^{-1}\left(\frac{r_p(\sigma'\sqrt{2t})}{\lambda_1+\nu\lambda_2}\right)^2 - 2t\sigma'^2 - r^2}{2r}.
\end{aligned}
\tag{B.121}
$$

Therefore,

$$
\begin{aligned}
&\Pr[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \wedge q(\boldsymbol{x}+\boldsymbol{\delta}) > 0 \,|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}]\\
=& \Pr\left[x_1 \leqslant \frac{\min\left\{r_p^{-1}\left(\frac{r_p(\sigma'\sqrt{2t})}{\lambda_1+\nu\lambda_2}\right)^2, T^2\right\} - 2t\sigma'^2 - r^2}{2r} \,\middle|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}\right].
\end{aligned}
\tag{B.122}
$$

By [427, Lemma I.23] and Equation (B.116), we thus have

$$
\begin{aligned}
&\Pr[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \wedge q(\boldsymbol{x}+\boldsymbol{\delta}) > 0 \,|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}]\\
=& \text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{\min\{T^2, 2\sigma'^2 kW(\frac{t}{k}e^{\frac{t}{k}}(\lambda_1+\nu\lambda_2)^{\frac{1}{k}})\}}{4r\sigma'\sqrt{2t}} - \frac{(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right) = u_1(t).
\end{aligned}
\tag{B.123}
$$

(2) $q(\boldsymbol{x}+\boldsymbol{\delta}) = 0$:

Similarly,

$$q(\boldsymbol{x}+\boldsymbol{\delta})=0 \iff x_1 > \frac{T^2-2t\sigma'^2-r^2}{2r}. \tag{B.124}$$

When $\lambda_1 \leqslant 0$, the condition $\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) = \lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t})$ can never be satisfied. When $\lambda_1 > 0$, we have

$$\begin{aligned}
&\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \\
\iff &\|\boldsymbol{x}+\boldsymbol{\delta}\|_2^2 \leqslant r_p^{-1}\left(\frac{r_p(\sigma'\sqrt{2t})}{\lambda_1}\right)^2 \\
\iff &x_1 \leqslant \frac{r_p^{-1}\left(\frac{r_p(\sigma'\sqrt{2t})}{\lambda_1}\right)^2 - 2t\sigma'^2 - r^2}{2r}.
\end{aligned} \tag{B.125}$$

Therefore, when $\lambda_1 \leqslant 0$,

$$\Pr[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \wedge q(\boldsymbol{x}+\boldsymbol{\delta})=0 \,|\, \|\boldsymbol{x}\|_2=\sigma'\sqrt{2t}] = 0. \tag{B.126}$$

When $\lambda_1 > 0$, the condition that $lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t})$ is equivalent to

$$x_1 \in \left(\frac{T^2-2t\sigma'^2-r^2}{2r}, \frac{r_p^{-1}\left(\frac{r_p(\sigma'\sqrt{2t})}{\lambda_1}\right)^2 - 2t\sigma'^2 - r^2}{2r}\right]. \tag{B.127}$$

Again, by [427, Lemma I.23] and Equation (B.116), we have

$$\begin{aligned}
&\Pr[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \wedge q(\boldsymbol{x}+\boldsymbol{\delta})=0 \,|\, \|\boldsymbol{x}\|_2=\sigma'\sqrt{2t}] \\
=&\max\left\{0, \text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{2\sigma'^2 kW(\frac{t}{k}e^{\frac{t}{k}}\lambda_1^{\frac{1}{k}}) - (\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right) - \text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2-(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right)\right\} \\
=&u_2(t).
\end{aligned} \tag{B.128}$$

(3) Combining the two cases:

Now we are ready to combine the two cases.

$$\begin{aligned}
&\Pr[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \,|\, \|\boldsymbol{x}\|_2=\sigma'\sqrt{2t}] \\
=&\Pr[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \wedge q(\boldsymbol{x}+\boldsymbol{\delta})>0 \,|\, \|\boldsymbol{x}\|_2=\sigma'\sqrt{2t}]+ \\
&\Pr[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > r_p(\sigma'\sqrt{2t}) \wedge q(\boldsymbol{x}+\boldsymbol{\delta})=0 \,|\, \|\boldsymbol{x}\|_2=\sigma'\sqrt{2t}] \\
=&\begin{cases} u_1(t), & \lambda_1 \leqslant 0 \\ u_1(t)+u_2(t). & \lambda_1 > 0 \end{cases}
\end{aligned} \tag{B.129}$$

QED.

B.2.4 Discussion on Uniqueness of Feasible Pair

As we sketched in Section 2.5.3, in general cases, the pair that satisfies constraints in Equation (2.19) is unique. We formally state this finding and prove it in Theorem B.2.

Theorem B.2 (Uniqueness of Feasible Solution in Equation (2.19))**.** Under the same setting of Theorem 2.5, if $Q_A \in (0,1)$ and $P_A \in (Q_A/\nu, 1-(1-Q_A)/\nu)$, then there is the pair (λ_1, λ_2) that satisfies both $P(\lambda_1, \lambda_2) = P_A$ and $Q(\lambda_1, \lambda_2) = Q_A$ is unique.

We prove the theorem in the end of this section, which is based on the strict monotonicity of two auxiliary functions: $g(\lambda_1 + \nu\lambda_2) := Q(\lambda_1, \lambda_2)$ and $h(\lambda_1)$ (defined in Section 2.5.3). For other types of smoothing distributions $\mathcal{P}$ and $\mathcal{Q}$, in Theorem B.6 we characterize and prove a sufficient condition that guarantees the uniqueness of feasible pair.

We observe that the feasible region of (P_A, Q_A) is

$$\mathcal{R} = \{(x, y) : y/\nu \leqslant x \leqslant 1 - (1-y)/\nu, 0 \leqslant y \leqslant 1\}. \tag{B.130}$$

Therefore, the theorem states that when (P_A, Q_A) is an internal point of $\mathcal{R}$, the feasible solution is unique and we can use our proposed method to find out such a feasible solution and thus solve the dual problem (**D**). Now, the edge cases are that (P_A, Q_A) lies on the boundary of $\mathcal{R}$. We discuss all these cases and show that these boundary cases correspond to degenerate problems that are easy to solve respectively:

- $Q_A = 0$:
  When $Q_A = 0$, and $P_A \in (0,1)$ (otherwise, trivially $R(\lambda_1, \lambda_2) = P_A \in \{0,1\}$ solves (**D**)), we have $\lambda_1 + \nu\lambda_2 \to 0^+$ and thus $R(\lambda_1, \lambda_2) = \mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_2(t)$. Since $u_2(t)$ only involves λ_1, we only require λ_1 to be unique to deploy the method. Since $P_A = P(\lambda_1, \lambda_2) = \mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_3(t, \lambda_1) \cdot \mathbb{I}[t \geqslant T^2/(2\sigma'^2)]$ and $P_A \in (0,1)$, by similar arguments as in Theorem B.2, we know λ_1 is unique. Hence, all feasible pairs give the same $R(\lambda_1, \lambda_2)$, i.e., have the same objective value and the proposed method that computes a feasible one is sufficient for solving (**D**).

- $Q_A = 1$:
  When $Q_A = 1$ and $P_A \in (0,1)$, we observe that $u_1(t) \leqslant \text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2}{4r\sigma'\sqrt{2t}} - \frac{(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right)$ where equality is feasible with the selected $(\lambda_1 + \nu\lambda_2) \to +\infty$ and hence the maximum of $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_1(t)$ among all feasible (λ_1, λ_2) is a constant. On the other hand, since

$u_2(t)$ only involves λ_1 that is unique as discussed in "$Q_A = 0$" case, all feasible pairs give the same value of $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_2(t)$. As a result, the maximum of $R(\lambda_1, \lambda_2)$ can be computed by adding the unique value of $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_2(t)$ and the constant corresponding to the maximum of $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_1(t)$ among all feasible (λ_1, λ_2), which solves the dual problem (**D**).

- $P_A = Q_A/\nu$:
  We assume $P_A, Q_A \in (0, 1)$ (otherwise covered by former cases). In this case, λ_1 satisfies that $h(\lambda_1) = P_A - Q_A/\nu = 0$, so $\lambda_1 \to 0^+$. As a result, $u_2(t) = 0$ for all $t > 0$ and $R(\lambda_1, \lambda_2) = \mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_1(t)$. We observe that $u_1(t)$ is only related to $(\lambda_1 + \nu\lambda_2)$ where $(\lambda_1 + \nu\lambda_2)$ satisfying $Q(\lambda_1, \lambda_2) = Q_A$ is unique since $Q_A \in (0, 1)$. Thus, any feasible (λ_1, λ_2) would have the same $(\lambda_1 + \nu\lambda_2)$ and hence leads to the same $R(\lambda_1, \lambda_2)$. So the proposed method that finds one feasible (λ_1, λ_2) suffices for solving (**D**).

- $P_A = 1 - \frac{1-Q_A}{\nu}$:
  We again assume $P_A, Q_A \in (0, 1)$ (otherwise covered by former cases). In this case, λ_1 satisfies that $h(\lambda_1) = 1 - 1/\nu$. Since $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, \mathbb{I}[t < T^2/(2\sigma'^2)] = 1/\nu$, we know $u_3(t, \lambda_1) = 1$ for $t \geqslant T^2/(2\sigma'^2)$ except a zero-measure set, and thus $\lambda_1 \to +\infty$. As a result, $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_2(t) = 1 - \mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2-(\sigma'\sqrt{2t}-r)^2}{4r\sigma'\sqrt{2t}}\right)$ is a constant. Similar as "$P_A = Q_A/\nu$" case, feasible $(\lambda_1 + \nu\lambda_2)$ is unique. Therefore, feasible (λ_1, λ_2) leads to a unique $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_1(t)$. We compute out these two quantities $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_2(t)$ and $\mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_1(t)$ so as to obtain the unique $R(\lambda_1, \lambda_2)$ that solves (**D**).

We remark that in practice, we never observe any instances that correspond to these edge cases though we implemented these techniques for solving them.

Proof of Theorem B.2. The high-level proof sketch is implied in the derivation of our feasible (λ_1, λ_2) finding method introduced in Section 2.5.3. We first show $h(\lambda_1)$ is monotonically strictly increasing in a neighborhood of λ_1 where $h(\lambda_1) = P_A - Q_A/\nu$, so the λ_1 that satisfies $P_A - Q_A/\nu$ is unique. We then define $g(\gamma) = \nu\, \mathbb{E}_{t\sim\Gamma(d/2-k,1)}\, u_3(t, \gamma) \cdot \mathbb{I}[t \leqslant T^2/(2\sigma'^2)]$, and show its strict monotonicity around the neighborhood of λ^* that satisfies $g(\gamma^*) = Q_A$, which indicates that $(\lambda_1 + \nu\lambda_2)$ that satisfies Q_A is unique. Combining this two arguments, we know feasible (λ_1, λ_2) is unique. Note that the key in this proof is the local *strict* monotonicity, and the global (nonstrict) monotonicity for both $h(\cdot)$ and $g(\cdot)$ is apparent from their expressions. We now present the proofs for the local strict monotonicity of these two functions h and g.

(1) $h(\lambda_1)$ is monotonically strictly increasing in a neighborhood of λ_1 where $h(\lambda_1) = P_A - Q_A/\nu$.

From the theorem condition, we know that $h(\lambda_1) \in (0, 1 - 1/\nu)$. Thus, there exists $t_0 \geqslant T^2/(2\sigma'^2)$, such that $u_3(t_0, \lambda_1) \in (0, 1)$ (otherwise $h(\lambda_1) = 0$ or $1 - 1/\nu$). From the closed-form equation of u_3, for any neighboring $\lambda_1' \neq \lambda_1$, we will have $W(t_0/ke^{t_0/k}\lambda_1^{-1/k}) \neq W(t_0/ke^{t_0/k}\lambda_1'^{-1/k})$ by the monotonicity of Lambert W function. On the other hand, since $u_3(t_0, \lambda_1) \in (0, 1)$ and $\text{BetaCDF}_{\frac{d-1}{2}}(\cdot)$ is also strict monotonic in the neighborhood, we have $u_3(t_0, \lambda_1') \neq u_3(t_0, \lambda_1)$. Same happens to t_0's neighborhood, i.e., $\exists \delta > 0$, s.t., $\forall t \in (t_0 - \delta, t_0 + \delta)$, $u_3(t, \lambda_1') \neq u_3(t, \lambda_1)$ and $\text{sgn}(u_3(t, \lambda_1') - u_3(t, \lambda_1))$ is consistent for any $t \in (t_0 - \delta, t_0 + \delta)$. As a result, $h(\lambda_1) \neq h(\lambda_1')$. By definition and the fact that $h(\cdot)$ is monotonically non-decreasing, the argument is proved.

(2) $g(\gamma)$ is monotonically strictly increasing in a neighborhood of γ^* where $g(\gamma^*) = Q_A$.

Since $Q_A \in (0, 1)$ by the theorem condition, we know that there exists $t_0 \in (0, T^2/(2\sigma'^2))$, such that $u_3(t_0, \gamma^*) \in (0, 1)$ (otherwise $g(\gamma^*) = 0$ or 1, which contradicts the theorem condition). Following the same reasoning as in (1)'s proof, for any $\gamma' \neq \gamma^*$ that lies in a sufficiently small neighborhood of γ^*, we have $u_3(t_0, \gamma^*) \neq u_3(t_0, \gamma')$, and $\exists \delta > 0$, s.t., $\forall t \in (t_0 - \delta, t_0 + \delta)$, $u_3(t, \gamma^*) \neq u_3(t, \gamma')$ and $\text{sgn}(u_3(t, \gamma^*) - u_3(t, \gamma'))$ is consistent for any $t \in (t_0 - \delta, t_0 + \delta)$. As a result, $g(\gamma^*) \neq g(\gamma')$. By definition and the fact that $g(\cdot)$ is monotonically non-decreasing, the argument is proved. QED.

B.3 EXTENSIONS OF DSRS COMPUTATIONAL METHODS

In this appendix, we exemplify a few extensions of DSRS computational method.

B.3.1 Certification with Standard and Truncated Standard Gaussian

In main text and Theorem 2.5, we focus on DSRS certification with generalized Gaussian as $\mathcal{P}$ and truncated generalized Gaussian as $\mathcal{Q}$, which has theoretical advantages (Theorem 2.2). On the other hand, DSRS can also be applied to other distributions. Concretely, to certify robustness with standard Gaussian as $\mathcal{P}$ and truncated standard Gaussian as $\mathcal{Q}$, we can directly plug the following theorem's numerical integration expressions into the described DSRS algorithm (Algorithm 2.1).

Theorem B.3. In $\mathbf{D}_{\boldsymbol{\delta}}(P_A, Q_A)$, let $r = \|\boldsymbol{\delta}\|_2$, when $\mathcal{P} = \mathcal{N}(\sigma)$ and $\mathcal{Q} = \mathcal{N}_{\mathsf{trunc}}(T, \sigma)$, let $\nu := \Gamma\text{CDF}_{d/2}(T^2/(2\sigma^2))^{-1}$,

$$
\begin{aligned}
R(\lambda_1, \lambda_2) :=& \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon}) < \lambda_1 p(\boldsymbol{\epsilon}+\boldsymbol{\delta}) + \lambda_2 q(v\epsilon+\boldsymbol{\delta})] \\
=& \begin{cases} \mathbb{E}_{t\sim\Gamma(d/2,1)}\, u_1(t), & \lambda_1 \leqslant 0 \\ \mathbb{E}_{t\sim\Gamma(d/2,1)}\, u_1(t) + u_2(t), & \lambda_1 > 0 \end{cases} \text{ where} \\
u_1(t) =& \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{\min\{T^2, 2t\sigma^2 + 2\sigma^2\ln(\lambda_1+\nu\lambda_2)\}}{4r\sigma\sqrt{2t}} - \frac{(\sigma\sqrt{2t}-r)^2}{4r\sigma\sqrt{2t}}\right), \\
u_2(t) =& \max\left\{0, \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{2t\sigma^2 + 2\sigma^2\ln\lambda_1 - (\sigma\sqrt{2t}-r)^2}{4r\sigma\sqrt{2t}}\right) - \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{T^2-(\sigma\sqrt{2t}-r)^2}{4r\sigma\sqrt{2t}}\right)\right\}. \\
P(\lambda_1, \lambda_2) :=& \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] \\
=& \mathop{\mathbb{E}}_{t\sim\Gamma(d/2,1)} \begin{cases} u_3(t, \lambda_1), & t \geqslant T^2/(2\sigma^2) \\ u_3(t, \lambda_1+\nu\lambda_2), & t < T^2/(2\sigma^2). \end{cases} \text{ where} \\
u_3(t,\lambda) =& \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{1}{2} + \frac{r^2 + 2\sigma^2\ln\lambda}{4r\sigma\sqrt{2t}}\right). \\
Q(\lambda_1, \lambda_2) :=& \Pr_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{\epsilon}) + \lambda_2 q(\boldsymbol{\epsilon})] \\
=& \nu \mathop{\mathbb{E}}_{t\sim\Gamma(d/2,1)} u_3(t, \lambda_1+\nu\lambda_2)\cdot\mathbb{I}[t \leqslant T^2/(2\sigma^2)].
\end{aligned}
\tag{B.131}
$$

In above equations, $\Gamma(d/2, 1)$ is gamma distribution and $\Gamma\mathrm{CDF}_{d/2}$ is its CDF, and $\mathrm{BetaCDF}_{\frac{d-1}{2}}$ is the CDF of distribution $\mathrm{Beta}(\frac{d-1}{2}, \frac{d-1}{2})$.

Proof of Theorem B.3. The proof largely follows the same procedure as the proof of Theorem 2.5. The only difference is that, since $\mathcal{P} = \mathcal{N}(\sigma)$, let $r_p(\|\boldsymbol{\epsilon}\|_2) = p(\boldsymbol{\epsilon})$, different from Equation (B.116), now

$$
\begin{aligned}
& r_p^{-1}(\lambda r_p(\sigma\sqrt{2t}))^2 \\
=& -2\sigma^2\ln(\lambda r_p(\sigma\sqrt{2t})\cdot(2\pi\sigma^2)^{d/2}) \\
=& -2\sigma^2\ln\left(\frac{\lambda}{(2\pi\sigma^2)^{d/2}}\exp\left(-\frac{2t\sigma^2}{2\sigma^2}\right)\cdot(2\pi\sigma^2)^{d/2}\right) \\
=& -2\sigma^2(\ln\lambda - t) = 2t\sigma^2 - 2\sigma^2\ln\lambda.
\end{aligned}
\tag{B.132}
$$

By plugging this equation into the proof of Theorem 2.5, we prove Theorem B.3. QED.

In practice, DSRS with standard Gaussian and truncated standard Gaussian as smoothing distributions gives marginal improvements over Neyman-Pearson-based certification. This is because, for standard Gaussian distribution, the noise magnitude is particularly concentrated on a thin shell as reflected by the green curve in Figure 2.3. As a result, the truncated standard Gaussian as $\mathcal{Q}$ either has a tiny density overlap with $\mathcal{P}$ or provides highly similar

Table B.2: The numerical integration expression for P, Q, and R (see definition in Theorem 2.5). See Appendix B.3.2 for notation description.

$$P(\lambda_1,\,\lambda_2) = \mathop{\mathbb{E}}_{x\sim\Gamma(\frac{d}{2}-k)} \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{(r+\sigma'\sqrt{2x})^2 - g^{-1}(\lambda_1 g(\sigma'\sqrt{2x}) + \lambda_2 h(\sigma'\sqrt{2x}))^2}{4r\sigma'\sqrt{2x}}\right)$$

$$Q(\lambda_1,\,\lambda_2) = \mathop{\mathbb{E}}_{x\sim\Gamma(\frac{d}{2}-k)} \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{(r+\beta'\sqrt{2x})^2 - g^{-1}(\lambda_1 g(\beta'\sqrt{2x}) + \lambda_2 h(\beta'\sqrt{2x}))^2}{4r\beta'\sqrt{2x}}\right)$$

$$R(\lambda_1,\,\lambda_2) = \mathop{\mathbb{E}}_{x\sim\Gamma(\frac{d}{2}-k)} \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{1}{2} + \frac{\overline{(\lambda_1 g + \lambda_2 h)^{-1}}(g(\sigma'\sqrt{2x}))^2 - r^2 - 2x\sigma'^2}{4r\sigma'\sqrt{2x}}\right) - \mathrm{BetaCDF}_{\frac{d-1}{2}}\left(\frac{1}{2} + \frac{\underline{(\lambda_1 g + \lambda_2 h)^{-1}}(g(\sigma'\sqrt{2x}))^2 - r^2 - 2x\sigma'^2}{4r\sigma'\sqrt{2x}}\right)$$

Table B.3: Simplified terms in numerical integration for P and Q, where W is the real-valued branch of the Lambert W function.

Functions		$P(\lambda_1,\,\lambda_2)$	$Q(\lambda_1,\,\lambda_2)$
Term to Simplify		$g^{-1}\left(\lambda_1 g(\sigma'\sqrt{2x}) + \lambda_2 h(\sigma'\sqrt{2x})\right)^2$	$g^{-1}\left(\lambda_1 g(\beta'\sqrt{2x}) + \lambda_2 h(\beta'\sqrt{2x})\right)^2$
Simplified Terms	$k=0$	$-2\sigma'^2 \ln\left(\lambda_1 \exp(-x) + \lambda_2 \left(\frac{\sigma'}{\beta'}\right)^d \exp\left(-\frac{\sigma'^2}{\beta'^2}x\right)\right)$	$-2\sigma'^2 \ln\left(\lambda_1 \exp\left(-\frac{\beta'^2}{\sigma'^2}x\right) + \lambda_2 \left(\frac{\sigma'}{\beta'}\right)^d \exp(-x)\right)$
	$k>0$	$2k\sigma'^2 W\left(\frac{x}{k}\left(\lambda_1 \exp(-x) + \lambda_2 \left(\frac{\sigma'}{\beta'}\right)^{d-2k} \exp\left(-\frac{\sigma'^2}{\beta'^2}x\right)\right)^{-1/k}\right)$	$2k\sigma'^2 W\left(\frac{x}{k}\cdot\frac{\beta'^2}{\sigma'^2}\left(\lambda_1 \exp\left(-\frac{\beta'^2}{\sigma'^2}\right) + \lambda_2 \left(\frac{\sigma'}{\beta'}\right)^{d-2k} \exp(-x)\right)^{-1/k}\right)$

information (i.e., $Q_A \approx P_A$). In either case, $\mathcal{Q}$ provides little additional information. Therefore, in practice, we do not use standard Gaussian and truncated standard Gaussian as $\mathcal{P}$ and $\mathcal{Q}$, which is also justified by Theorem 2.2, though DSRS can provide certification for this setting.

B.3.2 Certification with Generalized Gaussian with Different Variances

We now consider the robustness certification with smoothing distribution $\mathcal{P} = \mathcal{N}^{\mathtt{g}}(k,\sigma)$ and additional smoothing distribution $\mathcal{Q} = \mathcal{N}^{\mathtt{g}}(k,\beta)$ where σ and β are different (i.e., different variance).

Computational Method Description

Hereinafter, for this $\mathcal{P}$ and $\mathcal{Q}$ we define the radial density function $g(r) := p(\boldsymbol{x})$ and $h(r) := q(\boldsymbol{x})$ for any $\|\boldsymbol{x}\|_2 = r$, where p and q are the density functions of $\mathcal{P}$ and $\mathcal{Q}$ respectively. $\overline{(\lambda_1 g + \lambda_2 h)^{-1}}(x) := \max y \quad \text{s.t. } \lambda_1 g(y) + \lambda_2 h(y) = x$ and similarly $\underline{(\lambda_1 g + \lambda_2 h)^{-1}}(x) := \min y \quad \text{s.t. } \lambda_1 g(y) + \lambda_2 h(y) = x$.

In this case, we can still have the numerical expression for $P(\lambda_1,\lambda_2)$, $Q(\lambda_1,\lambda_2)$, and $R(\lambda_1,\lambda_2)$ as shown in Theorem B.4.

Theorem B.4. When the smoothing distributions $\mathcal{P} = \mathcal{N}^{\mathtt{g}}(k,\sigma)$ and additional smoothing distribution $\mathcal{Q} = \mathcal{N}^{\mathtt{g}}(k,\beta)$, let $P(\lambda_1,\lambda_2)$, $Q(\lambda_1,\lambda_2)$, and $R(\lambda_1,\lambda_2)$ be as defined in Theorem 2.5, then P, Q, and R can be computed by expressions in Table B.2.

In Table B.2, the numerical integration requires the computation of several inverse functions. In this subsection, we simplify the numerical integration expressions for P and Q by deriving the closed forms of these inverse functions, as shown in Table B.3. In the actual implementation of numerical integration, for P and Q, we use these simplified expressions to compute; for R, benefited from the unimodality (Lemma B.10), we deploy a simple binary search to compute.

Theorem B.5. When the smoothing distributions $\mathcal{P} = \mathcal{N}^{\mathfrak{g}}(k, \sigma)$ and $\mathcal{Q} = \mathcal{N}^{\mathfrak{g}}(k, \beta)$, the terms $g^{-1}(\lambda_1 g(\sigma'\sqrt{2x}) + \lambda_2 h(\sigma'\sqrt{2x}))^2$ and $g^{-1}(\lambda_1 g(\beta'\sqrt{2x}) + \lambda_2 h(\beta'\sqrt{2x}))^2$ in $P(\lambda_1, \lambda_2)$ and $Q(\lambda_1, \lambda_2)$'s computational expressions (see Table B.2) are equivalent to those shown in Table B.3.

With the method to compute P, Q, and R for given λ_1 and λ_2, now the challenge is to solve λ_1 and λ_2 such that $P(\lambda_1, \lambda_2) = P_A$ and $Q(\lambda_1, \lambda_2) = Q_A$.

Luckily, as Theorem B.6 shows, for given P_A and Q_A, such (λ_1, λ_2) pair is unique. Indeed, such uniqueness holds not only for this $\mathcal{P}$ and $\mathcal{Q}$ but also for a wide range of smoothing distributions.

Theorem B.6 (Uniqueness)**.** Suppose distributions $\mathcal{P}$ and $\mathcal{Q}$'s are ℓ_p-spherically symmetric, i.e., there exists radial density functions g and h such that $p(\boldsymbol{x}) = g(\|\boldsymbol{x}\|_p)$ and $q(\boldsymbol{x}) = h(\|\boldsymbol{x}\|_p)$, then if g and h are continuous and $\frac{g}{h}$ is continuous and strictly monotonic almost everywhere, for any given $(P_A, Q_A) \in \mathbb{R}_+^2$, there is at most one (λ_1, λ_2) pair satisfying constraint of Equation (2.19).

The proof is shown in the next subsection, which is based on Cauchy's mean value theorem of the probability integral.

With Theorem B.6 and Proposition 2.3, we can use joint binary search as shown in Algorithm B.1 to find λ_1 and λ_2 that can be viewed as the intersection of two curves. At a high level, Each time, we leverage the monotonicity to get a point $(\lambda_1^{mid}, \lambda_2^{mid})$ on the P's curve, then compute corresponding Q, and update the binary search intervals based on whether $Q(\lambda_1^{mid}, \lambda_2^{mid}) > Q_A$. We shrink the intervals for both λ_1 and λ_2 (Lines 5 and 7) in Algorithm B.1 to accelerate the search. The algorithm is plugged into Line 8 of Algorithm 2.1.

Algorithm B.1: DUALBINARYSEARCH for λ_1 and λ_2.

Data: Query access to $P(\cdot,\,\cdot)$ and $Q(\cdot,\,\cdot)$, P_A and Q_A

Result: λ_1 and λ_2 satisfying constraints $P(\lambda_1, \lambda_2) = P_A, Q(\lambda_1, \lambda_2) = Q_A$

$\lambda_1^L \leftarrow -M, \lambda_1^U \leftarrow +M, \lambda_2^L \leftarrow -M, \lambda_2^U \leftarrow +M$; /* M is a large positive number */

while $\lambda_1^U - \lambda_1^L > eps$ **do**

  $\lambda_1^{mid} \leftarrow (\lambda_1^L + \lambda_1^U)/2$;

  Binary search for $\lambda_2^{mid} \in [\lambda_2^L,\, \lambda_2^U]$ such that $P(\lambda_1^{mid},\, \lambda_2^{mid}) = P_A$; /* $(\lambda_1^{mid},\, \lambda_2^{mid})$ lies on red curve */

  if $Q(\lambda_1^{mid},\, \lambda_2^{mid}) < Q_A$ **then**

    $\lambda_1^U \leftarrow \lambda_1^{mid}, \lambda_2^L \leftarrow \lambda_2^{mid}$; /* $(\lambda_1^{mid},\, \lambda_2^{mid})$ is right to intersection */

  else

    $\lambda_1^L \leftarrow \lambda_1^{mid}, \lambda_2^U \leftarrow \lambda_2^{mid}$; /* $(\lambda_1^{mid},\, \lambda_2^{mid})$ is left to intersection */

end

return $(\lambda_1^L,\, \lambda_2^L)$; /* for soundness: $R(\lambda_1^L,\, \lambda_2^L)$ lower bounds (**D**) */

Proofs

Proof of Theorem B.4. The ℓ_2-radial density functions of $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ have these expressions: $g(r) \propto r^{-k} \exp(-r^2/(2\sigma'^2))$ and $h(r) \propto r^{-k} \exp(-r^2/(2\beta'^2))$. When r increases, r^{-k}, $\exp(-r^2/(2\sigma'^2))$, and $\exp(-r^2/(2\beta'^2))$ decrease so that g and h are both strictly decreasing. Therefore, they have inverse functions, which are denoted by g^{-1} and h^{-1}. Now we are ready to derive the expressions.

(I.) We start with P:

$$
\begin{aligned}
&P(\lambda_1,\, \lambda_2) \\
&= \int_{\mathbb{R}^d} \mathbb{I}\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 q(\boldsymbol{x})\right] p(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} && \text{(B.133)} \\
&\overset{(1.)}{=} \int_0^\infty y\, \mathrm{d}y \int_{\substack{p(\boldsymbol{x})=y \\ p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 h(\boldsymbol{x})}} \frac{\mathrm{d}\boldsymbol{x}}{\|\nabla p(\boldsymbol{x})\|_2} && \text{(B.134)} \\
&\overset{(2.)}{=} \int_0^\infty y\, \mathrm{d}y \int_{\substack{p(\boldsymbol{x})=y \\ p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 h(\boldsymbol{x})}} -\frac{\mathrm{d}\boldsymbol{x}}{g'\left(g^{-1}(y)\right)} && \text{(B.135)} \\
&\overset{(3.)}{=} \int_0^\infty y\, \mathrm{d}y \cdot \frac{\pi^{d/2} d g^{-1}(y)^{d-1}}{(d/2)!} \left(-\frac{\mathrm{d}\boldsymbol{x}}{g'(g^{-1}(y))}\right) \cdot \Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 q(\boldsymbol{x}) \,\middle|\, p(\boldsymbol{x}) = y\right] && \text{(B.136)} \\
&\overset{(4.)}{=} \int_0^\infty g(t)\, \mathrm{d}t \cdot \frac{\pi^{d/2} d t^{d-1}}{(d/2)!} \cdot \Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(t) + \lambda_2 h(t) \,\middle|\, \|\boldsymbol{x}\|_2 = t\right] && \text{(B.137)}
\end{aligned}
$$

$$\overset{(5.)}{=} \int_0^\infty \frac{1}{(2\sigma'^2)^{\frac{d}{2}-k}\pi^{\frac{d}{2}}} \cdot \frac{\Gamma(d/2)}{\Gamma(d/2-k)} \cdot t^{-2k} \exp\left(-\frac{t^2}{2\sigma'^2}\right) \cdot \frac{\pi^{d/2} dt^{d-1}}{(d/2)!} \cdot \tag{B.138}$$

$$\Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(t) + \lambda_2 h(t) \,\middle|\, \|\boldsymbol{x}\|_2 = t\right] \tag{B.139}$$

$$= \int_0^\infty \frac{2}{(2\sigma'^2)^{\frac{d}{2}-k}\Gamma(\frac{d}{2}-k)} t^{d-2k-1} \exp\left(-\frac{t^2}{2\sigma'^2}\right) \cdot \Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(t) + \lambda_2 h(t) \,\middle|\, \|\boldsymbol{x}\|_2 = t\right] \tag{B.140}$$

$$\overset{(6.)}{=} \frac{1}{(2\sigma'^2)^{\frac{d}{2}-k}\Gamma(\frac{d}{2}-k)} \int_0^\infty t^{\frac{d}{2}-k-1} \exp\left(-\frac{t}{2\sigma'^2}\right) \cdot \tag{B.141}$$

$$\Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(\sqrt{t}) + \lambda_2 h(\sqrt{t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \sqrt{t}\right] \tag{B.142}$$

$$\overset{(7.)}{=} \frac{1}{\Gamma(\frac{d}{2}-k)} \int_0^\infty t^{\frac{d}{2}-2k-1} \exp(-t) \cdot \Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(\sigma'\sqrt{2t}) + \lambda_2 h(\sigma'\sqrt{2t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}\right] \tag{B.143}$$

$$\overset{(8.)}{=} \mathop{\mathbb{E}}_{t\sim\Gamma(\frac{d}{2}-k)} \Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(\sigma'\sqrt{2t}) + \lambda_2 h(\sigma'\sqrt{2t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}\right]. \tag{B.144}$$

As a reminder, d is the input dimension. The $\Gamma(\cdot)$ here refer to either the Gamma distribution (in $t \sim \Gamma(\frac{d}{2}-k)$) or Gamma function (in some denominators). In the above integration: (1.) uses level-set sliced integration as first proposed in [427]; (2.) leverages the fact that $p(\boldsymbol{x})$ is ℓ_2-symmetric and $g'(\cdot) < 0$; (3.) injects the surface area of ℓ_2-sphere with radius $g^{-1}(y)$; (4.) alters the integral variable: $t = g^{-1}(y)$, which yields $\mathrm{d}t = \mathrm{d}y/g'(t) = \mathrm{d}y/g'(g^{-1}(y))$ and $y = g(t)$; (5.) injects the expression of $g(t)$; (6.) alters the integral variable from t to t^2; (7.) does re-scaling; and (8.) observes that the integral can be expressed by expectation over Gamma distribution.

Due to the isotropy, let $r = \|\boldsymbol{\delta}\|_2$, we can deem $\boldsymbol{\delta} = (r, 0, \ldots, 0)^\intercal$ by rotating the axis. Then we simplify the probability term by observing that

$$\begin{aligned}
&\begin{cases} \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t} \\ p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(\sigma'\sqrt{2t}) + \lambda_2 h(\sigma'\sqrt{2t}) \end{cases} \\
\iff &\begin{cases} x_1^2 + \sum_{i=2}^d x_i^2 = 2t\sigma'^2 \\ (x_1 - r)^2 + \sum_{i=2}^d x_i^2 \geqslant g^{-1}\left(\lambda_1 g(\sigma'\sqrt{2t}) + \lambda_2 h(\sigma'\sqrt{2t})\right)^2 \end{cases} \\
\Longrightarrow &x_1 \leqslant \frac{2t\sigma'^2 - g^{-1}\left(\lambda_1 g(\sigma'\sqrt{2t}) + \lambda_2 h(\sigma'\sqrt{2t})\right)^2 + r^2}{2r}.
\end{aligned} \tag{B.145}$$

Lemma B.9 (Lemma I.23; [427]). If $(x_1, \cdots, x_d)$ is sampled uniformly from the unit sphere $\mathbb{S}^{d-1} \subseteq \mathbb{R}^d$, then

$$\frac{1+x_1}{2} \text{ is distributed as Beta}\left(\frac{d-1}{2}, \frac{d-1}{2}\right). \tag{B.146}$$

Combining Lemma B.9 and Equation (B.145), we get

$$\begin{aligned}&\Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(\sigma'\sqrt{2t}) + \lambda_2 h(\sigma'\sqrt{2t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}\right] \\ =&\text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{(r+\sigma'\sqrt{2t})^2}{4r\sigma'\sqrt{2t}} - \frac{g^{-1}\left(\lambda_1 g(\sigma'\sqrt{2t}) + \lambda_2 h(\sigma'\sqrt{2t})\right)^2}{4r\sigma'\sqrt{2t}}\right).\end{aligned} \tag{B.147}$$

Injecting Equation (B.147) into (8.) yields the expression shown in Table B.2.

(II.) The integration for Q is similar:

$$\begin{aligned}&Q(\lambda_1,\, \lambda_2) \\ =&\int_{\mathbb{R}^d} \mathbb{I}[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 q(x)] q(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} && \text{(B.148)} \\ =&\int_0^\infty y\, \mathrm{d}y \int_{\substack{q(\boldsymbol{x})=y \\ p(\boldsymbol{x}-\boldsymbol{\delta})<\lambda_1 p(\boldsymbol{x})+\lambda_2 q(\boldsymbol{x})}} \frac{\mathrm{d}\boldsymbol{x}}{\|\nabla q(\boldsymbol{x})\|_2} && \text{(B.149)} \\ =&\int_0^\infty y\, \mathrm{d}y \int_{\substack{q(\boldsymbol{x})=y \\ p(\boldsymbol{x}-\boldsymbol{\delta})<\lambda_1 p(\boldsymbol{x})+\lambda_2 q(\boldsymbol{x})}} -\frac{\mathrm{d}\boldsymbol{x}}{h'(h^{-1}(y))} && \text{(B.150)} \\ =&\int_0^\infty h(t)\, \mathrm{d}t \cdot \frac{\pi^{d/2} d t^{d-1}}{(d/2)!} \cdot \Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 q(\boldsymbol{x}) \,\middle|\, \|\boldsymbol{x}\|_2 = t\right] && \text{(B.151)} \\ =&\frac{1}{(2\beta'^2)^{\frac{d}{2}-k}\Gamma(\frac{d}{2}-k)} \int_0^\infty t^{\frac{d}{2}-k-1} \exp\left(-\frac{t}{2\beta'^2}\right) \cdot \Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(\sqrt{t}) + \lambda_2 h(\sqrt{t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \sqrt{t}\right] \\ & && \text{(B.152)} \\ =&\mathop{\mathbb{E}}_{t\sim\Gamma(\frac{d}{2}-k)} \Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(\beta'\sqrt{2t}) + \lambda_2 h(\beta'\sqrt{2t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \beta'\sqrt{2t}\right], && \text{(B.153)}\end{aligned}$$

where

$$\begin{aligned}&\Pr\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 g(\beta'\sqrt{2t}) + \lambda_2 h(\beta'\sqrt{2t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \beta'\sqrt{2t}\right] \\ =&\text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{(r+\beta'\sqrt{2t})^2}{4r\beta'\sqrt{2t}} - \frac{g^{-1}\left(\lambda_1 g(\beta'\sqrt{2t}) + \lambda_2 h(\beta'\sqrt{2t})\right)^2}{4r\beta'\sqrt{2t}}\right).\end{aligned} \tag{B.154}$$

(III.) Finally, we derive the integration for R:

$$
\begin{aligned}
& R(\lambda_1,\, \lambda_2) \\
=& \int_{\mathbb{R}^d} \mathbb{I}\left[p(\boldsymbol{x}-\boldsymbol{\delta}) < \lambda_1 p(\boldsymbol{x}) + \lambda_2 q(\boldsymbol{x})\right] p(\boldsymbol{x}-\boldsymbol{\delta})\,\mathrm{d}\boldsymbol{x} && \text{(B.155)} \\
=& \int_{\mathbb{R}^d} \mathbb{I}\left[p(\boldsymbol{x}) < \lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta})\right] p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} && \text{(B.156)} \\
=& \int_0^\infty y\,\mathrm{d}y \int_{\substack{p(\boldsymbol{x})=y \\ p(\boldsymbol{x})<\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta})}} \frac{\mathrm{d}\boldsymbol{x}}{\|\nabla p(\boldsymbol{x})\|_2} && \text{(B.157)} \\
=& \int_0^\infty y\,\mathrm{d}y \int_{\substack{p(\boldsymbol{x})=y \\ p(\boldsymbol{x})<\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta})}} -\frac{\mathrm{d}\boldsymbol{x}}{g'(g^{-1}(y))} && \text{(B.158)} \\
=& \int_0^\infty g(t)\,\mathrm{d}t \cdot \frac{\pi^{d/2} dt^{d-1}}{(d/2)!} \cdot \Pr\left[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > p(\boldsymbol{x}) \,\middle|\, \|\boldsymbol{x}\|_2 = t\right] && \text{(B.159)} \\
=& \frac{1}{(2\sigma'^2)^{\frac{d}{2}-k}\Gamma(\frac{d}{2}-k)} \int_0^\infty t^{\frac{d}{2}-k-1} \exp\left(-\frac{t}{2\sigma'^2}\right) \cdot && \text{(B.160)} \\
& \Pr\left[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > g(\sqrt{t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \sqrt{t}\right] && \text{(B.161)} \\
\overset{(9.)}{=}& \mathop{\mathbb{E}}_{t\sim\Gamma(\frac{d-k}{2})} \Pr\left[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > g(\sigma'\sqrt{2t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}\right]. && \text{(B.162)}
\end{aligned}
$$

To simplify the probability term, this time we have

$$
\begin{aligned}
& \begin{cases} \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}, \\ \lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta}) + \lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > g(\sigma'\sqrt{2t}) \end{cases} && \text{(B.163)} \\
\iff & \begin{cases} x_1^2 + \sum_{i=2}^d x_i^2 = 2t\sigma'^2 \\ (\lambda_1 g + \lambda_2 h)\left(\sqrt{(x_1+r)^2 + \sum_{i=2}^d x_i^2}\right) > g(\sigma'\sqrt{2t}) \end{cases} && \text{(B.164)} \\
\Longrightarrow & (\lambda_1 g + \lambda_2 h)\left(\sqrt{2t\sigma'^2 + r^2 + 2x_1 r}\right) > g(\sigma'\sqrt{2t}), && \text{(B.165)}
\end{aligned}
$$

where $r = \|\boldsymbol{\delta}\|_2$ and we deem $\boldsymbol{\delta} = (r,\, 0,\, \ldots,\, 0)^\mathsf{T}$ by rotating the axis.

Lemma B.10. The function $(\lambda_1 g + \lambda_2 h)$ is unimodal in its domain $(0,\, +\infty)$.

Proof of Lemma B.10. We expand $(\lambda_1 g + \lambda_2 h)$ then consider its derivative.

$$\begin{aligned}&\lambda_1 g(r) + \lambda_2 h(r)\\ =&C_1\lambda_1 r^{-2k}\exp\left(-\frac{r^2}{2\sigma'^2}\right) + C_2\lambda_2 r^{-2k}\exp\left(-\frac{r^2}{2\beta'^2}\right),\end{aligned} \tag{B.166}$$

where C_1 and C_2 is the constant normalization coefficient of g and h respectively.

$$\begin{aligned}&(\lambda_1 g + \lambda_2 h)'(r)\\ =&-\left(C_1\lambda_1 2k r^{-2k-1}\exp\left(-\frac{r^2}{2\sigma'^2}\right) + C_1\lambda_1\cdot\frac{r^{-2k+1}}{\sigma'^2}\exp\left(-\frac{r^2}{2\sigma'^2}\right)\right.\\ &\left.+C_2\lambda_2 2k r^{-2k-1}\exp\left(-\frac{r^2}{2\beta'^2}\right) + C_2\lambda_2\cdot\frac{r^{-2k+1}}{\beta'^2}\exp\left(-\frac{r^2}{2\beta'^2}\right)\right).\end{aligned} \tag{B.167}$$

Now we show that

$$(\lambda_1 g + \lambda_2 h)'(r) = 0 \tag{B.168}$$

has at most one solution.

When either $\lambda_1 = 0$ or $\lambda_2 = 0$, since both g and h are monotonic, $\lambda_1 g + \lambda_2$ is monotonic and there is no solution. Thus, we assume $\lambda_1,\ \lambda_2 \neq 0$. We observe that

$$(\lambda_1 g + \lambda_2 h)'(r) = 0 \tag{B.169}$$

$$\iff C_1\lambda_1\left(2k + \frac{r^2}{\sigma'^2}\right)\exp\left(-\frac{r^2}{2\sigma'^2}\right) + C_2\lambda_2\left(2k + \frac{r^2}{\beta'^2}\right)\exp\left(-\frac{r^2}{2\beta'^2}\right) = 0 \tag{B.170}$$

$$\iff -\frac{C_1\lambda_1}{C_2\lambda_2}\cdot\frac{2k + \frac{r^2}{\sigma'^2}}{2k + \frac{r^2}{\beta'^2}} = \exp\left(\frac{r^2}{2\sigma'^2} - \frac{r^2}{2\beta'^2}\right) \tag{B.171}$$

$$\overset{x:=r^2}{\iff} -\frac{C_1\lambda_1}{C_2\lambda_2}\cdot\frac{2k + \frac{x}{\sigma'^2}}{2k + \frac{x}{\beta'^2}} = \exp\left(\frac{x}{2\sigma'^2} - \frac{x}{2\beta'^2}\right) \tag{B.172}$$

$$\iff -\frac{C_2\lambda_2}{C_1\lambda_1}\cdot\frac{2k + \frac{x}{\beta'^2}}{2k + \frac{x}{\sigma'^2}} = \exp\left(\frac{x}{2\beta'^2} - \frac{x}{2\sigma'^2}\right). \tag{B.173}$$

We focus on Equation (B.173). Without loss of generality, we assume $\sigma' > \beta'$, then both function $x \mapsto \frac{2k + x/\beta'^2}{2k + x/\sigma'^2}$ and function $x \mapsto \exp\left(x/(2\beta'^2) - x/(2\sigma'^2)\right)$ are monotonically increasing. If $\lambda_1\lambda_2 > 0$, the LHS and RHS of Equation (B.173) are continuous and monotonic in opposite directions. Thus, there is at most one solution to Equation (B.173). If $\lambda_1\lambda_2 < 0$,

the LHS is monotonic increasing but the derivative is decreasing because

$$-\frac{C_2\lambda_2}{C_1\lambda_1}\cdot\frac{2k+\frac{x}{\beta'^2}}{2k+\frac{x}{\sigma'^2}}=-\frac{C_2\lambda_2}{C_1\lambda_1}\cdot\frac{1+\frac{x}{2k\beta'^2}}{1+\frac{x}{2k\sigma'^2}} \tag{B.174}$$

where the numerator $1+\frac{x}{2k\beta'^2}$ is linearly increasing and the denominator $1+\frac{x}{2k\sigma'^2}$ is also linearly increasing. On the other hand, the RHS is monotonic increasing and the derivative is also increasing because

$$\frac{1}{2\beta'^2}-\frac{1}{2\sigma'^2}>0. \tag{B.175}$$

As a result, the difference function between RHS and LHS is monotone and there is at most one solution. Thus, we have shown Equation (B.173) has at most one solution.

Given that $(\lambda_1 g+\lambda_2 h)'$ is also continuous, we thus know the function $(\lambda_1 g+\lambda_2 h)$ is unimodal. QED.

Moreover, since g and h approach 0 when $r\to\infty$, $(\lambda_1 g+\lambda_2 h)$ approaches 0 when $r\to\infty$. We define

$$\overline{(\lambda_1 g+\lambda_2 h)^{-1}}(y):=\max y' \quad \text{s.t.} \quad \lambda_1 g(y')+\lambda_2 h(y')=x, \tag{B.176}$$

$$\underline{(\lambda_1 g+\lambda_2 h)^{-1}}(y):=\min y' \quad \text{s.t.} \quad \lambda_1 g(y')+\lambda_2 h(y')=x. \tag{B.177}$$

Then, Lemma B.10 and $(\lambda_1 g+\lambda_2 h)\to 0$ when $r\to\infty$ imply that, when $y>0$,

$$\begin{aligned} & y_0\in\left(\underline{(\lambda_1 g+\lambda_2 h)^{-1}}(y),\ \overline{(\lambda_1 g+\lambda_2 h)^{-1}}(y)\right) \\ \iff & \lambda_1 g(y_0)+\lambda_2 h(y_0)>y. \end{aligned} \tag{B.178}$$

We simplify Equation (B.165) by observing that $g(\sigma'\sqrt{2t})>0$:

$$\text{Equation (B.165)}$$

$$\iff \sqrt{2t\sigma'^2+r^2+2x_1 r}\in\left(\underline{(\lambda_1 g+\lambda_2 h)^{-1}}(g(\sigma'\sqrt{2t})),\overline{(\lambda_1 g+\lambda_2 h)^{-1}}(g(\sigma'\sqrt{2t}))\right) \tag{B.179}$$

$$\iff \frac{\underline{(\lambda_1 g+\lambda_2 h)^{-1}}(g(\sigma'\sqrt{2t}))^2-2t\sigma'^2-r^2}{2r}\leqslant x_1\leqslant\frac{\overline{(\lambda_1 g+\lambda_2 h)^{-1}}(g(\sigma'\sqrt{2t}))^2-2t\sigma'^2-r^2}{2r}. \tag{B.180}$$

Combining Lemma B.9 with Equation (B.180) we get

$$\begin{aligned}
&\Pr\left[\lambda_1 p(\boldsymbol{x}+\boldsymbol{\delta})+\lambda_2 q(\boldsymbol{x}+\boldsymbol{\delta}) > g(\sigma'\sqrt{2t}) \,\middle|\, \|\boldsymbol{x}\|_2 = \sigma'\sqrt{2t}\right] \\
=&\text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{1}{2}+\frac{\overline{(\lambda_1 g+\lambda_2 h)^{-1}}(g(\sigma'\sqrt{2t}))^2-2t\sigma'^2-r^2}{4r\sigma'\sqrt{2t}}\right) \\
&-\text{BetaCDF}_{\frac{d-1}{2}}\left(\frac{1}{2}+\frac{\underline{(\lambda_1 g+\lambda_2 h)^{-1}}(g(\sigma'\sqrt{2t}))^2-2t\sigma'^2-r^2}{4r\sigma'\sqrt{2t}}\right).
\end{aligned} \tag{B.181}$$

Combining the above equation with (9.) yields the expression shown in Table B.2. QED.

Proof of Theorem B.5. To prove the theorem, the main work we need to do is deriving the closed-form expression for the inverse function g^{-1}, where

$$g(r) = \frac{1}{(2\sigma'^2)^{\frac{d}{2}-k}\pi^{\frac{d}{2}}} \cdot \frac{\Gamma(\frac{d}{2})}{\Gamma(\frac{d}{2}-k)} r^{-2k} \exp\left(-\frac{r^2}{2\sigma^2}\right). \tag{B.182}$$

(I.) When $k = 0$,

$$\begin{aligned}
&\text{Equation (B.182)} \\
\iff & y = \frac{1}{(2\sigma'^2\pi)^{\frac{d}{2}}} \exp\left(-\frac{g^{-1}(y)^2}{2\sigma'^2}\right) && \text{(B.183)} \\
\iff & g^{-1}(y)^2 = -2\sigma'^2 \ln\left((2\sigma'^2\pi)^{\frac{d}{2}} y\right). && \text{(B.184)}
\end{aligned}$$

(II.) When $k > 0$, we notice that the Lambert W function W is the inverse function of $w \mapsto we^w$, i.e., $W(x)\exp(W(x)) = x$. We let the normalizing coefficient of $g(r)$ be

$$C := \frac{1}{(2\sigma'^2)^{\frac{d}{2}-k}\pi^{\frac{d}{2}}} \cdot \frac{\Gamma(\frac{d}{2})}{\Gamma(\frac{d}{2}-k)}. \tag{B.185}$$

Then

$$\begin{aligned}
&\text{Equation (B.182)} \\
\iff & y = C g^{-1}(y)^{-2k} \exp\left(-\frac{g^{-1}(y)^2}{2\sigma'^2}\right) && \text{(B.186)} \\
\iff & \frac{C}{y} = g^{-1}(y)^2 k \exp\left(\frac{g^{-1}(y)^2}{2\sigma'^2}\right) && \text{(B.187)} \\
\iff & \left(\frac{C}{y}\right)^{\frac{1}{k}} = g^{-1}(y)^2 \exp\left(\frac{g^{-1}(y)^2}{k\sigma'^2}\right) && \text{(B.188)}
\end{aligned}$$

$$\iff \frac{1}{2k\sigma^2}\left(\frac{C}{y}\right)^{\frac{1}{k}} = W^{-1}\left(\frac{g^{-1}(y)^2}{2k\sigma'^2}\right) \tag{B.189}$$

$$\iff g^{-1}(y)^2 = 2\sigma'^2 kW\left(\frac{1}{2k\sigma'^2}\left(\frac{C}{y}\right)^{\frac{1}{k}}\right). \tag{B.190}$$

Plugging in Equations (B.184) and (B.190) to $g^{-1}(\lambda_1 g(\sigma'\sqrt{2x}) + \lambda_2 h(\sigma'\sqrt{2x}))^2$ and $g^{-1}(\lambda_1 g(\beta'\sqrt{2x}) + \lambda_2 h(\beta'\sqrt{2x}))^2$ for $k = 0$ and $k > 0$ case, then results in Table B.3 follow from algebra. QED.

Proof of Theorem B.6. We prove the theorem by contradiction. Suppose that the (λ_1, λ_2) that satisfy the constraint of Equation (2.19) are not unique, and we let $(\lambda_1^a, \lambda_2^a)$ and $(\lambda_1^b, \lambda_2^b)$ be such two pairs. Without loss of generality, we assume $\lambda_1^a \neq \lambda_1^b$.

If $\lambda_2^a = \lambda_2^b$, we have $P(\lambda_1^a, \lambda_2^a) = P(\lambda_1^b, \lambda_2^b)$, i.e., the region

$$\Big\{\boldsymbol{x} - \boldsymbol{\delta} : p(\boldsymbol{x} - \boldsymbol{\delta}) \in \Big[\min\{\lambda_1^a, \lambda_1^b\}p(\boldsymbol{x}) + \lambda_2^a q(\boldsymbol{x}), \max\{\lambda_1^a, \lambda_1^b\}p(\boldsymbol{x}) + \lambda_2^a q(\boldsymbol{x})\Big)\Big\} \tag{B.191}$$

has zero mass under distribution $\mathcal{P}$. Given that $\mathcal{P}$ and $\mathcal{Q}$ have positive and continuous density functions almost everywhere, the volume of the region is non-zero thus the mass under distribution $\mathcal{P}$ is also non-zero. Therefore, we also have $\lambda_2^a \neq \lambda_2^b$. Because of the partial monotonicity of P and Q functions (shown in Section 2.5.3), without loss of generality, we assume

$$\lambda_1^a < \lambda_1^b, \quad \lambda_2^a > \lambda_2^b. \tag{B.192}$$

Lemma B.11. There exists a point $r_0 \geqslant 0$, either (1) or (2) is satisfied.

(1) When $r > r_0$,

$$\Pr[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1^a p(\boldsymbol{\epsilon}) + \lambda_2^a q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r] \geqslant \Pr[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1^b p(\boldsymbol{\epsilon}) + \lambda_2^b q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r]; \tag{B.193}$$

(B.194)

when $r < r_0$,

$$\Pr[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1^a p(\boldsymbol{\epsilon}) + \lambda_2^a q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r] \leqslant \Pr[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1^b p(\boldsymbol{\epsilon}) + \lambda_2^b q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r]. \tag{B.195}$$

(2) When $r > r_0$,

$$\Pr[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1^a p(\boldsymbol{\epsilon}) + \lambda_2^a q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r] \leqslant \Pr[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1^b p(\boldsymbol{\epsilon}) + \lambda_2^b q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r]; \tag{B.196}$$

$$\tag{B.197}$$

when $r < r_0$,

$$\Pr[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1^a p(\boldsymbol{\epsilon}) + \lambda_2^a q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r] \geqslant \Pr[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1^b p(\boldsymbol{\epsilon}) + \lambda_2^b q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r]. \tag{B.198}$$

Note that in $\Pr[\cdot \,|\, \|\boldsymbol{\epsilon}\|_p = r]$, the vector $\varepsilon \in \mathbb{R}^d$ is uniformly sampled from the ℓ_p-sphere of radius r.

Proof of Lemma B.11. For a given r, since $\mathcal{P}$ and $\mathcal{Q}$ are both ℓ_p-spherically symmetric, and the density functions are both positive almost everywhere, as long as

$$\lambda_1^a g(r) + \lambda_2^a h(r) \leqslant \lambda_1^b g(r) + \lambda_2^b h(r), \tag{B.199}$$

then

$$\begin{aligned} &[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1^a p(\boldsymbol{\epsilon}) + \lambda_2^a q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r] \\ \leqslant &[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \lambda_1^b p(\boldsymbol{\epsilon}) + \lambda_2^b q(\boldsymbol{\epsilon}) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r]. \end{aligned} \tag{B.200}$$

It still holds if we change the "$\leqslant$"'s to "$\geqslant$"'s in both Equations (B.199) and (B.200). Meanwhile,

$$\lambda_1^a g(r) + \lambda_2^a h(r) \leqslant \lambda_1^b g(r) + \lambda_2^b h(r) \iff \frac{g(r)}{h(r)} > \frac{\lambda_2^b - \lambda_2^a}{\lambda_1^a - \lambda_1^b}. \tag{B.201}$$

Since $g(r)/h(r)$ is strictly monotonic, there exists *at most* one point $r_0 \geqslant 0$ that divides

$$\frac{g(r)}{h(r)} > \frac{\lambda_2^b - \lambda_2^a}{\lambda_1^a - \lambda_1^b} \quad \text{and} \quad \frac{g(r)}{h(r)} < \frac{\lambda_2^b - \lambda_2^a}{\lambda_1^a - \lambda_1^b}. \tag{B.202}$$

If that r_0 exists, from Equations (B.199) to (B.201) the lemma statement follows.

Now we only need to show that r_0 exists. Assume that the point r_0 does not exist, it implies that for all r, we have either

$$\lambda_1^a g(r) + \lambda_2^a h(r) < \lambda_1^b g(r) + \lambda_2^b h(r) \tag{B.203}$$

or

$$\lambda_1^a g(r) + \lambda_2^a h(r) > \lambda_1^b g(r) + \lambda_2^b h(r) \tag{B.204}$$

while $P(\lambda_1^a,\, \lambda_2^a) = P(\lambda_1^b,\, \lambda_2^b) > 0$ and $Q(\lambda_1^a,\, \lambda_2^a) = Q(\lambda_1^b,\, \lambda_2^b) > 0$. It implies that for almost every r,

$$\Pr[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) \in (a,\, b) \,\big|\, \|\boldsymbol{\epsilon}\|_p = r] = 0 \tag{B.205}$$

where

$$\begin{aligned} a &= \min\{\lambda_1^a g(r) + \lambda_2^a h(r),\, \lambda_1^b g(r) + \lambda_2^b h(r)\}, \\ b &= \max\{\lambda_1^a g(r) + \lambda_2^a h(r),\, \lambda_1^b g(r) + \lambda_2^b h(r)\}. \end{aligned} \tag{B.206}$$

This violates the continuous assumption on both $\mathcal{P}$ and $\mathcal{Q}$. Therefore, point r_0 exists. QED.

With Lemma B.11, we define auxiliary function $D : \mathbb{R}_+ \to \mathbb{R}$ such that

$$\begin{aligned} D(r) = &\Pr[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1^a p(\boldsymbol{\epsilon}) + \lambda_2^a q(\boldsymbol{\epsilon}) \,|\, \|\boldsymbol{\epsilon}\|_p = r] \\ &- \Pr[p(\boldsymbol{\epsilon} - \boldsymbol{\delta}) < \lambda_1^b p(\boldsymbol{\epsilon}) + \lambda_2^b q(\boldsymbol{\epsilon}) \,|\, \|\boldsymbol{\epsilon}\|_p = r]. \end{aligned} \tag{B.207}$$

We let $S(r)$ be the surface area of ℓ_p sphere of radius r. Then the P and Q can be written in integral form:

$$P(\lambda_1,\, \lambda_2) = \int_0^\infty \Pr[p(\varepsilon - \boldsymbol{\delta}) < \lambda_1 p(\varepsilon) + \lambda_2 p(\varepsilon) \,|\, \|\varepsilon\|_p = r] \cdot g(r) S(r)\, \mathrm{d}r, \tag{B.208}$$

$$Q(\lambda_1,\, \lambda_2) = \int_0^\infty \Pr[p(\varepsilon - \boldsymbol{\delta}) < \lambda_1 p(\varepsilon) + \lambda_2 p(\varepsilon) \,|\, \|\varepsilon\|_p = r] \cdot h(r) S(r)\, \mathrm{d}r. \tag{B.209}$$

Since $P(\lambda_1^a,\, \lambda_2^a) = P(\lambda_1^b,\, \lambda_2^b)$ and $Q(\lambda_1^a,\, \lambda_2^a) = Q(\lambda_1^b,\, \lambda_2^b)$ by our assumption, simple arrangement yields

$$\int_0^{r_0} D(r) g(r) S(r)\, \mathrm{d}r = \int_{r_0}^\infty (-D(r)) g(r) S(r)\, \mathrm{d}r \neq 0, \tag{B.210}$$

$$\int_0^{r_0} D(r) h(r) S(r)\, \mathrm{d}r = \int_{r_0}^\infty (-D(r)) h(r) S(r)\, \mathrm{d}r \neq 0. \tag{B.211}$$

As Lemma B.11 shows, $D(r)$ where $r \in [0,\, r_0]$ always has the same sign as $-D(r)$ where $r \in [r_0,\, +\infty]$, and the last inequality ($\neq 0$) is again due to the continuity of p and q and the fact that $P(\lambda_1^a,\, \lambda_2^a) > 0$ and $Q(\lambda_1^a,\, \lambda_2^a) > 0$. Now we can divide Equation (B.210) by Equation (B.211) and apply the Cauchy's mean value theorem, which yields

$$\frac{D(\xi_1) g(\xi_1) S(\xi_1)}{D(\xi_1) h(\xi_1) S(\xi_1)} = \frac{D(\xi_2) g(\xi_2) S(\xi_2)}{D(\xi_2) h(\xi_2) S(\xi_2)}, \tag{B.212}$$

where $\xi_1 \in (0,\, r_0)$ and $\xi_2 \in (r_0,\, +\infty)$. Apparently, it requires

$$\frac{g(\xi_1)}{h(\xi_1)} = \frac{g(\xi_2)}{h(\xi_2)}. \tag{B.213}$$

However, g/h is strictly monotonic. By contradiction, there is no distinct pair $(\lambda_1^a,\, \lambda_2^a)$ and $(\lambda_1^b,\, \lambda_2^b)$ satisfying the constraint of Equation (2.19) simultaneously. QED.

Remark B.3. Suppose $\mathcal{P}$ and $\mathcal{Q}$ are ℓ_p-radial extended Gaussian/Laplace distributions, i.e., their density functions $p(\boldsymbol{x})$ and $q(\boldsymbol{x})$ are

$$\begin{aligned} p(\boldsymbol{x}) &\propto \|\boldsymbol{x}\|_p^{-k} \exp\left(-\|\boldsymbol{x}\|_p/\sigma\right)^\alpha, \\ q(\boldsymbol{x}) &\propto \|\boldsymbol{x}\|_p^{-k} \exp\left(-\|\boldsymbol{x}\|_p/\beta\right)^\alpha \end{aligned} \tag{B.214}$$

with $\alpha > 0$ (for Gaussian $\alpha = 2$ and for Laplace $\alpha = 1$). Note that this is a broader family than the family considered in DSRS shown in the main text. we have

$$\frac{g}{h} \propto \exp(r/\beta - r/\sigma)^\alpha \tag{B.215}$$

that is strictly monotonic. Thus, Theorem B.6 is applicable for this large family of smoothing distributions that are commonly used in the literature [77, 165, 204, 427, 437, 442].

B.3.3 Discussion on Certification of Other ℓ_p Norms

DSRS can also be extended to provide a certified robust radius under ℓ_p norm other than ℓ_2.

Different from the case of ℓ_2 certification, for other ℓ_p norm the challenge is to compute $P(\lambda_1, \lambda_2)$, $Q(\lambda_1, \lambda_2)$, and $R(\lambda_1, \lambda_2)$ as defined in Theorem 2.5. For ℓ_2 certification, as shown by Theorems 2.5 and B.4, there exist closed-form expressions for these quantities that can be efficiently implemented with numerical integrations. However, for other ℓ_p norms, finding such closed-form expressions for P, Q, and R is challenging.

Luckily, we notice that P, Q, and R are all probability-based definitions, as long as we can effectively sample from $\mathcal{P}$ and $\mathcal{Q}$ and effectively compute the density functions $p(\cdot)$ and $q(\cdot)$, we can estimate these function quantities from the empirical means of Monte-Carlo sampling.

Compared to numerical integration based ℓ_2 certification, Monte-Carlo sampling has *sampling uncertainty* and *efficiency* problems. Here we discuss these two problems in detail and how we can alleviate them.

Sampling Uncertainty and Mitigations. The empirical means for P and Q are stochastic, which breaks the nice properties of P and Q (shown in Section 2.5.3) with respect to (λ_1, λ_2) as the different queries to P and Q fluctuate around the actual value. Therefore, the joint binary search (Algorithm B.1) may fail to return the correct (λ_1, λ_2) pair. Thus, we propose a stabilization trick: the *same* set of samples is used when querying P and Q during the joint binary search. With this same set of samples, it can be easily verified that the nice properties (Propositions 2.2 and 2.3) still hold even if P and Q are empirical means. To guarantee the certification soundness in the context of probabilistic Monte-Carlo sampling, we introduce a different set of samples to test whether the solved (λ_1, λ_2) indeed upper bounds the intersection point (if the test fails we fall back to the classical Neyman-Pearson-based certification though it seldom happens). Since the test is also probabilistic, we need to accumulate this additional failing probability and use the lower bound of the confidence interval for soundness, which results in $1 - 2\alpha = 99.8\%$ certification instead of $1 - \alpha = 99.9\%$ as in classical randomized smoothing certification [77, 427]. Note that the existence of additional confidence intervals caused by Monte-Carlo sampling makes DSRS based on Monte-Carlo sampling slightly looser than DSRS based on numerical integration.

Efficiency Concern and Mitigations. Traditionally we need to sample several (denoted as M, in our implementation we set $M = 5 \times 10^4$) high-dimensional vectors for *each* P or Q computation, which induces the efficiency concern. With the usage of the aforementioned stabilization trick, now we only sample one set of M samples during the whole joint binary search instead of during each P and Q computation. Combining with the testing phase sampling, the whole algorithm needs to sample $2M$ vectors rather than $\mathcal{O}(M \log^2 M)$ without the stabilization trick, so it greatly solves the efficiency concern. Moreover, we notice that for the samples $\{\epsilon_i\}_{i=1}^M$ we only need to care about its densities $\{p(\varepsilon_i)\}_{i=1}^M, \{q(\varepsilon_i)\}_{i=1}^M$ and $\{p(\varepsilon_i - \boldsymbol{\delta})\}_{i=1}^M$. Thus, we store only these three quantities instead of the whole d-dimensional vectors $\{\epsilon_i\}_{i=1}^M$, reducing the space complexity from $\mathcal{O}(M \times d)$ to $\mathcal{O}(M)$.

These techniques significantly improve efficiency in practice. Although DSRS based on Monte-Carlo sampling is still slightly looser and slower than DSRS based on numerical integration, DSRS based on Monte-Carlo sampling makes certifying robustness under other ℓ_p norms feasible. In this work, we focus on the ℓ_2 norm, because additive randomized smoothing is not optimal for other norms (e.g., ℓ_1 [214]) or the state-of-the-art certification can be directly translated from ℓ_2 certification (e.g., ℓ_∞ [427] and semantic transformations [223]). Moreover, to the best of our knowledge, standard ℓ_2 certification is the most challenging setting where additive randomized smoothing achieves state-of-the-art and no other work can achieve visibly tighter robustness certification than standard Neyman-Pearson certifica-

tion [215, 267, 427].

B.4 IMPLEMENTATION AND OPTIMIZATIONS

In this appendix, we discuss the implementation tricks and optimizations, along with our simple heuristic for selecting the hyperparameter T in the additional smoothing distribution $\mathcal{Q} = \mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$.

B.4.1 Implementation Details

We implement DSRS in Python with about two thousand lines of code. The tool uses `PyTorch`[16] to query a given base classifier with Monte-Carlo sampling in order to derive the confidence intervals $[\underline{P_A}, \overline{P_A}]$ and $[\underline{Q_A}, \overline{Q_A}]$. Then, the tool builds the whole DSRS pipeline on `SciPy`[17] and `NumPy`[18]. Specifically, the numerical integration is implemented with `scipy.integrate.quad()` method. We exploit the full independence across the certification for different input instances and build the tool to be fully parallelizable on CPUs. By default, we utilize 10 processes, and the number of processes can be dynamically adjusted.

The tool is built in a flexible way that adding new smoothing distributions is not only theoretically straightforward but also easy in practice. In the future, we plan to extend the tool to 1) provide GPU support; 2) reuse existing certification results from previous instances with similar confidence intervals to achieve higher efficiency. We will also support more smoothing distributions.

In the implementation, we widely use the logarithmic scale, since many quantities in the computation have varied scales. For example, since the input dimension d is typically over 500 on a real-world dataset, the density functions p and q decay very quickly along with the increase of noise magnitude. Another example is the input variable for $\ln(\cdot)$ and $W(\cdot)$ in Theorem 2.5. These variables are exponential with respect to the input dimension d so they could be very large or small. To mitigate this, we perform all computations with varied scales in logarithmic scale to improve the precision and floating-point soundness. For example, we implement a method `lnlogadd` to compute $\log(\lambda_1 \exp(x_1) + \lambda_2 \exp(x_2))$ and apply method `wlog` in [427] to compute $W(\exp(x))$. We remark that in the binary search for dual variables (see Section 2.5.3), we also use the logarithmic scale for λ_1 and λ_2.

[16] `http://pytorch.org/`
[17] `https://scipy.org/scipylib/index.html`
[18] `https://numpy.org/`

The code, model weights, and all experiment data are publicly available at `https://github.com/llylly/DSRS`.

B.4.2 T Heuristics

As briefly outlined in Section 2.6.1, we apply a simple yet effective heuristic to determine the hyperparameter T in additional smoothing distribution $\mathcal{Q} = \mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$.

Concretely, we first sample the prediction probability from the original smoothing distribution $\mathcal{P}$ and get the confidence lower bound $\underline{P_A}$ of $P_A = f^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}$. Then, we use the following empirical expression to determine T from $\underline{P_A}$:

$$T = \sigma \sqrt{\frac{2d}{d-2k} \Gamma\mathrm{CDF}^{-1}_{d/2-k}(p)}, \tag{B.216}$$

where

$$p = \max\{-0.08 \ln(1 - \underline{P_A}) + 0.2, 0.5\}. \tag{B.217}$$

It can be viewed as we first parameterize T by p and then find a simple heuristic to determine p by $\underline{P_A}$.

The T's parameterization with p is inspired by the probability mass under original smoothing distribution $\mathcal{P} = \mathcal{N}^{\mathsf{g}}(k, \sigma)$ if true-decision region is concentrated in a ℓ_2-ball centered at $\boldsymbol{x}_0$. Concretely, from $\mathcal{P}$'s definition,

$$\Pr_{\boldsymbol{\epsilon} \sim \mathcal{P}}[\|\boldsymbol{\epsilon}\|_2 \leqslant T] = p. \tag{B.218}$$

Then, we use a randomly sampled CIFAR-10 training set containing $1,000$ points with models trained using Consistency [165] under $\sigma = 0.50$ to sweep all $p \in \{0.1, 0.2, \cdots, 0.9\}$. We plot the minimum p and maximum p that gives the highest certified robust radius as a segment and find Equation (B.217) fits the general tendency well as shown in Appendix B.4.2. Thus, we use this simple heuristic to determine T. Empirically, this simple heuristic generalizes well and is competitive with more complex methods as shown in Appendix B.5.

Another heuristic that we have deployed is the fall-back strategy. When the empirical probability $\widetilde{P_A} = 1$, i.e., if for all the sampled $\boldsymbol{\epsilon} \sim \mathcal{P}$, $F(\boldsymbol{x}_0 + \boldsymbol{\epsilon}) = y_0$, then we fall back to still using $\mathcal{P}$ instead of using another distribution $\mathcal{Q}$ for the second round of sampling. This strategy is inspired by a finding that, with the fixed sampling budget, if $\underline{P_A}$ is already very high, it is more efficient to use more samples to further increase the confidence interval of $\underline{P_A}$ rather than querying imprecise information under another distribution $\mathcal{Q}$. Notice that such strategy does not break the high-confidence soundness of our certification, because

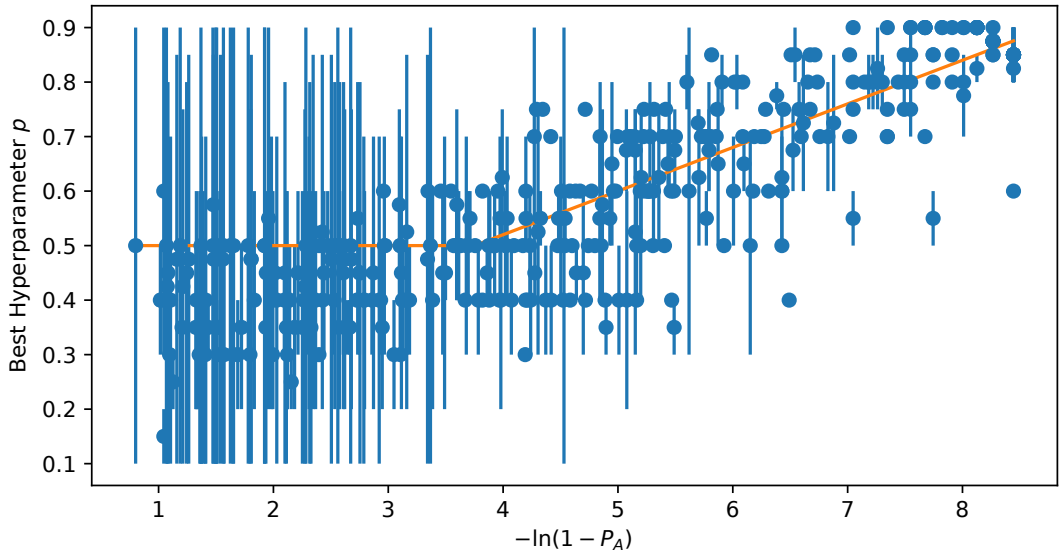


Figure B.1: Our p heuristic (Equation (B.217)) shown as orange curve fits the generalization tendency of optimal p ranges (shown as blue segments) well.

$\Pr[\widetilde{P_A} = n/N \,|\, P_A \leqslant t] \geqslant \Pr[\widetilde{P_A} = n/N \wedge \widetilde{P_A^{\text{half}}} = 1 \,|\, P_A \leqslant t]$ for any $t < 1$ and Bernoulli distributed sampling (which is our case), and we use Bernoulli confidence interval that corresponds to $\Pr[\widetilde{P_A} = n/N \,|\, P_A \leqslant t]$. Due to the tight experiment time, we only deployed this strategy on ImageNet evaluation but not on MNIST and CIFAR-10 evaluations.

B.4.3 Training Strategy for Generalized Gaussian Smoothing

We train the base classifiers on each dataset using Gaussian augmentation training [77], Consistency training [165], and SmoothMix training [166], which are typical training methods for randomized smoothing. We do not consider other training approaches such as [56, 218, 317, 437] because: (1) Some training approaches either require additional data [56, 317], or relatively time-consuming [317], or are reported to be not as effective as later approaches [218]; (2) Selected training approaches are widely used or achieve state-of-the-art with high training efficiency.

On MNIST, for all training methods, we use a convolutional neural network with 4 convolutional layers and 3 fully connected layers following Cohen et al. [77] as the base classifiers' architecture. On CIFAR-10, for all training methods, we use ResNet-110 [142] as the base classifiers' architecture. These architecture settings follow the prior work on smoothed classifier training [77, 317, 437].

On both MNIST and CIFAR-10, for all training methods, we train for 150 epochs. The learning rate is 0.01 and is decayed by 0.1 at the 50th and 100th epoch. For Consistency training, the hyperparameter $\lambda = 5$ on MNIST and $\lambda = 20$ on CIFAR-10. We use two noise vectors per training batch per instance. These are the best hyperparameters reported in [165]. The batch size is 256 on both MNIST and CIFAR-10 following [165]. For SmoothMix training, we directly use the best hyperparameters from their open-source repository: `https://github.com/jh-jeong/smoothmix/blob/main/EXPERIMENTS.MD`.

On ImageNet, we use ResNet-50 [142] as the base classifiers' architecture and finetune from the open-source model trained by Cohen et al. [77] with Gaussian smoothing. We train for 10 epochs due to the expensive training cost on ImageNet and we remark that better results can be achieved with a larger training time budget. The learning rate is 0.001 and is decayed at the end of every epoch by 0.1. For Consistency training, the hyperparameter $\lambda = 5$ and we use two noise vectors per training batch per instance following the best hyperparameters reported in [165]. For SmoothMix training, since the open-source repository does not contain the best hyperparameters, we select the hyperparameters as suggested in the original paper [166].

During training, the training samples are augmented by adding noises following *training smoothing distribution*. In typical training approaches [56, 77, 165, 317, 437], the training smoothing distribution is set to be the same as the original smoothing distribution $\mathcal{P}$ for constructing the smoothed classifier. However, for our generalized Gaussian $\mathcal{N}^{\mathrm{g}}(k, \sigma)$ as $\mathcal{P}$ with large k, we find this strategy gives a poor empirical performance.

To better train the base classifier for our original smoothing distribution $\mathcal{P}$ with large k, we introduce a warm-up stage on the training smoothing distribution. Suppose our original smoothing distribution $\mathcal{P}$ for constructing the smoothed classifier is $\mathcal{N}^{\mathrm{g}}(k_0, \sigma)$ We let $e_0 = 100$ be the number of warm-up epochs on MNIST and CIFAR-10, or $e_0 = 10000$ be the number of warm-up steps on ImageNet. In the first e_0 epochs (on MNIST and CIFAR-10) or steps (on ImageNet), we use the training smoothing distribution with smaller k. Formally, in the eth epoch/step where $e \leqslant e_0$, the training smoothing distribution $\mathcal{P}' = \mathcal{N}^{\mathrm{g}}(k, \sigma)$ where

$$k = \lceil k_0 - k_0^{1-e/e_0} \rceil. \tag{B.219}$$

For later epochs/steps, we use the original smoothing distribution $\mathcal{P}$ itself as the training smoothing distribution. This strategy gradually increases the k of training smoothing distribution throughout the training, so that the base classifier can be a better fit for the final desired distribution $\mathcal{P}$. Using this strategy, the smoothed classifier constructed from our trained base classifier has similar certified robustness compared to standard Gaussian

augmentation under classical Neyman-Pearson-based certification.

B.5 ADDITIONAL EXPERIMENTAL RESULTS

In this appendix, we present additional experiment results and studies.

B.5.1 Empirical Study Setup of Concentration Property

In Figure 2.3, we present our investigation of the decision regions of base classifiers. The investigation follows the following protocol: (1) We choose the base classifier from [317] on ImageNet trained for $\sigma = 0.5$ Gaussian smoothing as the subject. The reason is that this base classifier is one of the state-of-the-art certifiably robust classifiers on the large-scale ImageNet dataset and our code uses the same preprocessing parameters so it is easy to adopt their model. (2) We pick every 500-th image from the test set of the ImageNet dataset to form a subset of 100 samples. (3) We filter out the samples where the base classifier cannot classify correctly even without adding any noise, which results in 89 remaining samples. (4) For each of these 89 samples, for each perturbation magnitude r, we uniformly sample 1000 perturbation vectors from the hypersphere with ℓ_2-radius r and compute the empirical probability of true-prediction, where the step size of r is 10.

Figure 2.3 implies that for a vast majority of samples, the true-prediction samples are highly concentrated on an ℓ_2 ball around the clean input since there exists apparent ℓ_2 magnitude thresholds where the true-prediction probability is almost 1 within the thresholds and almost 0 beyond the thresholds. This implies that the concentration property may be achievable for real-world base classifiers in randomized smoothing.

In Figure B.3, we follow the same protocol but plot the landscape of base classifiers trained using generalized Gaussian distribution (instead of standard Gaussian distribution as in Figure 2.3). By comparing Figure B.3 and Figure 2.3, we find that although base classifiers in Figure 2.3 can achieve better certified robustness using Neyman-Pearson certification and generalized Gaussian smoothing (compare Figure 2.5(b) and Neyman-Pearson rows in Table B.9), they also sacrifice the concentration property, which can explain why DSRS improvements are much smaller on models in Section 2.6 compared with models in Figure 2.5(b). Thus, as discussed in Section 2.6, there may be a large space for exploring training approaches that favor DSRS certification by preserving the concentration property.

B.5.2 Effectiveness of T-Heuristics and Attempts on Better Optimization Tricks

Better Optimization Tricks. For obtaining a tighter certified radius, we should make the support of $\mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$ more aligned with the decision region while keeping the Q_A large enough. So except using a simple heuristic, we can also turn the search for an appropriate value of T into an optimization problem. In order to make the optimization more stable, here we will construct the optimization based on P_{con} that is more scale-invariant, and then transform it to get the final appropriate $T = \sigma\sqrt{2\Gamma\mathrm{CDF}^{-1}_{d/2}(P_{\mathsf{con}})}$.

The final optimization objective is now built as $P_{\mathsf{con}} = \arg\min -Q_A + \frac{\lambda}{2}(P_{\mathsf{con}} - P_A)^2$, where λ is a hyperparameter that controls the relative weight of the two loss terms. The Q_A here is estimated by sampling from the distribution $\mathcal{N}^{\mathsf{g}}_{\mathsf{trunc}}(k, T, \sigma)$ in which the T is determined by P_{con}; however, the actual process of such sampling is implemented by rejecting the sampled noise whose norm is bigger than T. Therefore, there will be no gradient obtained for P_{con} through the backward of the loss, namely, the optimized objective. So instead, we attempt to approximate the gradient comes from the first loss term $-Q_A$ with $G_{Q_A} = \mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[\nabla_{||\boldsymbol{\epsilon}||_2} \mathit{CrossEntropyLoss}(f(\boldsymbol{x}_0 + \boldsymbol{\epsilon}), y_0)]$. Then, we will only have the gradient information, and there is no explicit form of Q_A anymore. The P_A is estimated with the $\underline{P_A}$ which we have already obtained, so the final estimation of the gradient for P_{con} is $G_{Q_A} + \lambda(P_{\mathsf{con}} - \underline{P_A})$.

Experiment Setting. The P_{con} is optimized for different input test images respectively, and the initialized P_{con} is set to 0.7. For each input, we will update P_{con} 20 steps on datasets MNIST and CIFAR10 while updating only 10 steps on dataset ImageNet to reduce time complexity. For each step, we will sample $2,000$ times for estimating the term G_{Q_A}, and the learning rate is set to $2,000$. Besides, to avoid the P_{con} being optimized too large or too small, we will clip the final optimized P_{con} within 0.1 and 0.9. Since the optimization is a bit time-consuming, we only test it on CIFAR10 with $\sigma = 0.5$ and test it on MNIST and ImageNet with $\sigma = 1.0$. Different λ is also tried for different datasets and different training methods for getting better performance.

Performance. The final results are shown in Table B.4, and the certification approach based on the optimization tricks mentioned above is denoted as "Opt" in the table. As we can see, our simple T-Heuristics could still be competitive with the method based on complicated optimization but with a cheaper cost.

Table B.4: ℓ_2 certified robust accuracy w.r.t. different radii r for different certification approaches.

Dataset	Training Method	Certification Approach	Certified Accuracy under Radius r											
			0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6	1.8	2.0	2.2	2.4
MNIST	Gaussian Aug. ($\sigma = 1.00$)	Neyman-Pearson	95.5 %	93.5 %	90.0 %	86.1 %	80.4 %	72.8 %	61.4 %	50.2 %	36.6 %	25.2 %	14.5 %	8.5 %
		DSRS(T-heuristic)	95.5 %	93.5 %	90.2 %	86.9 %	81.4 %	74.4 %	64.6 %	55.2 %	42.8 %	30.9 %	20.3 %	11.3 %
		Opt ($\lambda = 7e-05$)	95.5%	93.5%	90.0%	86.9%	81.7%	74.9%	65.6%	55.8%	43.8%	30.5%	19.1%	10.2%
	Consistency ($\sigma = 1.00$)	Neyman-Pearson	94.5 %	92.6 %	89.3 %	85.9 %	80.7 %	74.4 %	65.9 %	56.9 %	44.1 %	34.4 %	23.3 %	12.8 %
		DSRS(T-heuristic)	94.5 %	92.8 %	89.3 %	86.3 %	81.4 %	76.1 %	68.3 %	59.5 %	50.7 %	39.8 %	30.7 %	20.0 %
		Opt ($\lambda = 6e-06$)	94.5%	92.8%	89.3%	86.2%	81.2%	75.8%	68.2%	59.6%	50.5%	39.6%	30.7%	19.8%
			Certified Accuracy under Radius r											
			0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2
CIFAR-10	Gaussian Aug. ($\sigma = 0.50$)	Neyman-Pearson	60.4 %	55.2 %	51.3 %	45.9 %	40.8 %	35.6 %	30.1 %	24.3 %	20.0 %	16.7 %	13.0 %	10.1 %
		DSRS(T-heuristic)	60.6 %	55.5 %	51.5 %	46.8 %	42.1 %	37.3 %	32.5 %	27.4 %	22.8 %	19.3 %	16.0 %	12.7 %
		Opt ($\lambda = 3e-06$)	60.6%	55.5%	51.3%	46.7%	42.0%	37.2%	32.5%	27.4%	23.0%	19.3%	16.0%	12.5%
	Consistency ($\sigma = 0.50$)	Neyman-Pearson	53.1 %	50.5 %	48.6 %	45.5 %	43.6 %	41.5 %	38.7 %	36.7 %	35.1 %	32.0 %	29.1 %	25.7 %
		DSRS(T-heuristic)	53.1 %	50.7 %	48.7 %	45.7 %	44.0 %	41.8 %	39.6 %	37.8 %	36.0 %	34.4 %	31.3 %	28.6 %
		Opt ($\lambda = 4e-06$)	53.1%	50.7%	48.7%	45.7%	44.0%	41.8%	39.5%	37.8%	36.0%	34.4%	31.4%	28.4%
ImageNet	Gaussian Aug. ($\sigma = 1.00$)	Neyman-Pearson	57.5 %	55.1 %	52.2 %	49.7 %	47.0 %	43.9 %	40.8 %	38.1 %	35.0 %	33.2 %	29.6 %	25.3 %
		DSRS(T-heuristic)	57.7 %	55.6 %	52.7 %	51.0 %	48.4 %	45.5 %	43.1 %	40.2 %	37.9 %	35.3 %	32.8 %	30.5 %
		Opt ($\lambda = 1e-05$)	57.7%	55.5%	52.4%	50.5%	48.2%	45.0%	42.9%	40.0%	38.0%	35.0%	32.7%	29.9%
	Consistency ($\sigma = 1.00$)	Neyman-Pearson	55.9 %	54.4 %	53.0 %	51.2 %	48.2 %	46.2 %	44.2 %	41.7 %	39.1 %	36.4 %	34.4 %	32.1 %
		DSRS(T-heuristic)	56.0 %	54.6 %	53.1 %	51.8 %	49.9 %	47.4 %	45.7 %	44.2 %	41.7 %	39.3 %	37.8 %	35.8 %
		Opt ($\lambda = 1e-05$)	56.0%	54.6%	53.1%	51.8%	49.7%	47.4%	45.3%	44.0%	41.6%	39.3%	37.8%	35.9%

B.5.3 Full Curves and Separated Tables

Separate Tables by Smoothing Variance

Table B.5: MNIST: Certified robust accuracy for models smoothed with different variance σ certified with different certification approaches.

Variance	Training Method	Certification Approach	Certified Accuracy under Radius r											
			0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00
0.25	Gaussian Aug. [77]	Neyman-Pearson	97.9%	96.4%	92.1%									
		DSRS	97.9%	96.6%	92.7%									
	Consistency [165]	Neyman-Pearson	98.4%	97.5%	94.4%									
		DSRS	98.4%	97.5%	95.4%									
	SmoothMix [166]	Neyman-Pearson	98.6%	97.6%	96.5%									
		DSRS	98.6%	97.7%	96.8%									
0.50	Gaussian Aug. [77]	Neyman-Pearson	97.8%	96.9%	94.6%	88.4%	78.7%	52.6%						
		DSRS	97.8%	97.0%	95.0%	89.8%	83.4%	59.1%						
	Consistency [165]	Neyman-Pearson	98.4%	97.3%	96.0%	92.3%	83.8%	67.5%						
		DSRS	98.4%	97.3%	96.0%	93.5%	87.1%	71.8%						
	SmoothMix [166]	Neyman-Pearson	98.2%	97.1%	95.4%	91.9%	85.1%	73.0%						
		DSRS	98.1%	97.1%	95.9%	93.4%	87.5%	76.6%						
1.00	Gaussian Aug. [77]	Neyman-Pearson	95.2%	91.9%	87.7%	80.6%	71.2%	57.6%	41.0%	25.5%	13.6%	6.2%	2.1%	0.9%
		DSRS	95.1%	91.8%	88.2%	81.5%	73.6%	61.6%	48.4%	34.1%	21.0%	10.6%	4.4%	1.2%
	Consistency [165]	Neyman-Pearson	93.9%	90.9%	86.4%	80.8%	73.0%	61.1%	49.1%	35.6%	21.7%	10.4%	4.1%	1.9%
		DSRS	93.9%	91.1%	86.9%	81.7%	75.2%	65.6%	55.8%	41.9%	31.4%	17.8%	8.6%	2.8%
	SmoothMix [166]	Neyman-Pearson	92.0%	88.9%	84.4%	78.6%	69.8%	60.7%	49.9%	40.2%	31.5%	22.2%	12.2%	4.9%
		DSRS	92.2%	89.0%	84.8%	79.7%	72.0%	63.9%	54.4%	46.2%	37.6%	29.2%	18.5%	7.2%

Due to the page limit, in the main text (Section 2.6, Table 2.2), we aggregate the certified robust accuracy across models with different smoothing variance $\sigma \in \{0.25, 0.50, 1.00\}$. To show the full landscape, we present the certified robust accuracy for each model trained with each variance. The evaluation protocol is the same as the one in the main text, and the tables for MNIST, CIFAR-10, and ImageNet models are Table B.5, Table B.6, and Table B.7 respectively. We observe that DSRS outperforms standard Neyman-Pearson certification for a wide range of radii.

Table B.6: CIFAR-10: Certified robust accuracy for models smoothed with different variance σ certified with different certification approaches.

Variance	Training Method	Certification Approach	Certified Accuracy under Radius r 0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00
0.25	Gaussian Aug. [77]	Neyman-Pearson	56.1%	35.7%	13.4%									
		DSRS	57.4%	39.4%	17.3%									
	Consistency [165]	Neyman-Pearson	61.8%	50.9%	34.7%									
		DSRS	62.5%	52.5%	37.8%									
	SmoothMix [166]	Neyman-Pearson	63.9%	53.3%	38.4%									
		DSRS	64.7%	55.5%	41.1%									
0.50	Gaussian Aug. [77]	Neyman-Pearson	53.7%	41.3%	27.7%	17.1%	9.1%	2.8%						
		DSRS	54.1%	42.7%	30.6%	20.3%	12.6%	4.0%						
	Consistency [165]	Neyman-Pearson	49.2%	43.9%	38.0%	32.3%	23.8%	18.1%						
		DSRS	49.6%	44.1%	38.7%	35.2%	28.1%	19.7%						
	SmoothMix [166]	Neyman-Pearson	53.2%	47.6%	40.2%	34.2%	26.7%	19.6%						
		DSRS	53.3%	48.5%	42.1%	35.9%	29.4%	21.7%						
1.00	Gaussian Aug. [77]	Neyman-Pearson	40.2%	32.6%	24.7%	18.9%	14.9%	10.2%	7.5%	4.1%	2.0%	0.7%	0.1%	0.1%
		DSRS	40.3%	33.1%	25.9%	20.6%	16.1%	12.5%	8.4%	6.4%	3.5%	1.8%	0.7%	0.1%
	Consistency [165]	Neyman-Pearson	37.2%	32.6%	29.6%	25.9%	22.5%	19.0%	16.4%	13.8%	11.2%	9.0%	7.1%	5.1%
		DSRS	37.1%	32.5%	29.8%	27.1%	23.5%	20.9%	17.6%	15.3%	13.1%	10.9%	8.9%	6.5%
	SmoothMix [166]	Neyman-Pearson	43.2%	39.5%	33.9%	29.1%	24.0%	20.4%	17.0%	13.9%	10.3%	7.8%	4.9%	2.3%
		DSRS	43.2%	39.7%	34.9%	30.0%	25.8%	22.1%	18.7%	16.1%	13.2%	10.2%	7.1%	3.9%

Table B.7: ImageNet: Certified robust accuracy for models smoothed with different variance σ certified with different certification approaches.

Variance	Training Method	Certification Approach	Certified Accuracy under Radius r 0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00
0.25	Gaussian Aug. [77]	Neyman-Pearson	57.1%	41.6%	17.4%									
		DSRS	58.4%	47.9%	24.4%									
	Consistency [165]	Neyman-Pearson	59.8%	49.8%	36.9%									
		DSRS	60.4%	52.4%	40.4%									
	SmoothMix [166]	Neyman-Pearson	46.7%	38.2%	28.2%									
		DSRS	47.4%	40.0%	29.8%									
0.50	Gaussian Aug. [77]	Neyman-Pearson	53.3%	47.0%	39.3%	33.2%	24.5%	17.0%						
		DSRS	54.1%	48.4%	41.4%	35.3%	28.8%	19.1%						
	Consistency [165]	Neyman-Pearson	53.6%	48.3%	43.3%	36.8%	31.4%	24.5%						
		DSRS	53.7%	49.9%	44.7%	39.3%	34.8%	27.4%						
	SmoothMix [166]	Neyman-Pearson	38.7%	33.5%	28.8%	24.6%	18.1%	13.5%						
		DSRS	39.1%	34.9%	30.3%	26.8%	21.6%	15.6%						
1.00	Gaussian Aug. [77]	Neyman-Pearson	42.5%	37.2%	33.0%	29.2%	24.8%	21.4%	17.6%	13.7%	10.2%	7.8%	5.7%	3.6%
		DSRS	42.5%	38.1%	34.4%	30.2%	27.0%	23.3%	21.3%	18.7%	14.2%	11.0%	9.0%	5.7%
	Consistency [165]	Neyman-Pearson	40.0%	38.3%	34.2%	31.8%	28.7%	25.6%	22.1%	19.1%	16.1%	14.0%	10.6%	8.5%
		DSRS	40.2%	38.5%	35.4%	32.6%	30.7%	28.1%	25.4%	22.6%	19.6%	17.4%	14.1%	10.4%
	SmoothMix [166]	Neyman-Pearson	29.8%	25.6%	21.8%	19.2%	17.0%	14.2%	11.8%	10.1%	8.9%	7.2%	6.0%	4.6%
		DSRS	29.7%	26.2%	23.0%	20.6%	18.0%	15.7%	14.0%	12.1%	9.9%	8.4%	7.2%	5.3%

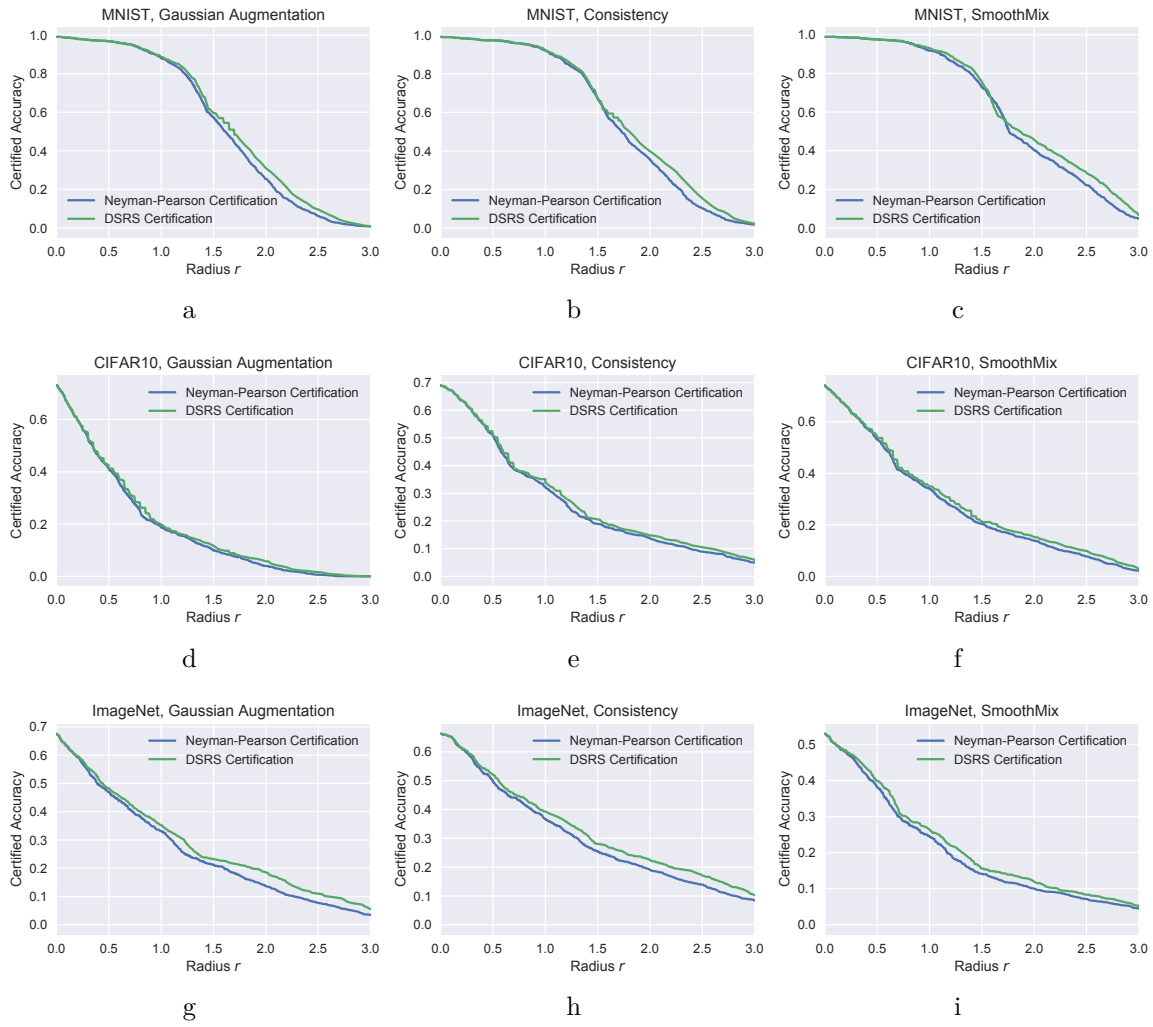


Figure B.2: Certified robust accuracy - radius curve corresponding to Table 2.2.

Curves

Following the convention [77, 317], we plot the certified robust accuracy - radius curve in Figure B.2.

The curves correspond to the certified robust accuracy data in Table 2.2 (in Section 2.6), i.e., the certified robust accuracy under each radius r is the maximum certified robust accuracy among models trained with variance $\sigma \in \{0.25, 0.50, 1.00\}$. We observe that among all medium to large radii (including those not shown in Table 2.2), DSRS provides higher certified robust accuracy than Neyman-Pearson certification. The margin of DSRS is relatively small on CIFAR-10 but is apparent on MNIST and ImageNet. Especially, the margins on ImageNet reflect that DSRS is particularly effective on large datasets.

Table B.8: Average certified radius (ACR) statistics. The smoothing variance hyperparameter $\sigma = 1.00$. The evaluation protocol is the same as that in the main text.

Training Method	Certification	MNIST	CIFAR-10	ImageNet
Gaussian Augmentation	Neyman-Pearson	1.550	0.447	0.677
	DSRS	1.629	0.469	0.750
	Relative Improvement	**+5.10%**	**+4.92%**	**+10.78%**
Consistency	Neyman-Pearson	1.645	0.636	0.796
	DSRS	1.730	0.659	0.862
	Relative Improvement	**+5.17%**	**+3.62%**	**+8.29%**
SmoothMix	Neyman-Pearson	1.716	0.676	0.490
	DSRS	1.806	0.712	0.525
	Relative Improvement	**+5.24%**	**+5.33%**	**+7.14%**

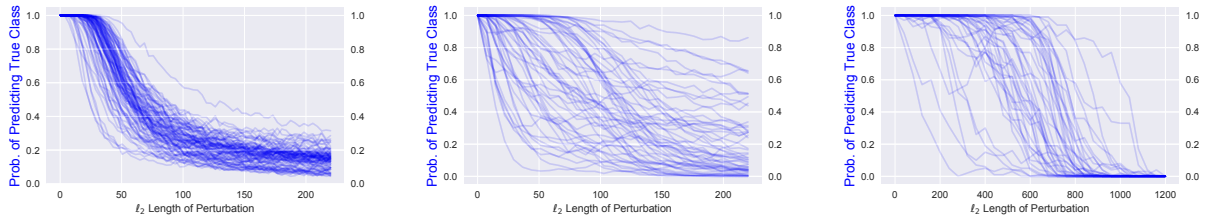
ACR Results

In the literature, another common metric of certified robustness is ACR (average certified radius) [165, 166, 437]. In Table B.8, we report the ACR comparison between Neyman-Pearson-based certification and our DSRS certification. Across the three smoothing variance choices $\sigma \in \{0.25, 0.50, 1.00\}$, we find $\sigma = 1.00$ yields the highest ACR, so we only report the ACR for models smoothed with $\sigma = 1.00$. As we can see, in all cases, DSRS significantly improves over Neyman-Pearson-based certification in terms of ACR.

B.5.4 Using Distribution with Different Variance as $\mathcal{Q}$

We take the models trained with Gaussian augmentation [77] and variance $\sigma = 1.00$ as examples. We use "DSRS-trunc" to represent DSRS using truncated generalized Gaussian as $\mathcal{Q}$, and "DSRS-var" to represent DSRS using generalized Gaussian with different variance as $\mathcal{Q}$, and compare their robustness certification (i.e., certified robust accuracy) in Table B.9. From the table, we find that on MNIST and CIFAR-10, DSRS-var is better than DSRS-trunc, whereas on ImageNet, DSRS-trunc is slightly better than DSRS-var. Both DSRS-trunc and DSRS-var are significantly better than Neyman-Pearson-based certification.

To investigate the reason, we follow the protocols for studying the concentration property in Appendix B.5.1 to plot the landscape of models on MNIST, CIFAR-10, and ImageNet, as shown in Figure B.3. From the figure, we find that the curves on ImageNet are generally steeper, which corresponds to that the concentration property is better satisfied on ImageNet. Therefore, we conjecture that when the concentration property (see Definition 2.3) is better satisfied, DSRS with truncated Gaussian as $\mathcal{Q}$ is better than Gaussian with different variance as $\mathcal{Q}$.



a MNIST model. b CIFAR-10 model. c ImageNet model.

Figure B.3: Probability of true-prediction w.r.t. ℓ_2 length of perturbations for base classifiers from Gaussian augmentation training studied in Appendix B.5.4. Figures are plotted following the same protocol as in Appendix B.5.1.

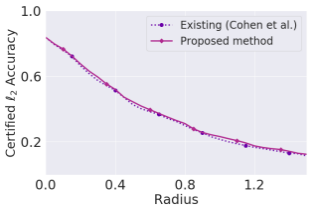
Table B.9: Comparison of DSRS certified robust accuracy with different types of additional smoothing distribution $\mathcal{Q}$ and Neyman-Pearson-based certification. Detail experiment settings are in Appendix B.5.4.

Dataset	Certification	Certified Accuracy under Radius r											
		0.25	0.50	0.75	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00
MNIST	Neyman-Pearson	**95.2%**	**91.9%**	87.7%	80.6%	71.2%	57.6%	41.0%	25.5%	13.6%	6.2%	2.1%	0.9%
	DSRS-trunc	95.1%	91.8%	87.9%	81.3%	72.9%	60.2%	46.1%	30.9%	17.4%	9.4%	3.6%	**1.2%**
	DSRS-var	95.1%	91.8%	**88.2%**	**81.5%**	**73.6%**	**61.6%**	**48.4%**	**34.1%**	**21.0%**	**10.6%**	**4.4%**	**1.2%**
CIFAR-10	Neyman-Pearson	40.2%	32.6%	24.7%	18.9%	14.9%	10.2%	7.5%	4.1%	2.0%	0.7%	0.1%	**0.1%**
	DSRS-trunc	**40.3%**	32.9%	25.5%	20.1%	15.7%	11.5%	8.0%	5.5%	2.7%	1.5%	0.6%	**0.1%**
	DSRS-var	**40.3%**	**33.1%**	**25.9%**	**20.6%**	**16.1%**	**12.5%**	**8.4%**	**6.4%**	**3.5%**	**1.8%**	**0.7%**	**0.1%**
ImageNet	Neyman-Pearson	42.5%	37.2%	33.0%	29.2%	24.8%	21.4%	17.6%	13.7%	10.2%	7.8%	5.7%	3.6%
	DSRS-trunc	42.5%	38.1%	34.4%	30.2%	**27.0%**	**23.3%**	**21.3%**	**18.7%**	14.2%	**11.0%**	**9.0%**	**5.7%**
	DSRS-var	**42.9%**	**38.5%**	**35.0%**	**31.0%**	26.5%	23.2%	21.0%	18.3%	**14.6%**	10.5%	8.2%	5.3%

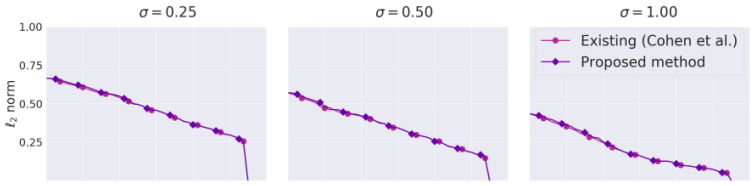
B.5.5 Comparison with Higher-Order Randomized Smoothing

It is difficult to have a direct comparison with higher-order randomized smoothing [215, 267], which is the only work to the best of our knowledge that uses additional information beyond P_A to tighten the robustness certification in randomized smoothing. This difficulty comes from the following reasons: (1) Higher-order randomized smoothing only supports standard Gaussian smoothing, while DSRS is particularly useful with generalized Gaussian smoothing. (2) All experiment evaluations in higher-order randomized smoothing are conducted with large sampling numbers ($N = 2 \times 10^5$ on CIFAR-10 and $N = 1.25 \times 10^6$ on ImageNet) that makes the evaluation costly, while DSRS is designed for practical sampling numbers ($N = 10^5$). (3) The code is not open-source yet [267]. Nonetheless, we can directly compare with the curves provided by Mohapatra et al. [267].

We digest the certified robust accuracy vs. ℓ_2 radius r curves from [267] in Figure B.4. As we can see, compared with Neyman-Pearson-based certification, the improvements from higher-order randomized smoothing are small especially on the large ImageNet dataset despite the excessive sampling numbers (1.25×10^6). In contrast, as shown in Figure B.2,



a [267, Figure 2(b)]: Higher-order randomized smoothing on CIFAR-10.



b [267, Figure 4]: Higher-order randomized smoothing on ImageNet for models trained with different smoothing variances.

Figure B.4: Higher-order randomized smoothing certification (solid curves) compared with standard Neyman-Pearson-based certification (dotted curves).

within only 10^5 sampling number, DSRS is visibly tighter than Neyman-Pearson-based certification. In fact, to the best of our knowledge, DSRS is the first model-agnostic approach that is visibly tighter than Neyman-Pearson-based certification under ℓ_2 radius.

B.6 DISCUSSIONS ON GENERALIZING DSRS FRAMEWORK

In this appendix, we first introduce prior work that leverages additional information for certification in randomized smoothing, then generalize our DSRS as a more general framework to theoretically compare with the related work and highlight future directions.

B.6.1 Existing Work on Leveraging Additional Information for Certification

We discuss all known work that leverages more information to achieve tighter robustness certification for randomized smoothing prior to this paper to the best of our knowledge.

Additional Black-Box Information. Our DSRS leverages additional information to tighten the certification for randomized smoothing. We leverage the information from an additional smoothing distribution. This information can be obtained from the base classifier that we *only* have query access on the predicted label. We call the information from this limited query access "black-box" information. The certification approaches that only require black-box information can be applied to any classification model regardless of the model structure. Thus, they are usually more general and more scalable. The classical Neyman-Pearson certification only requires black-box information.

Besides our DSRS, the only other form of additional black-box information that is leveraged is the higher-order information [215, 267]. Formally, our additional black-box information

has the form

$$\Pr_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[F_0(\boldsymbol{x}+\boldsymbol{\epsilon})=y]. \tag{B.220}$$

In contrast, the higher-order information, especially second-order information used by Levine et al. [215], Mohapatra et al. [267] has the form

$$\|\nabla f_0^{\mathcal{P}}(\boldsymbol{x}_0)_{y_0}\|_p = \|\nabla \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}}[F_0(\boldsymbol{x}_0+\boldsymbol{\epsilon})=y_0]\|_p \tag{B.221}$$

that is also shown to be estimable given the black-box query access. However, the higher-order information has several limitations: (1) It is hard to leverage the information beyond the second order. Therefore, only second-order information is used in existing certification approaches yet. However, to achieve optimal tightness, one needs to leverage infinite orders of information, which brings an infinite number of constraints and is thus intractable. In contrast, we show that extra information from only one additional distribution suffices to derive a strongly tight certification. (2) In practice, the second-order information shows marginal improvements in the widely used ℓ_2 and ℓ_∞ certification settings on real-world datasets [215, 267] and even such improvements require a large number of samples (usually in million order instead of ours 10^5).

Dvijotham et al. [106] also propose to use additional information to tighten the robustness certification for randomized smoothing ("full-information" setting). They formalize the tightest possible certification and compare it with Neyman-Pearson-based certifiation ("information-limited" setting), but in practice, they do not try to leverage information from distributions other than $\mathcal{P}$.

Constraining Model Structure. If we discard the "black-box information" constraint, we can obtain tighter robustness certification than classical Neyman-Pearson. For example, knowing the model structure can benefit the certification. Lee et al. [205] show that when the base classifier is a decision tree, we can use dynamic programming to derive a strongly tight certification against ℓ_0-bounded perturbations. Awasthi et al. [16] show that, if the base classifier first performs a known low-rank projection, then works on the low-rank projected space, for the corresponding smoothed classifier, we can have a tighter certification on both ℓ_2 and ℓ_∞ settings. However, it is challenging to find a projection such that the model preserves satisfactory performance while the projection rank is low. Indeed, the approach is evaluated on DCT basis to show the improvement on ℓ_∞ certification, and there exists a gap between the actual achieved certified robustness and the state of the art. We do not compare with these approaches since they impose additional assumptions on the base classifiers so their applicable scenarios are limited, and currently, the state-of-the-art base classifier does

not satisfy their imposed constraints under ℓ_2 and ℓ_∞ certification settings. Recently, for ℓ_1 certification, a deterministic and improved smoothing approach (a type of non-additive smoothing mechanism) is proposed to handle the case where input images are constrained in space $\{0, 1/255, \cdots, 244/255, 1\}^d$ [214]. This could be viewed as constraining the attack space from another aspect and implies that certified robustness can be improved by better smoothing mechanisms, which is orthogonal to our work that focuses on certification for existing smoothing mechanisms.

Confidence Smoothing. A group of certification approaches assumes that the base classifier outputs normalized confidence on the given input, and the smoothed classifier predicts the class with the highest expectation of normalized confidence under noised input. This assumption can be viewed as a special type of "Constraining Model Structure". Under this assumption, we can query and approximate the *quantile* of the confidence under noised input: $F_0(\boldsymbol{x}_0 + \boldsymbol{\epsilon})$ where $\boldsymbol{\epsilon} \sim \mathcal{P}$. With this information, we can leverage the Neyman-Pearson lemma in a more delicate way to provide a tighter (higher) lower bound of the expected confidence under perturbation, i.e., $\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{P}} F_0(\boldsymbol{x} + \boldsymbol{\delta} + \boldsymbol{\epsilon})$.

These certification approaches provide tighter certification than the classical Neyman-Pearson for the smoothed classifier that predicts the class with the highest expected normalized confidence. They are also useful for regression tasks such as object detection in computer vision as shown in [70]. However, for the classification task, for utilizable base classifiers (i.e., benign accuracy $> 50\%$ under noise), if we simply set the predicted class to have 100% confidence, we only increase the certified radius of the classifier and the certification for this "one-hot" base classifier only requires classical Neyman-Pearson. Thus, these certification approaches, e.g., [194], may not achieve higher certified robustness on the classification task than Neyman-Pearson and therefore we do not compare with them.

B.6.2 General Framework

Focusing on the certification with additional black-box information, we generalize the DSRS to allow more general additional information.

Definition B.2 (General Additional Black-Box Information)**.** For given base classifier F_0, suppose the true label at $\boldsymbol{x}_0$ is y_0, for certifying robustness at $\boldsymbol{x}_0$, the general additional

black-box information has the form

$$\begin{cases} \displaystyle\int_{\mathbb{R}^d} f_1(\boldsymbol{x})\mathbb{I}\left[F_0(\boldsymbol{x}) = y_0\right] \mathrm{d}\boldsymbol{x} = c_1, \\ \cdots \\ \displaystyle\int_{\mathbb{R}^d} f_i(\boldsymbol{x})\mathbb{I}\left[F_0(\boldsymbol{x}) = y_0\right] \mathrm{d}\boldsymbol{x} = c_i, \\ \cdots \\ \displaystyle\int_{\mathbb{R}^d} f_N(\boldsymbol{x})\mathbb{I}\left[F_0(\boldsymbol{x}) = y_0\right] \mathrm{d}\boldsymbol{x} = c_N, \end{cases} \tag{B.222}$$

where f_i and c_i are pre-determined; f_i is integrable in $\mathbb{R}^d$ and $c_i \in \mathbb{R}$ for $1 \leqslant i \leqslant N$.

Remark B.4. Obtaining the information in Equation (B.222) requires only the black-box access to whether $F_0(\boldsymbol{x})$ equals to y_0. We define the general additional black-box information in this way because: (1) The information from finite points is useless since the smoothed classifier has zero probability mass on finite points, so the useful information is based on integration; (2) To provide a lower bound of $\widetilde{F}_0^{\mathcal{P}}(\boldsymbol{x}_0 + \boldsymbol{\delta})_{y_0}$, we only need to care whether $F_0(\boldsymbol{x})$ equals to y_0 in region of interest.

Examples. (1) Our DSRS, the additional information $\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{Q}}[f(\boldsymbol{\epsilon})] = Q_A$ instantiates Definition B.2 by setting $N = 1$, $f_1(\cdot) = q(\cdot - \boldsymbol{x}_0)$ and $c_1 = Q_A$. (2) In [215, 267], the second-order information $\nabla f_0^{\mathcal{P}}(\boldsymbol{x}_0)$ instantiates Definition B.2 by setting $N = d$, $f_i(\boldsymbol{x}) = -\nabla p(\boldsymbol{x} - \boldsymbol{x}_0)_i$, and $c_i = \left(\nabla f_0^{\mathcal{P}}(\boldsymbol{x}_0)\right)_i$ according to Theorem 1 in [267]. We remark that due to the sampling difficulty, instead of using the whole vector $\nabla f_0^{\mathcal{P}}(\boldsymbol{x}_0)$ as the information, second-order smoothing [215, 267] uses its p-norm in practice. However, using the full information only gives tighter certification so we consider this a more ideal variant.

Then, we can extend the constrained optimization problem ($\mathbf{C}$) in Section 2.5.1 to ($\mathbf{C}^{\mathrm{ext}}$) to accommodate the general information as such

$$\underset{f}{\text{minimize}} \quad \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\delta} + \boldsymbol{\epsilon})] \tag{B.223}$$

$$\text{s.t.} \quad \mathop{\mathbb{E}}_{\boldsymbol{\epsilon}\sim\mathcal{P}}[f(\boldsymbol{\epsilon})] = P_A, \tag{B.224}$$

$$\int_{\mathbb{R}^d} f_1(\boldsymbol{\epsilon}) f(\boldsymbol{\epsilon}) \,\mathrm{d}\boldsymbol{\epsilon} = c_1, \cdots, \int_{\mathbb{R}^d} f_N(\boldsymbol{\epsilon}) f(\boldsymbol{\epsilon}) \,\mathrm{d}\boldsymbol{\epsilon} = c_N, \tag{B.225}$$

$$0 \leqslant f(\boldsymbol{\epsilon}) \leqslant 1 \quad \forall \boldsymbol{\epsilon} \sim \mathbb{R}^d. \tag{B.226}$$

Similarly, by the strong duality (Theorem 2.3), to solve the certification problem

$$\forall \boldsymbol{\delta} \text{ s.t. } \|\boldsymbol{\delta}\|_p \leqslant r,\ \mathbf{C}_{\boldsymbol{\delta}}^{\text{ext}}(P_A,\, c_1,\, \ldots,\, c_N) > 0.5, \tag{B.227}$$

we only need to solve the dual problem ($\mathbf{D}^{\text{ext}}$):

$$\underset{\eta,\lambda_1,\ldots,\lambda_N \in \mathbb{R}}{\text{maximize}} \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}} \left[p(\boldsymbol{\epsilon}) < \eta p(\boldsymbol{\epsilon}+\boldsymbol{\delta}) + \sum_{i=1}^{N} \lambda_i f_i(\boldsymbol{\epsilon}+\boldsymbol{\delta}) \right] \tag{B.228}$$

$$\text{s.t. } \Pr_{\boldsymbol{\epsilon}\sim\mathcal{P}} \left[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \eta p(\boldsymbol{\epsilon}) + \sum_{i=1}^{N} \lambda_i f_i(\boldsymbol{\epsilon}) \right] = P_A, \tag{B.229}$$

$$\int_{\mathbb{R}^d} \mathbb{I}\left[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \eta p(\boldsymbol{\epsilon}) + \sum_{i=1}^{N} \lambda_i f_i(\boldsymbol{\epsilon}) \right] f_1(\boldsymbol{\epsilon})\, \mathrm{d}\boldsymbol{\epsilon} = c_1, \tag{B.230}$$

$$\ldots$$

$$\int_{\mathbb{R}^d} \mathbb{I}\left[p(\boldsymbol{\epsilon}-\boldsymbol{\delta}) < \eta p(\boldsymbol{\epsilon}) + \sum_{i=1}^{N} \lambda_i f_i(\boldsymbol{\epsilon}) \right] f_N(\boldsymbol{\epsilon})\, \mathrm{d}\boldsymbol{\epsilon} = c_N. \tag{B.231}$$

We remark that this generalization shares a similar spirit as one type of generalization of Neyman-Pearson Lemma [69, 267]. Following the same motivation, Dvijotham et al. [106] try to generalize the certification by adding more constraints in their "full-information setting". However, it is unclear whether their constraints in f-divergences form have the same expressive power as ours in practice (i.e., the practicality of theoretically tight Hockey-Stick divergence). A study of these different types of generalization would be our future work.

More importantly, we believe that the pipeline of DSRS can be adapted to solve this generalized dual problem. We hope that this generalization and the corresponding DSRS could enable the exploration and exploitation of more types of additional information for tightening the robustness certification of randomized smoothing.

Implications. The generalized DSRS framework allows us to explicitly compare different types of additional information. For example, comparing our additional distribution information and higher-order information, we find that (1) for additional distribution information, from Theorem 2.1 and Corollary B.1, the strong tightness can be acheived for $N = C - 1$ where C is the number of classes; (2) for higher-order information, from [267,

Asymptotic-Optimality Remark], the strong tightness can be achieved when all orders of information are used, i.e., $N \to \infty$. This comparison suggests that our additional information from another smoothing distribution should be more efficient.

Another implication is that, from Corollary B.1, under multiclass setting, with proper choices of the $(C-1)$ additional smoothing distributions, if we base DSRS on solving dual problem ($\mathbf{D}^{\text{ext}}$), the DSRS can achieve *strong tightness* in multiclass setting.

B.6.3 Limitations and Future Directions

Despite the promising theoretical and empirical results of DSRS, DSRS still has some limitations that open an avenue for future work. We list the following future directions: (1) tighter certification from a more ideal additional smoothing distribution $\mathcal{Q}$: there may exist better smoothing distribution $\mathcal{Q}$ or better methods to optimize hyperparameters in $\mathcal{Q}$ than what we have considered in this work in terms of certifying larger certified radius in practice; (2) better training approach for DSRS: there may be a large space for exploring training approaches that favor DSRS certification since all existing training methods are designed for Neyman-Pearson-based certification. We believe that advances in this aspect can boost the robustness certification with DSRS to achieve state-of-the-art certified robustness. (3) better additional information: more generally, besides the prediction probability from an additional smoothing distribution, there may exist other useful additional information for certification in randomized smoothing. We hope our generalization of the DSRS framework can inspire future work in tighter and more efficient certification for randomized smoothing.

APPENDIX C: APPENDIX FOR CHAPTER 3

C.1 DERIVING DUAL PROBLEM WITH CUTTING PLANES

In this section we derive the dual formulation for neural network certification problem with general cutting planes (or arbitrarily added linear constraints across any layers). Our derivation is based on the linearly relaxed MIP formulation with original binary variables $\boldsymbol{z}$ intact, to allow us to add cuts also to the integer variable (the mostly commonly used triangle relaxation does not have $\boldsymbol{z}$). We first write the full LP relaxation with arbitrary cutting planes, as well as their corresponding dual variables:

$$
\begin{aligned}
& f^*_{\text{LP-cut}} = \min_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{z}} f(x) && && \text{(C.1)}\\
\text{s.t.}\quad & f(\boldsymbol{x}) = \boldsymbol{x}^{(L)}; \quad \boldsymbol{x}_0 - \epsilon \leqslant \boldsymbol{x} \leqslant \boldsymbol{x}_0 + \epsilon; && && \text{(C.2)}\\
& \boldsymbol{x}^{(i)} = \mathbf{W}^{(i)}\hat{\boldsymbol{x}}^{(i-1)} + \boldsymbol{b}^{(i)}; \quad i \in [L], && \Rightarrow \boldsymbol{\nu}^{(i)} \in \mathbb{R}^{d_i} && \text{(C.3)}\\
& \hat{x}^{(i)}_j \geqslant 0;\ j \in \mathcal{I}^{(i)} && \Rightarrow \mu^{(i)}_j \in \mathbb{R} && \text{(C.4)}\\
& \hat{x}^{(i)}_j \geqslant x^{(i)}_j;\ j \in \mathcal{I}^{(i)} && \Rightarrow \tau^{(i)}_j \in \mathbb{R} && \text{(C.5)}\\
& \hat{x}^{(i)}_j \leqslant u^{(i)}_j z^{(i)}_j;\ j \in \mathcal{I}^{(i)} && \Rightarrow \gamma^{(i)}_j \in \mathbb{R} && \text{(C.6)}\\
& \hat{x}^{(i)}_j \leqslant x^{(i)}_j - l^{(i)}_j(1 - z^{(i)}_j);\ j \in \mathcal{I}^{(i)} && \Rightarrow \pi^{(i)}_j \in \mathbb{R} && \text{(C.7)}\\
& \hat{x}^{(i)}_j = x^{(i)}_j;\ j \in \mathcal{I}^{+(i)} && && \text{(C.8)}\\
& \hat{x}^{(i)}_j = 0;\ j \in \mathcal{I}^{-(i)} && && \text{(C.9)}\\
& 0 \leqslant z^{(i)}_j \leqslant 1; \quad j \in \mathcal{I}^{(i)},\ i \in [L] && && \text{(C.10)}\\
& \sum_{i=1}^{L-1} \left(\boldsymbol{H}^{(i)}\boldsymbol{x}^{(i)} + \boldsymbol{G}^{(i)}\hat{\boldsymbol{x}}^{(i)} + \boldsymbol{Q}^{(i)}\boldsymbol{z}^{(i)} \right) \leqslant \boldsymbol{d}. && \Rightarrow \boldsymbol{\beta} \in \mathbb{R}^N && \text{(C.11)}
\end{aligned}
$$

The Lagrangian function can be constructed as:

$$
\begin{aligned}
f^*_{\text{LP-cut}} = \min_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{z}} \max_{\boldsymbol{\nu},\boldsymbol{\mu},\boldsymbol{\tau},\boldsymbol{\gamma},\boldsymbol{\pi},\boldsymbol{\beta}} & \boldsymbol{x}^{(L)} + \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top} \left(\boldsymbol{x}^{(i)} - \mathbf{W}^{(i)}\hat{\boldsymbol{x}}^{(i-1)} + \boldsymbol{b}^{(i)} \right)\\
& + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{(i)}} \left[\mu^{(i)}_j(-\hat{x}^{(i)}_j) + \tau^{(i)}_j(x^{(i)}_j - \hat{x}^{(i)}_j) + \gamma^{(i)}_j(\hat{x}^{(i)}_j - u^{(i)}_j z^{(i)}_j) + \pi^{(i)}_j(\hat{x}^{(i)}_j - x^{(i)}_j + l^{(i)}_j - l^{(i)}_j z^{(i)}_j) \right]\\
& + \boldsymbol{\beta}^\top \left[\sum_{i=1}^{L-1} \left(\boldsymbol{H}^{(i)}\boldsymbol{x}^{(i)} + \boldsymbol{G}^{(i)}\hat{\boldsymbol{x}}^{(i)} + \boldsymbol{Q}^{(i)}\boldsymbol{z}^{(i)} \right) - \boldsymbol{d} \right]
\end{aligned}
\qquad \text{(C.12)}
$$

$$\text{s.t. } \hat{x}_j^{(i)} = 0, j \in \mathcal{I}^{-(i)}; \quad \hat{x}_j^{(i)} = x_j^{(i)}, j \in \mathcal{I}^{+(i)}; \quad \boldsymbol{x}_0 - \epsilon \leqslant \boldsymbol{x} \leqslant \boldsymbol{x}_0 + \epsilon; \quad 0 \leqslant z_j^{(i)} \leqslant 1, j \in \mathcal{I}^{(i)}$$
$$\boldsymbol{\mu} \geqslant 0; \quad \boldsymbol{\tau} \geqslant 0; \quad \boldsymbol{\gamma} \geqslant 0; \quad \boldsymbol{\pi} \geqslant 0; \quad \boldsymbol{\beta} \geqslant 0. \tag{C.13}$$

Here $\boldsymbol{\mu}, \boldsymbol{\tau}, \boldsymbol{\gamma}, \boldsymbol{\pi}$ are shorthands for all dual variables in each layer and each neuron. Note that for some constraints, their dual variables are not created because they are trivial to handle. Rearrange the equation and swap the min and max (strong duality) gives us:

$$\begin{aligned} f^*_{\text{LP-cut}} = \max_{\boldsymbol{\nu},\boldsymbol{\mu},\boldsymbol{\tau},\boldsymbol{\gamma},\boldsymbol{\pi},\boldsymbol{\beta}} \min_{\boldsymbol{x},\hat{\boldsymbol{x}},\boldsymbol{z}} \; & (\boldsymbol{\nu}^{(L)} + 1)\boldsymbol{x}^{(L)} - \boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}\hat{\boldsymbol{x}}^{(0)} \\ & + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{+(i)}} \left(\nu_j^{(i)} + \boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)} - \boldsymbol{\nu}^{(i+1)\top}\mathbf{W}_{:,j}^{(i+1)} + \boldsymbol{\beta}^\top \boldsymbol{G}_{:,j}^{(i)} \right) x_j^{(i)} \\ & + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{-(i)}} \left(\nu_j^{(i)} + \boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)} \right) x_j^{(i)} \\ & + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{(i)}} \Big[\left(\nu_j^{(i)} + \boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)} + \tau_j^{(i)} - \pi_j^{(i)} \right) x_j^{(i)} \\ & + \left(-\boldsymbol{\nu}^{(i)\top}\mathbf{W}_{:,j}^{(i)} - \mu_j^{(i)} - \tau_j^{(i)} + \gamma_j^{(i)} + \pi_j^{(i)} + \boldsymbol{\beta}^\top \boldsymbol{G}_{:,j}^{(i)} \right) \hat{x}_j^{(i)} \\ & + \left(-u_j^{(i)}\gamma_j^{(i)} - l_j^{(i)}\pi_j^{(i)} + \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \right) z_j^{(i)} \Big] \\ & - \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top}\boldsymbol{b}^{(i)} + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{(i)}} \pi_j^{(i)} l_j^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{d} \end{aligned} \tag{C.14}$$

$$\begin{aligned} \text{s.t.} \quad & \boldsymbol{x}_0 - \epsilon \leqslant \boldsymbol{x} \leqslant \boldsymbol{x}_0 + \epsilon; \quad 0 \leqslant z_j^{(i)} \leqslant 1, j \in \mathcal{I}^{(i)} \\ & \boldsymbol{\mu} \geqslant 0; \quad \boldsymbol{\tau} \geqslant 0; \quad \boldsymbol{\gamma} \geqslant 0; \quad \boldsymbol{\pi} \geqslant 0; \quad \boldsymbol{\beta} \geqslant 0. \end{aligned} \tag{C.15}$$

Here $\mathbf{W}_{:,j}^{(i+1)}$ denotes the j-th column of $\mathbf{W}^{(i+1)}$. Note that for the term involving $j \in \mathcal{I}^{+(i)}$ we have replaced $\hat{\boldsymbol{x}}^{(i)}$ with $\boldsymbol{x}^{(i)}$ to obtain the above equation. For the the term $x_j^{(i)}, j \in \mathcal{I}^{+(i)}$ it is always 0 so does not appear.

Solving the inner minimization gives us the dual formulation:

$$\begin{aligned} f^*_{\text{LP-cut}} = \max_{\boldsymbol{\nu},\boldsymbol{\mu},\boldsymbol{\tau},\boldsymbol{\gamma},\boldsymbol{\pi},\boldsymbol{\beta}} \; & -\epsilon \|\boldsymbol{\nu}^{(1)\top}\mathbf{W}^{(1)}\boldsymbol{x}_0\|_1 - \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top}\boldsymbol{b}^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{d} \\ & + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{(i)}} \left[\pi_j^{(i)} l_j^{(i)} - \text{ReLU}(u_j^{(i)}\gamma_j^{(i)} + l_j^{(i)}\pi_j^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}) \right] \end{aligned} \tag{C.16}$$

$$\text{s.t. } \boldsymbol{\nu}^{(L)} = -1 \tag{C.17}$$

$$\nu_j^{(i)} = \boldsymbol{\nu}^{(i+1)\top}\mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top (\boldsymbol{H}_{:,j}^{(i)} + \boldsymbol{G}_{:,j}^{(i)}), \quad \text{for } j \in \mathcal{I}^{+(i)}, \; i \in [L-1] \tag{C.18}$$

$$\nu_j^{(i)} = -\boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)}, \quad \text{for } j \in \mathcal{I}^{-(i)},\ i \in [L-1] \tag{C.19}$$

$$\nu_j^{(i)} = \pi_j^{(i)} - \tau_j^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)} \quad \text{for } j \in \mathcal{I}^{(i)},\ i \in [L-1] \tag{C.20}$$

$$\left(\pi_j^{(i)} + \gamma_j^{(i)}\right) - \left(\mu_j^{(i)} + \tau_j^{(i)}\right) = \boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top \boldsymbol{G}_{:,j}^{(i)}, \quad \text{for } j \in \mathcal{I}^{(i)},\ i \in [L-1] \tag{C.21}$$

$$\text{s.t.} \quad \boldsymbol{\mu} \geqslant 0; \quad \boldsymbol{\tau} \geqslant 0; \quad \boldsymbol{\gamma} \geqslant 0; \quad \boldsymbol{\pi} \geqslant 0; \quad \boldsymbol{\beta} \geqslant 0. \tag{C.22}$$

The $\|\cdot\|_1$ comes from minimizing over $\boldsymbol{x}_0$ with the constraint $\boldsymbol{x}_0 - \epsilon \leqslant \boldsymbol{x} \leqslant \boldsymbol{x}_0 + \epsilon$, and the $\mathrm{ReLU}(\cdot)$ term comes from minimizing over $z_j^{(i)}$ with the constraint $0 \leqslant z_j^{(i)} \leqslant 1$.

Before we give the bound propagation procedure, we first give a technical lemma:

Lemma C.1. Given $u \geqslant 0$, $l \leqslant 0$, $\pi \geqslant 0$, $\gamma \geqslant 0$, and $\pi + \gamma = C$, and define the function:

$$g(\pi, \gamma) = -\mathrm{ReLU}(u\gamma + l\pi + q) + l\pi. \tag{C.23}$$

Then

$$\max_{\pi \geqslant 0, \gamma \geqslant 0} g(\pi, \gamma) = \begin{cases} l\pi^*, & \text{if } -uC \leqslant q \leqslant -lC \\ 0, & \text{if } q < -uC \\ -q, & \text{if } q > -lC \end{cases} \tag{C.24}$$

where the optimal values for π and γ are:

$$\pi^* = \max\left(\min\left(\frac{uC+q}{u-l}, C\right), 0\right), \quad \gamma^* = \max\left(\min\left(\frac{-lC-q}{u-l}, C\right), 0\right). \tag{C.25}$$

Proof. Case 1: when $u\gamma + l\pi + q \geqslant 0$, the objective becomes:

$$\max_{\pi \geqslant 0, \gamma \geqslant 0} g(\pi, \gamma) = -u\gamma - q \tag{C.26}$$

$$\text{s.t. } \pi + \gamma = C; \quad u\gamma + l\pi + q \geqslant 0. \tag{C.27}$$

Since q is a constant and the object only involves a non-negative variable γ with non-positive coefficient $-u$, γ needs to be as smaller as possible as long as the constraint is satisfied. Substitute $\pi = C - \gamma$ into the constraint $u\gamma + l\pi + q \geqslant 0$,

$$u\gamma + l(C - \gamma) + q \geqslant 0 \Rightarrow \gamma \geqslant \frac{-lC - q}{u - l} \tag{C.28}$$

Considering γ is within $[0, C]$, the optimal γ and π are:

$$\gamma^* = \max\left(\min\left(\frac{-lC-q}{u-l}, C\right)\right), \quad \pi^* = \max\left(\min\left(\frac{uC+q}{u-l}, C\right)\right) \tag{C.29}$$

Case 2: when $u\gamma + l\pi + q \leqslant 0$, the objective becomes:

$$\max_{\pi\geqslant 0, \gamma\geqslant 0} g(\pi, \gamma) = l\pi \tag{C.30}$$

$$\text{s.t. } \pi + \gamma = C \tag{C.31}$$

$$u\gamma + l\pi + q \leqslant 0 \tag{C.32}$$

Since q is a constant and the object only involves a non-negative variable π with non-positive coefficient l, π needs to be as smaller as possible as long as the constraint is satisfied. Substitute $\gamma = C - \pi$ into the constraint $u\gamma + l\pi + q \leqslant 0$,

$$u(C-\pi) + l\pi + q \leqslant 0 \Rightarrow \pi \geqslant \frac{uC+q}{u-l}. \tag{C.33}$$

The optimal π^* and γ^* are the same as in case 1. The optimal objective can be obtained by substitute π^* and γ^* into $g(\pi, \gamma)$. The objective depends on q because when q is too large ($q \geqslant -lC$) or too small ($q \leqslant -uC$), π^* and γ^* are fixed at either 0 or C. QED.

With this lemma, we are now ready to prove Theorem 3.1, the bound propagation rule with general cutting planes (GCP-CROWN):

Theorem 3.1 (Bound propagation with general cutting planes)**.** Given the following bound propagation rule on $\boldsymbol{\nu}$ with optimizable parameter $0 \leqslant \alpha_j^{(i)} \leqslant 1$ and $\boldsymbol{\beta} \geqslant 0$:

$$\boldsymbol{\nu}^{(L)} = -1 \tag{C.34}$$

$$\nu_j^{(i)} = \boldsymbol{\nu}^{(i+1)\top}\mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top(\boldsymbol{H}_{:,j}^{(i)} + \boldsymbol{G}_{:,j}^{(i)}), \qquad j \in \mathcal{I}^{+(i)},\ i \in [L-1] \tag{C.35}$$

$$\nu_j^{(i)} = -\boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)}, \qquad j \in \mathcal{I}^{-(i)},\ i \in [L-1] \tag{C.36}$$

$$\hat{\nu}_j^{(i)} := \boldsymbol{\nu}^{(i+1)\top}\mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top \boldsymbol{G}_{:,j}^{(i)} \qquad j \in \mathcal{I}^{(i)},\ i \in [L-1] \tag{C.37}$$

$$\nu_j^{(i)} := \max\left(\min\left(\frac{u_j^{(i)}[\hat{\nu}_j^{(i)}]_+ + \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}}{u_j^{(i)} - l_j^{(i)}}, [\hat{\nu}_j^{(i)}]_+\right), 0\right) - \alpha_j^{(i)}[\hat{\nu}_j^{(i)}]_- - \boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)} \qquad j \in \mathcal{I}^{(i)},\ i \in [L-1] \tag{C.38}$$

Then $f^*_{\text{LP-cut}}$ is lower bounded by the following objective with any valid $0 \leqslant \boldsymbol{\alpha} \leqslant 1$ and $\boldsymbol{\beta} \geqslant 0$:

$$g(\boldsymbol{\alpha}, \boldsymbol{\beta}) = -\epsilon \|\boldsymbol{\nu}^{(1)\top} \mathbf{W}^{(1)} \boldsymbol{x}_0\|_1 - \sum_{i=1}^{L} \boldsymbol{\nu}^{(i)\top} \boldsymbol{b}^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{d} + \sum_{i=1}^{L-1} \sum_{j \in \mathcal{I}^{(i)}} h_j^{(i)}(\boldsymbol{\beta}) \tag{C.39}$$

where $h_j^{(i)}(\boldsymbol{\beta})$ is defined as:

$$h_j^{(i)}(\boldsymbol{\beta}) = \begin{cases} \frac{u_j^{(i)} l_j^{(i)} [\hat{\nu}_j^{(i)}]_+ + l_j^{(i)} \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}}{u_j^{(i)} - l_j^{(i)}} & \text{if} \quad l_j^{(i)} [\hat{\nu}_j^{(i)}]_+ \leqslant \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \leqslant u_j^{(i)} [\hat{\nu}_j^{(i)}]_+; \\ 0 & \text{if} \quad \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \geqslant u_j^{(i)} [\hat{\nu}_j^{(i)}]_+; \\ \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} & \text{if} \quad \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \leqslant l_j^{(i)} [\hat{\nu}_j^{(i)}]_+. \end{cases} \tag{C.40}$$

Proof. Given (C.21), for $j \in \mathcal{I}^{(i)}$, observing that the upper and lower bounds of ReLU relaxations for a single neuron cannot be tight simultaneously [406], $\pi_j^{(i)} + \gamma_j^{(i)}$ and $\mu_j^{(i)} + \tau_j^{(i)}$ cannot be both non-zero[19]. Thus, we have

$$\pi_j^{(i)} + \gamma_j^{(i)} = \left[\boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top \boldsymbol{G}_{:,j}^{(i)} \right]_+ := \left[\hat{\nu}_j^{(i)} \right]_+, \tag{C.41}$$

$$\mu_j^{(i)} + \tau_j^{(i)} = \left[\boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top \boldsymbol{G}_{:,j}^{(i)} \right]_- := \left[\hat{\nu}_j^{(i)} \right]_-. \tag{C.42}$$

To avoid clutter, we define $\hat{\nu}_j^{(i)} := \boldsymbol{\nu}^{(i+1)\top} \mathbf{W}_{:,j}^{(i+1)} - \boldsymbol{\beta}^\top \boldsymbol{G}_{:,j}^{(i)}$.

To derived the proposed bound propagation rule, in (C.16), we must eliminate variable $\gamma_j^{(i)}$ and $\pi_j^{(i)}$. Ignoring terms related to $\boldsymbol{\nu}$, we observe that we can optimize the term $\pi_j^{(i)} l_j^{(i)} - \text{ReLU}(u_j^{(i)} \gamma_j^{(i)} + l_j^{(i)} \pi_j^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)})$ for each j for $\mathcal{I}^{(i)}$ individually. We seek the optimal solution for the follow optimization problem on function h:

$$\max_{\pi_j^{(i)} \geqslant 0, \gamma_j^{(i)} \geqslant 0} h_j^{(i)}(\pi_j^{(i)}, \gamma_j^{(i)}; \boldsymbol{\beta}) : = \pi_j^{(i)} l_j^{(i)} - \text{ReLU}(u_j^{(i)} \gamma_j^{(i)} + l_j^{(i)} \pi_j^{(i)} - \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}) \tag{C.43}$$

$$\text{s.t.} \;\; \pi_j^{(i)} + \gamma_j^{(i)} = \left[\hat{\nu}_j^{(i)} \right]_+; \quad \pi_j^{(i)} \geqslant 0; \quad \gamma_j^{(i)} \geqslant 0. \tag{C.44}$$

Note that we optimize over $\pi_j^{(i)}$ and $\gamma_j^{(i)}$ here and treat $\boldsymbol{\beta}$ as a constant. Applying Lemma C.1 with $\pi = \pi_j^{(i)}$, $\gamma = \gamma_j^{(i)}$, $u = u_j^{(i)}$, $l = l_j^{(i)}$, $q = -\boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}$, $C = [\hat{\nu}_j^{(i)}]_+$ we obtain the optimal

[19] In theorey, it is possible that $\tau_j^{(i)} \neq 0$, $\gamma_j^{(i)} \neq 0$, $\pi_j^{(i)} = \mu_j^{(i)} = 0$ or $\pi_j^{(i)} \neq 0$, $\mu_j^{(i)} \neq 0$, $\tau_j^{(i)} = \gamma_j^{(i)} = 0$. This situation may happen when $u_j^{(i)}$ or $l_j^{(i)}$ is exactly tight and the solution $x_j^{(i)}$ is at the intersection of one lower and upper linear equatlities. Practically, this can be avoided by adding a small δ to $u_j^{(i)}$ or $l_j^{(i)}$, and in most scenarios $u_j^{(i)}$ are $l_j^{(i)}$ are loose bounds so this situation will not occur. The same argument is also applicable to [406].

objective for (C.43):

$$h_j^{(i)}(\boldsymbol{\beta}) = \begin{cases} l_j^{(i)}\pi_j^{(i)} & \text{if } \ l_j^{(i)}[\hat{\nu}_j^{(i)}]_+ \leqslant \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \leqslant u_j^{(i)}[\hat{\nu}_j^{(i)}]_+, \\ 0 & \text{if } \ \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \geqslant u_j^{(i)}[\hat{\nu}_j^{(i)}]_+, \\ \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} & \text{if } \ \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)} \leqslant l_j^{(i)}[\hat{\nu}_j^{(i)}]_+, \end{cases} \tag{C.45}$$

$$\pi_j^{(i)} = \max\left(\min\left(\frac{u_j^{(i)}[\hat{\nu}_j^{(i)}]_+ + \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}}{u_j^{(i)} - l_j^{(i)}}, C\right), 0\right). \tag{C.46}$$

Substitute this solution of $\pi_j^{(i)}$ into (C.20), and also observe that due to (C.42), $\tau_j^{(i)}$ is a variable between 0 and $\left[\hat{\nu}_j^{(i)}\right]_-$, so we can rewrite (C.20) as:

$$\nu_j^{(i)} := \max\left(\min\left(\frac{u_j^{(i)}[\hat{\nu}_j^{(i)}]_+ + \boldsymbol{\beta}^\top \boldsymbol{Q}_{:,j}^{(i)}}{u_j^{(i)} - l_j^{(i)}}, [\hat{\nu}_j^{(i)}]_+\right), 0\right) - \alpha_j^{(i)}[\hat{\nu}_j^{(i)}]_- - \boldsymbol{\beta}^\top \boldsymbol{H}_{:,j}^{(i)}. \tag{C.47}$$

Here $0 \leqslant \alpha_j^{(i)} \leqslant 1$ is an optimizable parameter that can be updated via its gradient during bound propagation, and any $0 \leqslant \alpha_j^{(i)} \leqslant 1$ produces valid lower bounds. All above substitutions replace each dual variable $\boldsymbol{\pi}$, $\boldsymbol{\gamma}$ and $\boldsymbol{\mu}$ to a valid setting in the dual formulation, so the final objective $g(\boldsymbol{\alpha}, \boldsymbol{\beta})$ is always a sound lower bound of $f^*_{\text{LP-cut}}$. This theorem establishes the soundness of GCP-CROWN. QED.

Note that an optimal setting of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ do not necessarily lead to the optimal primal value $f^*_{\text{LP-cut}}$. The main reason is that (C.43) gives a closed-form solution for $\boldsymbol{\pi}$ and $\boldsymbol{\gamma}$ to eliminate these variables without considering other dual variables like $\boldsymbol{\nu}$ to simplify the bound propagation process. To achieve theoretical optimality, $\boldsymbol{\pi}$ and $\boldsymbol{\gamma}$ can also be optimized.

C.2 DETAILS AND BACKGROUND OF GCP-CROWN WITH MIP CUTS

Branch-and-bound. Our work is based on branch-and-bound (BaB), a powerful framework for neural network certification [49] which many state-of-the-art verifiers are based on [115, 149, 286, 389]. Here we give a brief introduction for the concept of branch-and-bound for readers who are not familiar with this field.

We illustrate the branch-and-bound process in Figure C.1. Neural network certification seeks to minimize the objective (3.1): $f^* = \min_{\boldsymbol{x}} f(\boldsymbol{x}), \forall \boldsymbol{x} \in \mathcal{C}$, where $f(\boldsymbol{x})$ is a ReLU network under our setting. A positive f^* indicates that the network can be verified. However,

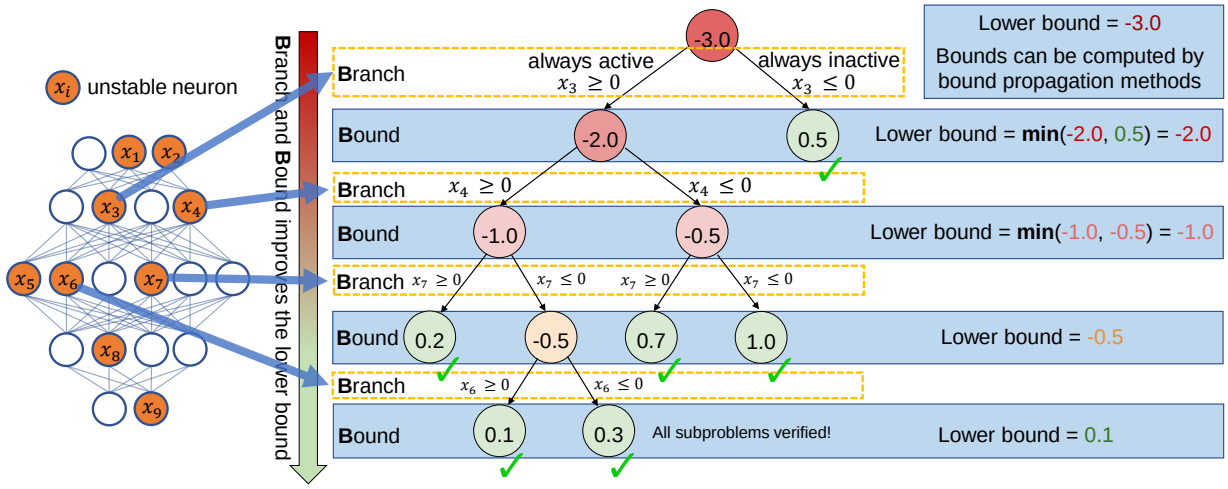


Figure C.1: **B**ranch and **B**ound (BaB) for neural network certification. A *unstable* ReLU neuron (e.g., x_1 may receive either negative or positive inputs for some input $\boldsymbol{x}$ in perturbation set $\mathcal{C}$. On the other hand, a stable ReLU neuron is always positive or negative for all $\boldsymbol{x} \in \mathcal{C}$, so it is a linear operation and branching is not needed.

since ReLU neurons are non-linear, this optimization problem is non-convex, and it usually requires us to relax ReLU neurons with linear constraints (e.g., the LP relaxation in (3.12)) to obtain a lower bound for objective f^*. This lower bound (-3.0 at the root node in Figure C.1) might become negative, even if f^* (the unknown ground-truth) is positive. Branch-and-bound is a systematic method to tighten this lower bound. First, we split an unstable ReLU neuron, such as x_3, into two cases: $x_3 \geqslant 0$ and $x_3 \leqslant 0$ (the *branching step*), producing two subproblems each with an additional constraint $x_3 \geqslant 0$ or $x_3 \leqslant 0$. In each subproblem, neuron x_3 does not need to be relaxed anymore, so the lower bound of f^* becomes tighter in each of the two subdomains in the subsequent *bounding step* (-3.0 becomes -2.0 and 0.5) in Figure C.1. Any subproblem with a positive lower bound is successfully verified and no further split is needed. Then, we repeat the branching and bounding steps on subproblems that still have a negative lower bound. We terminate when all unstable neurons are split or all subproblems are verified. In Figure 3, all leaf subproblems have positive lower bounds so the network is verified. On the other hand, if we have split all unstable neurons and there still exist domains with negative bounds, a counter-example can be constructed.

The contribution of this work is to improve the bounding step using cutting planes. As one can observe in Figure C.1, if we can make the bounds tighter, they approach to zero faster and less branching is needed. Since the number of the subproblems may be exponential to the number of unstable neurons in the worst case, it is crucial to improve the bounds quickly.

Informally, cutting planes are additional valid constraints for the relaxed subproblems, and adding these additional constraints as in (3.19) makes the lower bounds tighter. We refer the readers to integer programming literature [36, 78] for more details on cutting plane methods. Existing neural network verifiers mostly use bound propagation method [286, 343, 389, 425, 443] for the bounding step thanks to their efficiency and scalability, but none of them can handle general cutting planes. Our GCP-CROWN is the first bound propagation method that supports general cutting planes, and we demonstrated its effectiveness in Section 3.4 when combined with cutting planes generated by a MIP solver.

Soundness and Completeness. GCP-CROWN is sound because Theorem 3.1 guarantees that we always obtain a sound lower bound of the certification problem as long as valid cutting planes (generated by a MIP solver in our case) are added. Our method, when combined with branch-and-bound, is also complete because we provide a strict improvement in the bounding step over existing methods such as [389], and the arguments for completeness in existing verifiers using branch-and-bound on splitting ReLU neurons such as [48, 389] are still valid for this work.

C.3 ADDITIONAL EXPERIMENT DETAILS

C.3.1 Experimental Setup

Our experiments are conducted on a desktop with an AMD Ryzen 9 5950X CPU, one NVIDIA RTX 3090 GPU (24GB GPU memory), and 64GB CPU memory. Our implementation is based on the open-source α,β-CROWN verifier[20] with cutting plane related code added. Both α-CROWN+MIP and GCP-CROWN with MIP cuts use all 16 CPU cores and 1 GPU. `gurobi` is used as the MIP solver in α-CROWN+MIP. Although `gurobi` usually outperforms other MIP solvers for NN certification problems, it cannot export cutting planes, so our cutting planes are acquired by the `cplex` [160] solver (version 22.1.0.0). We use the Adam optimizer [183] to solve both $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ with 20 iterations. The learning rates are set as 0.1 and 0.02 (0.01 for `oval20`) for optimizing $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ respectively. We decay the learning rates with a factor of 0.9 (0.8 for `oval20`) per iteration. Timeout for properties in `oval20` are set as 1 hour follow the original benchmark. Timeout for `oval21` and `cifar10-resnet` are set as 720s and 300s respectively, the same as in VNN-COMP 2021. Timeout for SDP-FO models are set as 600s for α-CROWN+MIP and MN-BaB, and a shorter 200s timeout is

[20] `https://github.com/Verified-Intelligence/alpha-beta-CROWN`

Table C.1: Model structures used in our experiments. The notation Conv(a, b, c) stands for a conventional layer with a input channel, b output channels and a kernel size of $c \times c$. Linear(a, b) stands for a fully connected layer with a input features and b output features. ResBlock(a, b) stands for a residual block that has a input channels and b output channels. We have ReLU activation functions between two consecutive linear or convolutional layers.

Model name	Model structure	Batch size
Base (CIFAR-10)	Conv(3, 8, 4) - Conv(8, 16, 4) - Linear(1024, 100) - Linear(100, 10)	1024
Wide (CIFAR-10)	Conv(3, 16, 4) - Conv(16, 32, 4) - Linear(2048, 100) - Linear(100, 10)	1024
Deep (CIFAR-10)	Conv(3, 8, 4) - Conv(8, 8, 3) - Conv(8, 8, 3) - Conv(8, 8, 4) - Linear(412, 100) - Linear(100, 10)	1024
`cifar10-resnet2b`	Conv(3, 8, 3) - ResBlock(8, 16) - ResBlock(16, 16)) - Linear(1024, 100) - Linear(100, 10)	2048
`cifar10-resnet4b`	Conv(3, 16, 3) - ResBlock(16, 32) - ResBlock(32, 32)) - Linear(512, 100) - Linear(100, 10)	2048
CNN-A-Adv (MNIST)	Conv(1, 16, 4) - Conv(16, 32, 4) - Linear(1568, 100) - Linear(100, 10)	4096
CNN-A-Adv/-4 (CIFAR-10)	Conv(3, 16, 4) - Conv(16, 32, 4) - Linear(2048, 100) - Linear(100, 10)	4096
CNN-B-Adv/-4 (CIFAR-10)	Conv(3, 32, 5) - Conv(32, 128, 4) - Linear(8192, 250) - Linear(250, 10)	1024
CNN-A-Mix/-4 (CIFAR-10)	Conv(3, 16, 4) - Conv(16, 32, 4) - Linear(2048, 100) - Linear(100, 10)	4096

used for GCP-CROWN with MIP cuts.

We summarize the model structures and batch size used in our experiments in Table C.1. The CIFAR-10 Base, Wide and Deep models are used in `oval20` and `oval21` benchmarks.

C.3.2 A case study on cutting planes

The effectiveness of GCP-CROWN with MIP cuts motivates us to take a more careful look at the cutting planes generated by off-the-shelf MIP solvers, and understand how well it can contribute to tighten the lower bounds and strengthen certification performance. We use the `oval21` as a sample case study because GCP-CROWN with MIP cuts is very effective on this benchmark.

The `oval21` benchmark has 30 instances, each with 9 target labels (properties) to certify. Among the total of 270 properties, we filter out the easy cases where fast incomplete verifiers like α-CROWN can certify directly, and 39 *hard properties* remain which must be solved using branch and bound and/or cutting planes. Cutting planes are generated on these hard properties and they greatly help GCP-CROWN.

Number of Cuts Used to Certify Each Property. The maximal number of cuts applied per property is 4,162 and the minimal number is 318. On average, we have 1,683 cuts applied to our GCP-CROWN to solve these hard properties.

Improvements on Lower Bounds. In branch-and-bound, we lower bound the objective of Eq. 3.1 with ReLU split constraints [389]. A tighter lower bound can reduce the number

of branches used and usually leads to stronger certification. We measure how well the cuts generated by off-the-shelf solvers in improving the tightness of the lower bound for certification. Without branching and cutting planes, the average α-CROWN lower bound is -2.54. With generated cutting planes, we can improve the lower bound by 0.51 on average and can directly certify 4 out of 39 hard properties without branching. The lower bound without branching can be maximally improved by 1.54 and minimally improved by 0.04.

Structure of Generated Cuts. Finally, we investigate the variables involved in each generated cutting plane. In total, the MIP solver generates 65,647 cutting planes in total for the 39 hard properties. 65,301 of the cuts involve variables across multiple layers. It indicates that single layer cutting planes (e.g., constraints involving pre- and post-activation variables of a single ReLU layer only) commonly used in previous works [273, 342] are not optimal in general. Additionally, all of 65,647 cuts have at least one ReLU *integer variable* $\boldsymbol{z}^{(i)}$ used, which was not supported in existing bound propagation methods. 65,197 of all cuts involve at least one variable of input neurons. 23,600 of all cuts have at least one pre-ReLU variables and 51,617 of them have at least one post-ReLU variables.

APPENDIX D: APPENDIX FOR CHAPTER 4

D.1 DETAILED ANALYSIS AND PROOFS OF ENSEMBLE ROBUSTNESS

In this appendix, we first show the omitted theoretical results in Section 4.1, which are the robustness conditions for Max-Margin Ensemble (MME) and the comparison between the robustness of ensemble model and single model. We then present all proofs for these theoretical results.

D.1.1 Detailed Theoretical Results and Discussion

Here we present the theoretical results omitted from Section 4.1 along with some discussions.

Robustness Condition of MME

For MME, we have the following robustness condition.

Theorem D.1 (Gradient and Confidence Margin Condition for MME Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$, and $\mathcal{M}_{\text{MME}}$ as an MME defined over base models $\{F_1, F_2\}$. $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) = y_0$. Both F_1 and F_2 are β-smooth.

- (Sufficient Condition) If for any $y_1, y_2 \in [C]$ such that $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r}(f_1^{y_0/y_1}(\boldsymbol{x}_0) + f_2^{y_0/y_2}(\boldsymbol{x}_0)) - 2\beta r, \tag{D.1}$$

  then $\mathcal{M}_{\text{MME}}$ is r-robust at point $\boldsymbol{x}_0$.

- (Necessary Condition) Suppose for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : ||\boldsymbol{\delta}||_2 \leqslant r\}$, for any $i \in \{1, 2\}$, either $F_i(\boldsymbol{x}) = y_0$ or $F_i^{(2)}(\boldsymbol{x}) = y_0$. If $\mathcal{M}_{\text{MME}}$ is r-robust at point $\boldsymbol{x}_0$, then for any $y_1, y_2 \in [C]$ such that $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r}(f_1^{y_0/y_1}(\boldsymbol{x}_0) + f_2^{y_0/y_2}(\boldsymbol{x}_0)) + 2\beta r. \tag{D.2}$$

Comparing with the robustness conditions of MME (Theorem 4.1), the conditions for MME have highly similar forms. Thus, the discussion for ERI in main text (Equation (4.6)) still applies here, including the positive impact of diversified gradients and large confidence

margins towards MME ensemble robustness in both sufficient and necessary conditions and the implication of small model-smoothness bound β. A major distinction is that the condition for MME is limited to two base models. This is because the "maximum" operator in MME protocol poses difficulties for expressing the robust conditions in succinct continuous functions of base models' confidence. Therefore, Taylor expansion cannot be applied. We leave the extension to $N > 2$ base models as future work, and we conjecture the tendency would be similar as Equation (4.5). The theorem is proved in Appendix D.1.2.

Comparison between Ensemble Robustness and Single-Model Robustness

To compare the robustness of ensemble models and single models, we have the following corollary that is extended from Theorem 4.1 and Theorem D.1.

Corollary D.1 (Comparison of Ensemble and Single-Model Robustness)**.** Given an input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Suppose we have two β-smooth base models $\{F_1, F_2\}$, which are both r-robust at point $\boldsymbol{x}_0$. For any $\Delta \in [0, 1)$:

- (Weighted Ensemble) Define Weighted Ensemble $\mathcal{M}_{\text{WE}}$ with base models $\{F_1, F_2\}$. Suppose $\mathcal{M}_{\text{WE}}(\boldsymbol{x}_0) = y_0$. If for any label $y_i \neq y_0$, the base models' smoothness $\beta \leqslant \Delta \cdot \min\{f_1^{y_0/y_i}(\boldsymbol{x}_0), f_2^{y_0/y_i}(\boldsymbol{x}_0)\}/(c^2 r^2)$, and the gradient cosine similarity

$$\cos\langle \nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0), \nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0) \rangle \leqslant \cos\theta, \tag{D.3}$$

  then the $\mathcal{M}_{\text{WE}}$ with weights $\{w_1, w_2\}$ is at least R-robust at point $\boldsymbol{x}_0$ with

$$R = r \cdot \frac{1-\Delta}{1+\Delta}\left(1 - C_{\text{WE}}(1-\cos\theta)\right)^{-1/2}, \text{where} \tag{D.4}$$

  $C_{\text{WE}} = \min_{y_i : y_i \neq y_0} \frac{2 w_1 w_2 f_1^{y_0/y_i}(\boldsymbol{x}_0) f_2^{y_0/y_i}(\boldsymbol{x}_0)}{(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0))^2}$, $c = \max\{\frac{1-\Delta}{1+\Delta}\left(1 - C_{\text{WE}}(1-\cos\theta)\right)^{-1/2}, 1\}$.

- (Max-Margin Ensemble) Define Max-Margin Ensemble $\mathcal{M}_{\text{MME}}$ with the base models $\{F_1, F_2\}$. Suppose $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) = y_0$. If for any label $y_1 \neq y_0$ and $y_2 \neq y_0$, the base models' smoothness $\beta \leqslant \Delta \cdot \min\{f_1^{y_0/y_1}(\boldsymbol{x}_0), f_2^{y_0/y_2}(\boldsymbol{x}_0)\}/(c^2 r^2)$, and the gradient cosine similarity $\cos\langle \nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0), \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0) \rangle \leqslant \cos\theta$, then the $\mathcal{M}_{\text{MME}}$ is at least R-robust at point $\boldsymbol{x}_0$ with

$$R = r \cdot \frac{1-\Delta}{1+\Delta}\left(1 - C_{\text{MME}}(1-\cos\theta)\right)^{-1/2}, \text{where} \tag{D.5}$$

$$C_{\text{MME}} = \min_{\substack{y_1,y_2: \\ y_1,y_2\neq y_0}} \frac{2f_1^{y_0/y_1}(\boldsymbol{x}_0)f_2^{y_0/y_2}(\boldsymbol{x}_0)}{(f_1^{y_0/y_1}(\boldsymbol{x}_0)+f_2^{y_0/y_2}(\boldsymbol{x}_0))^2}, c = \max\{\tfrac{1-\Delta}{1+\Delta}\left(1 - C_{\text{MME}}(1-\cos\theta)\right)^{-1/2}, 1\}.$$

The proof is given in Appendix D.1.3.

Optimizing Weighted Ensemble. As we can observe from Corollary D.1, we can adjust the weights $\{w_1, w_2\}$ for Weighted Ensemble to change C_{WE} and the certified robust radius (Equation (D.4)). Then comes the problem of which set of weights can achieve the highest certified robust radius. Since larger C_{WE} results in higher radius, we need to choose

$$(w_1^{OPT}, w_2^{OPT}) = \arg\max_{w_1,w_2} \min_{y_i:y_i\neq y_0} \frac{2w_1w_2f_1^{y_0/y_i}(\boldsymbol{x}_0)f_2^{y_0/y_i}(\boldsymbol{x}_0)}{(w_1f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2f_2^{y_0/y_i}(\boldsymbol{x}_0))^2}. \tag{D.6}$$

Since this quantity is scale-invariant, we can fix w_1 and optimize over w_2 to get the optimal weights. In particular, if there are only two classes, we have a closed-form solution

$$\begin{aligned}(w_1^{OPT}, w_2^{OPT}) &= \arg\max_{w_1,w_2} \frac{2w_1w_2f_1^{y_0/y_1}(\boldsymbol{x}_0)f_2^{y_0/y_1}(\boldsymbol{x}_0)}{(w_1f_1^{y_0/y_1}(\boldsymbol{x}_0) + w_2f_2^{y_0/y_1}(\boldsymbol{x}_0))^2} \\ &= \{k\cdot f_2^{y_0/y_1}(\boldsymbol{x}_0), k\cdot f_1^{y_0/y_1}(\boldsymbol{x}_0) : k\in\mathbb{R}_+\},\end{aligned} \tag{D.7}$$

and corresponding C_{WE} achieves the maximum $1/2$.

For a special case—average weighted ensemble, we get the corresponding certified robust radius by setting $w_1 = w_2$ and plug the yielded

$$C_{\text{WE}} = \min_{y_i:y_i\neq y_0} \frac{2f_1^{y_0/y_i}(\boldsymbol{x}_0)f_2^{y_0/y_i}(\boldsymbol{x}_0)}{(f_1^{y_0/y_i}(\boldsymbol{x}_0) + f_2^{y_0/y_i}(\boldsymbol{x}_0))^2} \in (0, 1/2]. \tag{D.8}$$

into Equation (D.4).

Comparison between Ensemble and Single-model Robustness. The similar forms of R in the corollary allow us to discuss the Weighted Ensemble and Max-Margin Ensemble together. Specifically, we let C be either C_{WE} or C_{MME}, then

$$R = r\cdot\frac{1-\Delta}{1+\Delta}\left(1 - C(1-\cos\theta)\right)^{-1/2}. \tag{D.9}$$

Since when $R > r$, both ensembles have higher certified robustness than the base models, we solve this condition for $\cos\theta$:

$$R > r \iff \left(\frac{1-\Delta}{1+\Delta}\right)^2 > 1 - C(1-\cos\theta) \iff \cos\theta \leqslant 1 - \frac{4\Delta}{C(1+\Delta)^2}. \tag{D.10}$$

Notice that $C \in (0, 1/2]$. From this condition, we can easily observe that when the gradient cosine similarity is smaller, it is more likely that the ensemble has higher certified robustness than the base models. When the model is smooth enough, according to the condition on β, we can notice that Δ could be close to zero. As a result, $1-\frac{4\Delta}{C(1+\Delta)^2}$ is close to 1. *Thus, unless the gradient of base models is (or close to) colinear, it always holds that the ensemble (either WE or MME) has higher certified robustness than the base models.*

Larger Certified Radius with Larger Number of Base Models N. Following the same methodology, we can further observe that larger number of base models N can lead to larger certified radius as the following proposition shows.

Proposition D.1 (More Base Models Lead to Higher Certified Robustness of Weighted Ensemble)**.** At clean input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$, suppose all base models $\{f_i\}_{i=1}^{N+M}$ are β-smooth. Suppose the Weighted Ensemble $\mathcal{M}_1$ of base models $\{f_i\}_{i=1}^{N}$ and $\mathcal{M}_2$ of base models $\{f_i\}_{i=N+1}^{N+M}$ are both r-robust according to the sufficient condition in Theorem 4.1, and for any $y_i \neq y_0$ the $\mathcal{M}_1$ and $\mathcal{M}_2$'s ensemble gradients ($\sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0)$ and $\sum_{j=N+1}^{N+M} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0)$) are non-zero and not colinear, then the Weighted Ensemble $\mathcal{M}$ of $\{f_i\}_{i=1}^{N+M}$ is r'-robust for some $r' > r$.

Proof of Proposition D.1. For any $y_i \neq y_0$, since both $\mathcal{M}_1$ and $\mathcal{M}_2$ are r-robust according to the sufficient condition of Theorem 4.1, we have

$$\Big\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^{N} f_j^{y_0/y_i}(\boldsymbol{x}_0) - \beta r \sum_{j=1}^{N} w_j, \tag{D.11}$$

$$\Big\| \sum_{j=N+1}^{N+M} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=N+1}^{N+M} f_j^{y_0/y_i}(\boldsymbol{x}_0) - \beta r \sum_{j=N+1}^{N+M} w_j. \tag{D.12}$$

Adding above two inequalties we get

$$\Big\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 + \Big\| \sum_{j=N+1}^{N+M} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^{N+M} f_j^{y_0/y_i}(\boldsymbol{x}_0) - \beta r \sum_{j=1}^{N+M} w_j. \tag{D.13}$$

Since gradients of ensemble are not colinear and non-zero, from the triangle inequality,

$$\Big\| \sum_{j=1}^{N+M} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 < \Big\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 + \Big\| \sum_{j=N+1}^{N+M} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \tag{D.14}$$

and thus

$$\Big\| \sum_{j=1}^{N+M} w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 < \frac{1}{r} \sum_{j=1}^{N+M} f_j^{y_0/y_i}(\boldsymbol{x}_0) - \beta r \sum_{j=1}^{N+M} w_j, \tag{D.15}$$

which means we can increase r to r' and still keep the inequality hold with "$\leqslant$", and in turn certify a larger radius r' according to Theorem 4.1. QED.

Since DRT imposes diversified gradients via GD Loss, the "not colinear" condition easily holds for DRT ensemble, and therefore the proposition implies larger number of base models N lead to higher certified robustness of WE. For MME, we empirically observe similar trends.

Robustness Conditions for Smoothed Ensemble Models

Following the discussion in Section 4.1.3, for *smoothed* WE and MME, with the model-smoothness bound (Theorem 4.2), we can concretize the general robustness conditions in this way.

We define the *soft* smoothed confidence function $\bar{g}_f^{\epsilon}(\boldsymbol{x}) := \mathbb{E}_{\epsilon} f(\boldsymbol{x} + \epsilon)$. This definition is also used in the literature [193, 317, 437]. As a result, we revise the ensemble protocols of WE and MME by replacing the original confidences $\{f_i(\boldsymbol{x}_0)\}_{i=1}^N$ with these soft smoothed confidences $\{\bar{g}_i^{\epsilon}(\boldsymbol{x}_0)\}_{i=1}^N$. These protocols then choose the predicted class by treating these $\{\bar{g}_i^{\epsilon}(\boldsymbol{x}_0)\}_{i=1}^N$ as the base models' confidence scores. In the experiments, we did not actually evaluate these revised protocols since their robustness performance are expected to be similar as original ones [193, 317, 437]. The derived results connect smoothed ensemble robustness with the confidence scores.

Corollary D.2 (Gradient and Confidence Margin Conditions for Smoothed WE Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Let $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ be a Gaussian random variable. Define soft smoothed confidence $\bar{g}_i^{\epsilon}(\boldsymbol{x}) := \mathbb{E}_{\epsilon} f_i(\boldsymbol{x} + \epsilon)$ for each base model F_i ($1 \leqslant i \leqslant N$). The $\bar{G}^{\epsilon}_{\mathcal{M}_{\mathrm{WE}}}$ is a WE defined over soft smoothed base models $\{\bar{g}_i^{\epsilon}\}_{i=1}^N$ with weights $\{w_i\}_{i=1}^N$. $\bar{G}^{\epsilon}_{\mathcal{M}_{\mathrm{WE}}}(\boldsymbol{x}_0) = y_0$.

- (Sufficient Condition) The $\bar{G}^{\epsilon}_{\mathcal{M}_{\mathrm{WE}}}$ is r-robust at point $\boldsymbol{x}_0$ if for any $y_i \neq y_0$,

$$\Big\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} (\bar{g}_j^{\epsilon})^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^{N} w_j (\bar{g}_j^{\epsilon})^{y_0/y_i}(\boldsymbol{x}_0) - \frac{2r}{\sigma^2} \sum_{j=1}^{N} w_j, \tag{D.16}$$

- (Necessary Condition) If $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{WE}}}$ is r-robust at point $\boldsymbol{x}_0$, for any $y_i \neq y_0$,

$$\Big\| \sum_{j=1}^{N} w_j \nabla_{\boldsymbol{x}} (\bar{g}_j^{\epsilon})^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^{N} w_j (\bar{g}_j^{\epsilon})^{y_0/y_i}(\boldsymbol{x}_0) + \frac{2r}{\sigma^2} \sum_{j=1}^{N} w_j. \tag{D.17}$$

Corollary D.3 (Gradient and Confidence Margin Condition for Smoothed MME Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Let $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ be a Gaussian random variable. Define soft smoothed confidence $\bar{g}_i^{\epsilon}(\boldsymbol{x}) := \mathbb{E}_{\epsilon} f_i(\boldsymbol{x} + \epsilon)$ for either base model F_1 or F_2. The $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ is a MME defined over soft smoothed base models $\{\bar{g}_1^{\epsilon}, \bar{g}_2^{\epsilon}\}$. $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}(\boldsymbol{x}_0) = y_0$.

- (Sufficient Condition) If for any $y_1, y_2 \in [C]$ such that $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}} (\bar{g}_1^{\epsilon})^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} (\bar{g}_2^{\epsilon})^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r}((\bar{g}_1^{\epsilon})^{y_0/y_1}(\boldsymbol{x}_0) + (\bar{g}_2^{\epsilon})^{y_0/y_2}(\boldsymbol{x}_0)) - \frac{4r}{\sigma^2}, \tag{D.18}$$

  then $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ is r-robust at point $\boldsymbol{x}_0$.

- (Necessary Condition) Suppose for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : ||\boldsymbol{\delta}||_2 \leqslant r\}$, for any $i \in \{1, 2\}$, either $G_{F_i}(\boldsymbol{x}) = y_0$ or $G^{(2)}_{F_i}(\boldsymbol{x}) = y_0$. If $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ is r-robust at point $\boldsymbol{x}_0$, then for any $y_1, y_2 \in [C]$ such that $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}} (\bar{g}_1^{\epsilon})^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} (\bar{g}_2^{\epsilon})^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r}((\bar{g}_1^{\epsilon})^{y_0/y_1}(\boldsymbol{x}_0) + (\bar{g}_2^{\epsilon})^{y_0/y_2}(\boldsymbol{x}_0)) + \frac{4r}{\sigma^2}. \tag{D.19}$$

Remark D.1. The above two corollaries are extended from Theorem 4.1 and Theorem D.1 respectively, and correspond to our discussion in Section 4.1.3. We defer the proofs to Appendix D.1.5. From these two corollaries, we can explicit see that the *Ensemble-before-Smoothing* (see Definition 4.5) provides smoothed classifiers $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{WE}}}$ and $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ with bounded model-smoothness; and we can see the correlation between the robustness conditions and gradient diversity/confidence margin for smoothed ensembles.

D.1.2 Proofs of Robustness Conditions for General Ensemble Models

This subsection contains the proofs of robustness conditions. First, we connect the prediction of ensemble models with the arithmetic relations of confidence scores of base models. This connection is straightforward to establish for Weighted Ensemble (shown in Proposition D.2), but nontrivial for Max-Margin Ensemble (shown in Theorem D.2). Then, we prove the desired robustness conditions using Taylor expansion with Lagrange reminder.

Proposition D.2 (Robustness Condition for WE)**.** Consider an input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$, and an ensemble model $\mathcal{M}_{\text{WE}}$ constructed by base models $\{F_i\}_{i=1}^N$ with weights $\{w_i\}_{i=1}^N$. Suppose $\mathcal{M}_{\text{WE}}(\boldsymbol{x}_0) = y_0$. Then, the ensemble $\mathcal{M}_{\text{WE}}$ is r-robust at point $\boldsymbol{x}_0$ if and only if for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 \leqslant r\}$,

$$\min_{y_i \in [C]: y_i \neq y_0} \sum_{j=1}^{N} w_j f_j^{y_0/y_i}(\boldsymbol{x}) \geqslant 0. \tag{D.20}$$

Proof of Proposition D.2. According the the definition of r-robust, we know $\mathcal{M}_{\text{WE}}$ is r-robust if and only if for any point $\boldsymbol{x} := \boldsymbol{x}_0 + \boldsymbol{\delta}$ where $\|\boldsymbol{\delta}\|_2 \leqslant r$, $\mathcal{M}_{\text{WE}}(\boldsymbol{x}_0 + \boldsymbol{\delta}) = y_0$, which means that for any other label $y_i \neq y_0$, the confidence score for label y_0 is larger or equal than the confidence score for label y_i. It means that

$$\sum_{j=1}^{N} w_j f_j(\boldsymbol{x})_{y_0} \geqslant \sum_{j=1}^{N} w_j f_j(\boldsymbol{x})_{y_i} \tag{D.21}$$

for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 \leqslant r\}$. Since this should hold for any $y_i \neq y_0$, we have the sufficient and necessary condition

$$\min_{y_i \in [C]: y_i \neq y_0} \sum_{j=1}^{N} w_j f_j^{y_0/y_i}(\boldsymbol{x}) \geqslant 0. \tag{D.22}$$

QED.

Theorem D.2 (Robustness Condition for MME)**.** Consider an input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Let $\mathcal{M}_{\text{MME}}$ be an MME defined over base models $\{F_i\}_{i=1}^N$. Suppose: (1) $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) = y_0$; (2) for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 \leqslant r\}$, given any base model $i \in [N]$, either $F_i(\boldsymbol{x}) = y_0$ or $F_i^{(2)}(\boldsymbol{x}) = y_0$. Then, the ensemble $\mathcal{M}_{\text{MME}}$ is r-robust at point $\boldsymbol{x}_0$ *if and only if* for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 \leqslant r\}$,

$$\max_{i \in [N]} \min_{y_i \in [C]: y_i \neq y_0} f_i^{y_0/y_i}(\boldsymbol{x}) \geqslant \max_{i \in [N]} \min_{y_i' \in [C]: y_i' \neq y_0} f_i^{y_i'/y_0}(\boldsymbol{x}). \tag{D.23}$$

The theorem states the sufficient and necessary robustness condition for MME. We divide the two directions into the following two lemmas and prove them separately. We mainly use the alternative form of Equation (D.23) as such in the following lemmas and their proofs:

$$\max_{i \in [N]} \min_{y_i \in [C]: y_i \neq y_0} f_i^{y_0/y_i}(\boldsymbol{x}) + \min_{i \in [N]} \min_{y_i' \in [C]: y_i' \neq y_0} f_i^{y_0/y_i'}(\boldsymbol{x}) \geqslant 0. \tag{D.24}$$

Lemma D.1 (Sufficient Condition for MME)**.** Let $\mathcal{M}_{\text{MME}}$ be an MME defined over base models $\{F_i\}_{i=1}^N$. For any input $\boldsymbol{x}_0 \in \mathbb{R}^d$, the Max-Margin Ensemble $\mathcal{M}_{\text{MME}}$ predicts $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) = y_0$ *if*

$$\max_{i\in[N]} \min_{y_i\in[C]:y_i\neq y_0} f_i^{y_0/y_i}(\boldsymbol{x}_0) + \min_{i\in[N]} \min_{y_i'\in[C]:y_i'\neq y_0} f_i^{y_0/y_i'}(\boldsymbol{x}_0) \geqslant 0. \tag{D.25}$$

Proof of Lemma D.1. For brevity, for $i \in [N]$, we denote $y_i := F_i(\boldsymbol{x}_0), y_i' := F_i^{(2)}(\boldsymbol{x}_0)$ for each base model's top class and runner-up class at point $\boldsymbol{x}_0$.

Suppose $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) \neq y_0$, then according to ensemble definition (see Definition 4.3), there exists $c \in [N]$, such that $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) = F_c(\boldsymbol{x}_0) = y_c$, and

$$\forall i \in [N],\ i \neq c,\ f_c(\boldsymbol{x}_0)^{y_c/y_c'} > f_i(\boldsymbol{x}_0)^{y_i/y_i'}. \tag{D.26}$$

Because $y_c \neq y_0$, we have $f_c(\boldsymbol{x}_0)_{y_0} \leqslant f_c(\boldsymbol{x}_0)_{y_c'}$, so that $f_c(\boldsymbol{x}_0)^{y_c/y_0} \geqslant f_c(\boldsymbol{x}_0)^{y_c/y_c'}$. Now consider any model F_i where $i \in [N]$, we would like to show that there exists $y^* \neq y_0$, such that $f_i(\boldsymbol{x}_0)^{y_i/y_i'} \geqslant f_i(\boldsymbol{x}_0)^{y_0/y^*}$:

- If $y_i = y_0$, let $y^* := y_i'$, trivially $f_i(\boldsymbol{x}_0)^{y_i/y_i'} = f_i(\boldsymbol{x}_0)^{y_0/y^*}$;
- If $y_i \neq y_0$, and $y_i' \neq y_0$, we let $y^* := y_i'$, then $f_i(\boldsymbol{x}_0)^{y_i/y_i'} = f_i(\boldsymbol{x}_0)^{y_i/y^*} \geqslant f_i(\boldsymbol{x}_0)^{y_0/y^*}$;
- If $y_i \neq y_0$, but $y_i' = y_0$, we let $y^* := y_i$, then $f_i(\boldsymbol{x}_0)^{y_i/y_i'} = f_i(\boldsymbol{x}_0)^{y_i/y_0} \geqslant f_i(\boldsymbol{x}_0)^{y_0/y_i} = f_i(\boldsymbol{x}_0)^{y_0/y^*}$.

Combine the above findings with (D.26), we have:

$$\forall i \in [N],\ i \neq c,\ \exists y_c^* \in [C] \text{ and } y_c^* \neq y_0,\ \exists y_i^* \in [C] \text{ and } y_i^* \neq y_0,\ f_c(\boldsymbol{x}_0)^{y_c^*/y_0} > f_i(\boldsymbol{x}_0)^{y_0/y_i^*}. \tag{D.27}$$

Therefore, its negation

$$\exists i \in [N],\ i \neq c,\ \forall y_c^* \in [C] \text{ and } y_c^* \neq y_0,\ \forall y_i^* \in [C] \text{ and } y_i^* \neq y_0,\ f_c(\boldsymbol{x}_0)^{y_0/y_c^*} + f_i(\boldsymbol{x}_0)^{y_0/y_i^*} \geqslant 0 \tag{D.28}$$

implies $\mathcal{M}(\boldsymbol{x}_0) = y_0$. Since Equation (D.28) holds for any y_c^* and y_i^*, the equation is equivalent to

$$\exists i \in [N],\ i \neq c,\ \min_{y_c\in[C]:y_c\neq y_0} f_c(\boldsymbol{x}_0)^{y_0/y_c}(\boldsymbol{x}_0) + \min_{y_i'\in[C]:y_i'\neq y_0} f_i(\boldsymbol{x}_0)^{y_0/y_i'}(\boldsymbol{x}_0) \geqslant 0. \tag{D.29}$$

The existence qualifier over i can be replaced by maximum:

$$\min_{y_c \in [C]: y_c \neq y_0} f_c(\boldsymbol{x}_0)^{y_0/y_c}(\boldsymbol{x}_0) + \max_{i \in [N]} \min_{y_i' \in [C]: y_i' \neq y_0} f_i(\boldsymbol{x}_0)^{y_0/y_i'}(\boldsymbol{x}_0) \geqslant 0. \tag{D.30}$$

It is implied by

$$\max_{i \in [N]} \min_{y_i \in [C]: y_i \neq y_0} f_i^{y_0/y_i}(\boldsymbol{x}_0) + \min_{i \in [N]} \min_{y_i' \in [C]: y_i' \neq y_0} f_i^{y_0/y_i'}(\boldsymbol{x}_0) \geqslant 0. \tag{D.23}$$

Thus, Equation (D.23) is a sufficient condition for $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) = y_0$. QED.

Lemma D.2 (Necessary Condition for MME)**.** For any input $\boldsymbol{x}_0 \in \mathbb{R}^d$, if for any base model $i \in [N]$, either $F_i(\boldsymbol{x}_0) = y_0$ or $F_i^{(2)}(\boldsymbol{x}_0) = y_0$, then Max-Margin Ensemble $\mathcal{M}_{\text{MME}}$ predicting $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) = y_0$ implies

$$\max_{i \in [N]} \min_{y_i \in [C]: y_i \neq y_0} f_i^{y_0/y_i}(\boldsymbol{x}_0) + \min_{i \in [N]} \min_{y_i' \in [C]: y_i' \neq y_0} f_i^{y_0/y_i'}(\boldsymbol{x}_0) \geqslant 0. \tag{D.23}$$

Proof of Lemma D.2. Similar as before, for brevity, for $i \in [N]$, we denote $y_i := F_i(\boldsymbol{x}_0), y_i' := F_i^{(2)}(\boldsymbol{x}_0)$ for each base model's top class and runner-up class at point $\boldsymbol{x}_0$.

Suppose Equation (D.23) is not satisfied, it means that

$$\exists c \in [N],\, \exists y_c^* \in [C] \,\text{and}\, y_c^* \neq y_0,\, \forall i \in [N],\, \exists y_i^* \in [C] \,\text{and}\, y_i^* \neq y_0,\, f_c^{y_c^*/y_0}(\boldsymbol{x}_0) > f_i^{y_0/y_i^*}(\boldsymbol{x}_0). \tag{D.31}$$

- If $y_c = y_0$, then $f_c^{y_c^*/y_0}(\boldsymbol{x}_0) \leqslant 0$, which implies that $f_i^{y_0/y_i^*}(\boldsymbol{x}_0) < 0$, and hence $F_i(\boldsymbol{x}_0) \neq y_0$. Moreover, we know that $f_i^{y_i/y_i'}(\boldsymbol{x}_0) = f_i^{y_i/y_0}(\boldsymbol{x}_0) \geqslant f_i^{y_i^*/y_0}(\boldsymbol{x}_0) > f_c^{y_0/y_c^*}(\boldsymbol{x}_0) \geqslant f_c^{y_0/y_c'}(\boldsymbol{x}_0) = f_c^{y_c/y_c'}(\boldsymbol{x}_0)$ so $\mathcal{M}(\boldsymbol{x}_0) \neq F_c(\boldsymbol{x}_0) = y_0$.
- If $y_c \neq y_0$, i.e., $y_c' = y_0$, then $f_c^{y_c/y_0}(\boldsymbol{x}_0) \geqslant f_c^{y_c^*/y_0}(\boldsymbol{x}_0) > f_i^{y_0/y_1^*}(\boldsymbol{x}_0)$. If $F_i(\boldsymbol{x}_0) = y_0$, then $f_i^{y_0/y_i^*}(\boldsymbol{x}_0) \geqslant f_i^{y_0/y_i'}(\boldsymbol{x}_0) = f_i^{y_i/y_i'}(\boldsymbol{x}_0)$. Thus, $f_c^{y_c/y_c'}(\boldsymbol{x}_0) = f_c^{y_c/y_0}(\boldsymbol{x}_0) > f_i^{y_i/y_i'}(\boldsymbol{x}_0)$. As the result, $\mathcal{M}(\boldsymbol{x}_0) = F_c(\boldsymbol{x}_0) \neq y_0$.

For both cases, we show that $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) \neq y_0$, i.e., Equation (D.23) is a necessary condition for $\mathcal{M}(\boldsymbol{x}_0) = y_0$. QED.

Proof of Theorem D.2. Lemmas D.1 and D.2 are exactly the two directions (necessary and sufficient condition) of $\mathcal{M}_{\text{MME}}$ predicting label y_0 at point $\boldsymbol{x}$. Therefore, if the condition (Equation (D.23)) holds for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 \leqslant r\}$, the ensemble $\mathcal{M}_{\text{MME}}$ is r-robust at point $\boldsymbol{x}_0$; vice versa. QED.

For comparison, here we list the trivial robustness condition for single model.

Fact D.1 (Robustness Condition for Single Model)**.** Consider an input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Suppose a model F satisfies $F(\boldsymbol{x}_0) = y_0$. Then, the model F is r-robust at point $\boldsymbol{x}_0$ if and only if for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : \|\boldsymbol{\delta}\|_2 \leqslant r\}$,

$$\min_{y_i \in [C]: y_i \neq y_0} f^{y_0/y_i}(\boldsymbol{x}) \geqslant 0. \tag{D.32}$$

The fact is apparent given that the model predicts the class with the highest confidence.

Now we are ready to apply Taylor expansion to derive the robustness conditions shown in main text.

Theorem 4.1 (Gradient and Confidence Margin Condition for WE Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$, and $\mathcal{M}_{\text{WE}}$ as a WE defined over base models $\{F_i\}_{i=1}^N$ with weights $\{w_i\}_{i=1}^N$. $\mathcal{M}_{\text{WE}}(\boldsymbol{x}_0) = y_0$. All base model F_i's are β-smooth.

- (Sufficient Condition) $\mathcal{M}_{\text{WE}}$ is r-robust at point $\boldsymbol{x}_0$ if for any $y_i \neq y_0$,

$$\Big\| \sum_{j=1}^N w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^N w_j f_j^{y_0/y_i}(\boldsymbol{x}_0) - \beta r \sum_{j=1}^N w_j. \tag{D.33}$$

- (Necessary Condition) If $\mathcal{M}_{\text{WE}}$ is r-robust at point $\boldsymbol{x}_0$, then for any $y_i \neq y_0$,

$$\Big\| \sum_{j=1}^N w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^N w_j f_j^{y_0/y_i}(\boldsymbol{x}_0) + \beta r \sum_{j=1}^N w_j. \tag{D.34}$$

Proof of Theorem 4.1. From Taylor expansion with Lagrange remainder and the β-smoothness assumption on the base models, we have

$$\begin{aligned} \sum_{j=1}^N w_j f_j^{y_0/y_i}(\boldsymbol{x}_0) - r \Big\| \sum_{j=1}^N w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 - \frac{1}{2} r^2 \sum_{j=1}^N (2\beta w_j) &\leqslant \min_{\boldsymbol{x}: \|\boldsymbol{x} - \boldsymbol{x}_0\|_2 \leqslant r} \sum_{j=1}^N w_j f_j^{y_0/y_i}(\boldsymbol{x}) \\ &\leqslant \sum_{j=1}^N w_j f_j^{y_0/y_i}(\boldsymbol{x}_0) - r \Big\| \sum_{j=1}^N w_j \nabla_{\boldsymbol{x}} f_j^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 + \frac{1}{2} r^2 \sum_{j=1}^N (2\beta w_j), \end{aligned} \tag{D.35}$$

where the term $-\frac{1}{2} r^2 \sum_{j=1}^N (2\beta w_j)$ and $\frac{1}{2} r^2 \sum_{j=1}^N (2\beta w_j)$ are bounded from Lagrange remainder. Note that the difference $f_j^{y_0/y_i}$ is (2β)-smooth instead of β-smooth since it is the difference of two β-smooth function, and thus $\sum_{j=1}^N w_j f_j^{y_0/y_i}$ is $\sum_{j=1}^N (2\beta w_j)$-smooth. From Proposition D.2, the sufficient and necessary condition of WE's r-robustness is $\sum_{j=1}^N w_j f_j^{y_0/y_i}(\boldsymbol{x}) \geqslant 0$

for any $y_i \in [C]$ such that $y_i \neq y_0$, and any $\boldsymbol{x} = \boldsymbol{x}_0 + \boldsymbol{\delta}$ where $\|\boldsymbol{\delta}\|_2 \leqslant r$. Plugging this term into Equation (D.35) we get the theorem. QED.

Theorem D.1 (Gradient and Confidence Margin Condition for MME Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$, and $\mathcal{M}_{\text{MME}}$ as an MME defined over base models $\{F_1, F_2\}$. $\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0) = y_0$. Both F_1 and F_2 are β-smooth.

- (Sufficient Condition) If for any $y_1, y_2 \in [C]$ such that $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r}(f_1^{y_0/y_1}(\boldsymbol{x}_0) + f_2^{y_0/y_2}(\boldsymbol{x}_0)) - 2\beta r, \tag{D.36}$$

  then $\mathcal{M}_{\text{MME}}$ is r-robust at point $\boldsymbol{x}_0$.

- (Necessary Condition) Suppose for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : ||\boldsymbol{\delta}||_2 \leqslant r\}$, for any $i \in \{1, 2\}$, either $F_i(\boldsymbol{x}) = y_0$ or $F_i^{(2)}(\boldsymbol{x}) = y_0$. If $\mathcal{M}_{\text{MME}}$ is r-robust at point $\boldsymbol{x}_0$, then for any $y_1, y_2 \in [C]$ such that $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r}(f_1^{y_0/y_1}(\boldsymbol{x}_0) + f_2^{y_0/y_2}(\boldsymbol{x}_0)) + 2\beta r. \tag{D.37}$$

Proof of Theorem D.1. We prove the sufficient condition and necessary condition separately.

- (Sufficient Condition)
  From Lemma D.1, since there are only two base models, we can simplify the sufficient condition for $\mathcal{M}_{\text{MME}}(\boldsymbol{x}) = y_0$ as

$$\min_{y_i \in [C]: y_i \neq y_0} f_1^{y_0/y_i}(\boldsymbol{x}) + \min_{y_i' \in [C]: y_i' \neq y_0} f_2^{y_0/y_i'}(\boldsymbol{x}) \geqslant 0. \tag{D.38}$$

  In other words, for any $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$f_1^{y_0/y_1}(\boldsymbol{x}) + f_2^{y_0/y_2}(\boldsymbol{x}) \geqslant 0. \tag{D.39}$$

  With Taylor expansion and model-smoothness assumption, we have

$$\begin{aligned} &\min_{\boldsymbol{x}: \|\boldsymbol{x} - \boldsymbol{x}_0\|_2 \leqslant r} f_1^{y_0/y_1}(\boldsymbol{x}) + f_2^{y_0/y_2}(\boldsymbol{x}) \\ \geqslant &f_1^{y_0/y_1}(\boldsymbol{x}_0) + f_2^{y_0/y_2}(\boldsymbol{x}_0) - r\|\nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\|_2 - \frac{1}{2} \cdot 4\beta r^2. \end{aligned} \tag{D.40}$$

  Plugging this into Equation (D.39) yields the sufficient condition.

In the above equation, the term $-\frac{1}{2}\cdot 4\beta r^2$ is bounded from Lagrange remainder. Here, the 4β term comes from the fact that $f_1^{y_0/y_1}(\boldsymbol{x}) + f_2^{y_0/y_2}(\boldsymbol{x})$ is (4β)-smooth since it is the sum of difference of β-smooth function.

- (Necessary Condition)
  From Lemma D.2, similarly, the necessary condition for $\mathcal{M}_{\mathrm{MME}}(\boldsymbol{x}) = y_0$ is simplified to: for any $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$f_1^{y_0/y_1}(\boldsymbol{x}) + f_2^{y_0/y_2}(\boldsymbol{x}) \geqslant 0. \tag{D.41}$$

  Again, from Taylor expansion, we have

$$\begin{aligned}&\min_{\boldsymbol{x}:\|\boldsymbol{x}-\boldsymbol{x}_0\|_2\leqslant r} f_1^{y_0/y_1}(\boldsymbol{x}) + f_2^{y_0/y_2}(\boldsymbol{x}) \\ \leqslant & f_1^{y_0/y_1}(\boldsymbol{x}_0) + f_2^{y_0/y_2}(\boldsymbol{x}_0) - r\|\nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\|_2 + \frac{1}{2}\cdot 4\beta r^2.\end{aligned} \tag{D.42}$$

  Plugging this into Equation (D.39) yields the necessary condition.

  In the above equation, the term $+\frac{1}{2}\cdot 4\beta r^2$ is bounded from Lagrange remainder. The 4β term appears because of the same reason as before.

QED.

Since we will compare the robustness of ensemble models and the single model, we show the corresponding conditions for single-model robustness.

Proposition D.3 (Gradient and Confidence Margin Conditions for Single-Model Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Model $F(\boldsymbol{x}_0) = y_0$, and it is β-smooth.

- (Sufficient Condition) If for any $y_1 \in [C]$ such that $y_1 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}} f^{y_0/y_1}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r} f^{y_0/y_1}(\boldsymbol{x}_0) - \beta r, \tag{D.43}$$

  F is r-robust at point $\boldsymbol{x}_0$.

- (Necessary Condition) If F is r-robust at point $\boldsymbol{x}_0$, for any $y_1 \in [C]$ such that $y_1 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}} f^{y_0/y_1}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r} f^{y_0/y_1}(\boldsymbol{x}_0) + \beta r. \tag{D.44}$$

Proof of Proposition D.3. This proposition is apparent given the following inequality from Taylor expansion

$$f^{y_0/y_1}(\boldsymbol{x}_0)-r\|\nabla_{\boldsymbol{x}} f^{y_0/y_1}(\boldsymbol{x}_0)\|_2-\beta r^2 \leqslant \min_{\boldsymbol{x}:\|\boldsymbol{x}-\boldsymbol{x}_0\|_2\leqslant r} f^{y_0/y_1}(\boldsymbol{x}) \leqslant f^{y_0/y_1}(\boldsymbol{x}_0)-r\|\nabla_{\boldsymbol{x}} f^{y_0/y_1}(\boldsymbol{x}_0)\|_2+\beta r^2 \tag{D.45}$$

and the sufficient and necessary robust condition in Fact D.1. QED.

D.1.3 Proof of Robustness Comparison Results between Ensemble Models and Single Models

Corollary D.1 (Comparison of Ensemble and Single-Model Robustness)**.** Given an input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Suppose we have two β-smooth base models $\{F_1, F_2\}$, which are both r-robust at point $\boldsymbol{x}_0$. For any $\Delta \in [0, 1)$:

- (Weighted Ensemble) Define Weighted Ensemble $\mathcal{M}_{\mathrm{WE}}$ with base models $\{F_1, F_2\}$. Suppose $\mathcal{M}_{\mathrm{WE}}(\boldsymbol{x}_0) = y_0$. If for any label $y_i \neq y_0$, the base models' smoothness $\beta \leqslant \Delta \cdot \min\{f_1^{y_0/y_i}(\boldsymbol{x}_0), f_2^{y_0/y_i}(\boldsymbol{x}_0)\}/(c^2 r^2)$, and the gradient cosine similarity

$$\cos\langle \nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0), \nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0)\rangle \leqslant \cos\theta, \tag{D.46}$$

  then the $\mathcal{M}_{\mathrm{WE}}$ with weights $\{w_1, w_2\}$ is at least R-robust at point $\boldsymbol{x}_0$ with

$$R = r \cdot \frac{1-\Delta}{1+\Delta}\left(1 - C_{\mathrm{WE}}(1-\cos\theta)\right)^{-1/2}, \text{where} \tag{D.47}$$

  $C_{\mathrm{WE}} = \min_{y_i: y_i \neq y_0} \frac{2w_1 w_2 f_1^{y_0/y_i}(\boldsymbol{x}_0) f_2^{y_0/y_i}(\boldsymbol{x}_0)}{(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0))^2}, c = \max\{\frac{1-\Delta}{1+\Delta}\left(1 - C_{\mathrm{WE}}(1-\cos\theta)\right)^{-1/2}, 1\}.$

- (Max-Margin Ensemble) Define Max-Margin Ensemble $\mathcal{M}_{\mathrm{MME}}$ with the base models $\{F_1, F_2\}$. Suppose $\mathcal{M}_{\mathrm{MME}}(\boldsymbol{x}_0) = y_0$. If for any label $y_1 \neq y_0$ and $y_2 \neq y_0$, the base models' smoothness $\beta \leqslant \Delta \cdot \min\{f_1^{y_0/y_1}(\boldsymbol{x}_0), f_2^{y_0/y_2}(\boldsymbol{x}_0)\}/(c^2 r^2)$, and the gradient cosine similarity $\cos\langle \nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0), \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\rangle \leqslant \cos\theta$, then the $\mathcal{M}_{\mathrm{MME}}$ is at least R-robust at point $\boldsymbol{x}_0$ with

$$R = r \cdot \frac{1-\Delta}{1+\Delta}\left(1 - C_{\mathrm{MME}}(1-\cos\theta)\right)^{-1/2}, \text{where} \tag{D.48}$$

  $C_{\mathrm{MME}} = \min_{\substack{y_1, y_2: \\ y_1, y_2 \neq y_0}} \frac{2 f_1^{y_0/y_1}(\boldsymbol{x}_0) f_2^{y_0/y_2}(\boldsymbol{x}_0)}{(f_1^{y_0/y_1}(\boldsymbol{x}_0) + f_2^{y_0/y_2}(\boldsymbol{x}_0))^2}, c = \max\{\frac{1-\Delta}{1+\Delta}\left(1 - C_{\mathrm{MME}}(1-\cos\theta)\right)^{-1/2}, 1\}.$

Proof of Corollary D.1. We first prove the theorem for Weighted Ensemble. For arbitrary

$y_i \neq y_0$, we have

$$\|w_1 \nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 \nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0)\|_2 \tag{D.49}$$

$$= \sqrt{w_1^2 \|\nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0)\|_2^2 + w_2^2 \|\nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0)\|_2^2 + 2w_1 w_2 \langle \nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0),\, f_2^{y_0/y_i}(\boldsymbol{x}_0)\rangle} \tag{D.50}$$

$$\leqslant \sqrt{w_1^2 \|\nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0)\|_2^2 + w_2^2 \|\nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0)\|_2^2 + 2w_1 w_2 \|\nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0)\|_2 \|\nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0)\|_2 \cos\theta} \tag{D.51}$$

$$\overset{(i.)}{\leqslant} \sqrt{w_1^2 \left(\frac{1}{r} f_1^{y_0/y_i}(\boldsymbol{x}_0) + \beta r\right)^2 + w_2^2 \left(\frac{1}{r} f_2^{y_0/y_i}(\boldsymbol{x}_0) + \beta r\right)^2 + 2w_1 w_2 \left(\frac{1}{r} f_1^{y_0/y_i}(\boldsymbol{x}_0) + \beta r\right)\left(\frac{1}{r} f_2^{y_0/y_i}(\boldsymbol{x}_0) + \beta r\right)\cos\theta} \tag{D.52}$$

$$= \frac{1}{r}\sqrt{w_1^2 \left(f_1^{y_0/y_i}(\boldsymbol{x}_0) + \beta r^2\right)^2 + w_2^2 \left(f_2^{y_0/y_i}(\boldsymbol{x}_0) + \beta r^2\right)^2 + 2w_1 w_2 \left(f_1^{y_0/y_i}(\boldsymbol{x}_0) + \beta r^2\right)\left(f_2^{y_0/y_i}(\boldsymbol{x}_0) + \beta r^2\right)\cos\theta} \tag{D.53}$$

$$\overset{(ii.)}{\leqslant} \frac{1}{r} \cdot \left(1 + \frac{\Delta}{c^2}\right) \sqrt{w_1^2 f_1^{y_0/y_i}(\boldsymbol{x}_0)^2 + w_2^2 f_2^{y_0/y_i}(\boldsymbol{x}_0)^2 + 2w_1 w_2 f_1^{y_0/y_i}(\boldsymbol{x}_0) f_2^{y_0/y_i}(\boldsymbol{x}_0)\cos\theta} \tag{D.54}$$

$$= \frac{1}{r} \cdot \left(1 + \frac{\Delta}{c^2}\right) \sqrt{\left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right)^2 - 2(1-\cos\theta) w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)} \tag{D.55}$$

$$\overset{(iii.)}{\leqslant} \frac{1}{r} \cdot \left(1 + \frac{\Delta}{c^2}\right) \sqrt{1 - (1-\cos\theta) C_{\mathrm{WE}}} \left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) \tag{D.56}$$

where $(i.)$ follows from the necessary condition in Proposition D.3; $(ii.)$ uses the condition on β; and $(iii.)$ replaces $2w_1 w_2 f_1^{y_0/y_i}(\boldsymbol{x}_0) f_2^{y_0/y_i}(\boldsymbol{x}_0)$ leveraging C_{WE}. Now, we define

$$K := \frac{1-\Delta}{1+\Delta}\left(1 - C_{\mathrm{WE}}(1-\cos\theta)\right)^{-1/2}. \tag{D.57}$$

All we need to do is to prove that $\mathcal{M}_{\mathrm{WE}}$ is robust within radius Kr. To do so, from Equation (4.4), we upper bound $\|w_1 \nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 \nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0)\|_2$ by

$$\frac{1}{Kr}\left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) - \beta K r (w_1 + w_2) : \tag{D.58}$$

$$\begin{aligned}
&\|w_1 \nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 \nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0)\|_2 \\
\leqslant &\frac{1}{r} \cdot \left(1 + \frac{\Delta}{c^2}\right) \sqrt{1 - (1-\cos\theta) C_{\mathrm{WE}}} \left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) && \text{(D.59)} \\
\leqslant &\frac{1}{r}(1+\Delta)\sqrt{1 - (1-\cos\theta) C_{\mathrm{WE}}} \left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) && \text{(D.60)} \\
= &\frac{1}{r} \cdot \frac{1-\Delta}{\frac{1-\Delta}{1+\Delta}\left(1 - (1-\cos\theta) C_{\mathrm{WE}}\right)^{-1/2}} \left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) && \text{(D.61)} \\
= &\frac{1}{Kr}(1-\Delta)\left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) && \text{(D.62)} \\
\leqslant &\frac{1}{Kr}\left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0) - \Delta \min\{f_1^{y_0/y_i}(\boldsymbol{x}_0), f_2^{y_0/y_i}(\boldsymbol{x}_0)\}(w_1 + w_2)\right). && \text{(D.63)}
\end{aligned}$$

Notice that $\Delta \min\{f_1^{y_0/y_i}(\boldsymbol{x}_0), f_2^{y_0/y_i}(\boldsymbol{x}_0)\} \geqslant \beta c^2 r^2$ from β's condition, so

$$\|w_1 \nabla_{\boldsymbol{x}} f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 \nabla_{\boldsymbol{x}} f_2^{y_0/y_i}(\boldsymbol{x}_0)\|_2$$

$$\leqslant \frac{1}{Kr}\left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0) - \beta c^2 r^2 (w_1 + w_2)\right) \tag{D.64}$$

$$= \frac{1}{Kr}\left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) - \beta K r (w_1 + w_2) \cdot \frac{c^2}{K^2} \tag{D.65}$$

$$\leqslant \frac{1}{Kr}\left(w_1 f_1^{y_0/y_i}(\boldsymbol{x}_0) + w_2 f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) - \beta K r (w_1 + w_2). \tag{D.66}$$

From Equation (4.4), the theorem for Weighted Ensemble is proved.

Now we prove the theorem for Max-Margin Ensemble. Similarly, for any arbitrary y_1, y_2 such that $y_1 \neq y_0, y_2 \neq y_0$, we have

$$\begin{aligned} &\|\nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \\ \leqslant &\frac{1}{r} \cdot \left(1 + \frac{\Delta}{c^2}\right) \sqrt{1 - (1 - \cos\theta) C_{\mathrm{MME}}} \left(f_1^{y_0/y_1}(\boldsymbol{x}_0) + f_2^{y_0/y_2}(\boldsymbol{x}_0)\right). \end{aligned} \tag{D.67}$$

Now we define

$$K' := \frac{1 - \Delta}{1 + \Delta}\left(1 - C_{\mathrm{MME}}(1 - \cos\theta)\right)^{-1/2}. \tag{D.68}$$

Again, from β's condition we have $\Delta \min\{f_1^{y_0/y_1}(\boldsymbol{x}_0), f_2^{y_0/y_2}(\boldsymbol{x}_0)\} \geqslant \beta c^2 r^2$ and

$$\|\nabla_{\boldsymbol{x}} f_1^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_2^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{K'r}\left(f_1^{y_0/y_i}(\boldsymbol{x}_0) + f_2^{y_0/y_i}(\boldsymbol{x}_0)\right) - 2\beta K' r. \tag{D.69}$$

From Equation (D.1), the ensemble is $(K'r)$-robust at point $\boldsymbol{x}_0$, i.e., the theorem for Max-Margin Ensemble is proved. QED.

D.1.4 Proofs of Model-Smoothness Bounds for Randomized Smoothing

Theorem 4.2 (Model-Smoothness Upper Bound for $\bar{g}_f^\epsilon$)**.** Let $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ be a Gaussian random variable, then the soft smoothed confidence function $\bar{g}_f^\epsilon$ is $(2/\sigma^2)$-smooth.

Proof of Theorem 4.2. Recall that $\bar{g}_f^\epsilon(\boldsymbol{x})_j = \mathbb{E}_{\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)} f(\boldsymbol{x} + \epsilon)_j$, where $f(\boldsymbol{x} + \epsilon)_j$ is a function from $\mathbb{R}^d$ to $\{0, 1\}$. Therefore, to prove that $g_{\mathcal{M}}^\epsilon$ is $(2/\sigma^2)$-smooth, we only need to show that for any function $f : \mathbb{R}^d \to [0, 1]$, the function $\bar{f} := f * \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ is $(2/\sigma^2)$-smooth.

According to [317, Lemma 1], we have

$$\bar{f}(\boldsymbol{x}) = \frac{1}{(2\pi\sigma^2)^{d/2}} \int_{\mathbb{R}^d} f(\boldsymbol{t}) \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{t}\|_2^2}{2\sigma^2}\right) \mathrm{d}\boldsymbol{t}, \tag{D.70}$$

$$\nabla \bar{f}(\boldsymbol{x}) = \frac{1}{(2\pi\sigma^2)^{d/2}\sigma^2} \int_{\mathbb{R}^d} f(\boldsymbol{t})(\boldsymbol{x} - \boldsymbol{t}) \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{t}\|_2^2}{2\sigma^2}\right) \mathrm{d}\boldsymbol{t}. \tag{D.71}$$

To show $\bar{f}$ is $(2/\sigma^2)$-smooth, we only need to show that $\nabla \bar{f}$ is $(2/\sigma^2)$-Lipschitz. Let $\boldsymbol{H}_{\bar{f}}(\boldsymbol{x})$ be the Hessian matrix of $\bar{f}$. Thus, we only need to show that for any unit vector $\boldsymbol{u}$, $|\boldsymbol{u}^\intercal \boldsymbol{H}_{\bar{f}}(\boldsymbol{x})\boldsymbol{u}| \leqslant 2/\sigma^2$. By the isotropy of $\boldsymbol{H}_{\bar{f}}(\boldsymbol{x})$, it is sufficient to consider $\boldsymbol{u} = (1, 0, 0, \ldots, 0)^\intercal$, where $\boldsymbol{u}^\intercal \boldsymbol{H}_{\bar{f}}(\boldsymbol{x})\boldsymbol{u} = \boldsymbol{H}_{\bar{f}}(\boldsymbol{x})_{11}$. Now we only need to bound the absolute value of $\boldsymbol{H}_{\bar{f}}(\boldsymbol{x})_{11}$:

$$|\boldsymbol{H}_{\bar{f}}(\boldsymbol{x})_{11}| = \Big|\frac{1}{(2\pi\sigma^2)^{d/2}\sigma^2}\int_{\mathbb{R}^d} f(\boldsymbol{t}) \cdot \frac{\partial}{\partial \boldsymbol{x}_1}(\boldsymbol{x}-\boldsymbol{t}) \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{t}\|_2^2}{2\sigma^2}\right) \mathrm{d}\boldsymbol{t}\Big| \tag{D.72}$$

$$= \frac{1}{(2\pi\sigma^2)^{d/2}\sigma^2}\Big|\int_{\mathbb{R}^d} f(\boldsymbol{t}) \cdot \left(1 - \frac{(\boldsymbol{x}_1-\boldsymbol{t}_1)^2}{\sigma^2}\right) \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{t}\|_2^2}{2\sigma^2}\right) \mathrm{d}\boldsymbol{t}\Big| \tag{D.73}$$

$$\leqslant \frac{1}{(2\pi\sigma^2)^{d/2}\sigma^2}\Big|\int_{\mathbb{R}^d} \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{t}\|_2^2}{2\sigma^2}\right) \mathrm{d}\boldsymbol{t}\Big|$$
$$+ \frac{1}{(2\pi\sigma^2)^{d/2}\sigma^2}\Big|\int_{\mathbb{R}^d} \frac{(\boldsymbol{x}_1-\boldsymbol{t}_1)^2}{\sigma^2} \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{t}\|_2^2}{2\sigma^2}\right) \mathrm{d}\boldsymbol{t}\Big| \tag{D.74}$$

$$= \frac{1}{\sigma^2} + \frac{1}{\sqrt{2\pi\sigma^2}\sigma^2} \cdot 2\int_0^\infty \frac{x^2}{\sigma^2} \exp\left(-\frac{x^2}{2\sigma^2}\right) \mathrm{d}x \tag{D.75}$$

$$= \frac{1}{\sigma^2} + \sqrt{\frac{2}{\pi}} \cdot \frac{1}{\sigma^2}\int_0^\infty t^2 \exp(-t^2/2)\, \mathrm{d}t. \tag{D.76}$$

Let $\Gamma(\cdot)$ be the Gamma function, we note that

$$\int_0^\infty t^2 \exp(-t^2/2)\, \mathrm{d}t = \int_0^\infty t \exp(-t^2/2)\, \mathrm{d}(-t^2/2) = \sqrt{2}\int_0^\infty \sqrt{t}\exp(-t)\, \mathrm{d}t = \sqrt{2}\Gamma(3/2) = \sqrt{\pi/2}, \tag{D.77}$$

and thus

$$|\boldsymbol{H}_{\bar{f}}(\boldsymbol{x})_{11}| \leqslant \frac{1}{\sigma^2} + \sqrt{\frac{2}{\pi}} \cdot \frac{1}{\sigma^2} \cdot \sqrt{\frac{\pi}{2}} = \frac{2}{\sigma^2}, \tag{D.78}$$

which concludes the proof. QED.

Remark D.2. The model-smoothness upper bound Theorem 4.2 is not limited to the ensemble model with *Ensemble-before-Smoothing* strategy. Indeed, for arbitrary classification models, since the confidence score is in range [0, 1], the theorem still holds. If the confidence score is bounded in $[a, b]$, simple scaling yields the model-smoothness upper bound $\beta = \frac{2(b-a)}{\sigma^2}$.

Proposition D.4 (Model-Smoothness Lower Bound for $\bar{g}_f^\epsilon$)**.** There exists a smoothed confidence function $\bar{g}_f^\epsilon$ that is β-smooth if and only if $\beta \geqslant \left(\frac{1}{\sqrt{2\pi e}\sigma^2}\right)$.

Proof of Proposition D.4. We prove by construction. Consider the single dimensional input space $\mathbb{R}$, and a model f that has confidence 1 if and only if input $x \geqslant 0$. As a result,

$$g_f^\epsilon(x)_{y_0} = \frac{1}{\sqrt{2\pi}\sigma} \int_0^{+\infty} \exp\left(-\frac{(t-x)^2}{2\sigma^2}\right) \mathrm{d}t = \frac{1}{\sqrt{2\pi}\sigma} \int_{-x}^{+\infty} \exp\left(-\frac{t^2}{2\sigma^2}\right) \mathrm{d}t. \tag{D.79}$$

Thus,

$$\frac{\mathrm{d}g_f^\epsilon(x)_{y_0}}{\mathrm{d}x} = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right) \quad \text{and} \quad \left|\frac{\mathrm{d}g_f^\epsilon(x)_{y_0}^2}{\mathrm{d}^2 x}\right| = \frac{1}{\sqrt{2\pi}\sigma^2} \cdot \left|\frac{x}{\sigma}\right| \exp\left(-\frac{x^2}{2\sigma^2}\right). \tag{D.80}$$

By symmetry, we study the function $h(x) = x\exp(-x^2/2)$ for $x \geqslant 0$. We have $h'(x) = (1-x)\exp(-x^2/2)$. Thus, $h(x)$ obtains its maximum at $x_0 = 1$: $h(x_0) = \exp(-1/2)$, which implies that

$$\max\left|\frac{\mathrm{d}g_f^\epsilon(x)_{y_0}^2}{\mathrm{d}^2 x}\right| = \frac{\exp(-1/2)}{\sqrt{2\pi}\sigma^2} = \frac{1}{\sqrt{2\pi e}\sigma^2} \tag{D.81}$$

which implies $\beta \geqslant \dfrac{1}{\sqrt{2\pi e}\sigma^2}$ for this $\bar{g}_f^\epsilon$ per smoothness definition (Definition 4.4). QED.

D.1.5 Proofs of Robustness Conditions for Smoothed Ensemble Models

Corollary D.2 (Gradient and Confidence Margin Conditions for Smoothed WE Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Let $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ be a Gaussian random variable. Define soft smoothed confidence $\bar{g}_i^\epsilon(\boldsymbol{x}) := \mathbb{E}_\epsilon f_i(\boldsymbol{x}+\epsilon)$ for each base model F_i $(1 \leqslant i \leqslant N)$. The $\bar{G}^\epsilon_{\mathcal{M}_{\mathrm{WE}}}$ is a WE defined over soft smoothed base models $\{\bar{g}_i^\epsilon\}_{i=1}^N$ with weights $\{w_i\}_{i=1}^N$. $\bar{G}^\epsilon_{\mathcal{M}_{\mathrm{WE}}}(\boldsymbol{x}_0) = y_0$.

- (Sufficient Condition) The $\bar{G}^\epsilon_{\mathcal{M}_{\mathrm{WE}}}$ is r-robust at point $\boldsymbol{x}_0$ if for any $y_i \neq y_0$,

$$\Big\| \sum_{j=1}^N w_j \nabla_{\boldsymbol{x}} (\bar{g}_j^\epsilon)^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^N w_j (\bar{g}_j^\epsilon)^{y_0/y_i}(\boldsymbol{x}_0) - \frac{2r}{\sigma^2} \sum_{j=1}^N w_j, \tag{D.82}$$

- (Necessary Condition) If $\bar{G}^\epsilon_{\mathcal{M}_{\mathrm{WE}}}$ is r-robust at point $\boldsymbol{x}_0$, for any $y_i \neq y_0$,

$$\Big\| \sum_{j=1}^N w_j \nabla_{\boldsymbol{x}} (\bar{g}_j^\epsilon)^{y_0/y_i}(\boldsymbol{x}_0) \Big\|_2 \leqslant \frac{1}{r} \sum_{j=1}^N w_j (\bar{g}_j^\epsilon)^{y_0/y_i}(\boldsymbol{x}_0) + \frac{2r}{\sigma^2} \sum_{j=1}^N w_j. \tag{D.83}$$

Proof of Corollary D.2. Since $\bar{G}^\epsilon_{\mathcal{M}_{\mathrm{WE}}}$ is a WE defined over $\{\bar{g}_i^\epsilon\}_{i=1}^N$, we apply Theorem 4.1

directly for $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{WE}}}$. Notice that each $\bar{g}^{\epsilon}_i$ has model-smoothness bound $\beta = 2/\sigma^2$ from Theorem 4.2 and the corollary statement follows. QED.

Corollary D.3 (Gradient and Confidence Margin Condition for Smoothed MME Robustness)**.** Given input $\boldsymbol{x}_0 \in \mathbb{R}^d$ with ground-truth label $y_0 \in [C]$. Let $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ be a Gaussian random variable. Define soft smoothed confidence $\bar{g}^{\epsilon}_i(\boldsymbol{x}) := \mathbb{E}_{\epsilon} f_i(\boldsymbol{x} + \epsilon)$ for either base model F_1 or F_2. The $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ is a MME defined over soft smoothed base models $\{\bar{g}^{\epsilon}_1, \bar{g}^{\epsilon}_2\}$. $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}(\boldsymbol{x}_0) = y_0$.

- (Sufficient Condition) If for any $y_1, y_2 \in [C]$ such that $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}}(\bar{g}^{\epsilon}_1)^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}}(\bar{g}^{\epsilon}_2)^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r}((\bar{g}^{\epsilon}_1)^{y_0/y_1}(\boldsymbol{x}_0) + (\bar{g}^{\epsilon}_2)^{y_0/y_2}(\boldsymbol{x}_0)) - \frac{4r}{\sigma^2}, \quad \text{(D.84)}$$

  then $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ is r-robust at point $\boldsymbol{x}_0$.

- (Necessary Condition) Suppose for any $\boldsymbol{x} \in \{\boldsymbol{x}_0 + \boldsymbol{\delta} : ||\boldsymbol{\delta}||_2 \leqslant r\}$, for any $i \in \{1, 2\}$, either $G_{F_i}(\boldsymbol{x}) = y_0$ or $G^{(2)}_{F_i}(\boldsymbol{x}) = y_0$. If $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ is r-robust at point $\boldsymbol{x}_0$, then for any $y_1, y_2 \in [C]$ such that $y_1 \neq y_0$ and $y_2 \neq y_0$,

$$\|\nabla_{\boldsymbol{x}}(\bar{g}^{\epsilon}_1)^{y_0/y_1}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}}(\bar{g}^{\epsilon}_2)^{y_0/y_2}(\boldsymbol{x}_0)\|_2 \leqslant \frac{1}{r}((\bar{g}^{\epsilon}_1)^{y_0/y_1}(\boldsymbol{x}_0) + (\bar{g}^{\epsilon}_2)^{y_0/y_2}(\boldsymbol{x}_0)) + \frac{4r}{\sigma^2}. \quad \text{(D.85)}$$

Proof of Corollary D.3. Since $\bar{G}^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ is constructed over confidences $\bar{g}^{\epsilon}_1$ and $\bar{g}^{\epsilon}_2$, we can directly apply Theorem 4.1. Again, with the model-smoothness bound $\beta = 2/\sigma^2$ we can easily derive the corollary statement. QED.

D.2 ANALYSIS OF ENSEMBLE SMOOTHING STRATEGIES

In Section 4.1.3, we use the adapted randomized model smoothing strategy which is named *Ensemble-before-Smoothing* (EBS). We also consider *Ensemble-after-Smoothing* (*Ensemble-after-Smoothing*). Through the following analysis, we will show *Ensemble-before-Smoothing* generally provides higher certified robust radius than *Ensemble-after-Smoothing* which justifies our choice of the strategy.

The *Ensemble-before-Smoothing* strategy is defined in Definition 4.5. The *Ensemble-after-Smoothing* strategy is defined as such.

Definition D.1 (***Ensemble-after-Smoothing*** **(EAS))**. Let $\mathcal{M}$ be an ensemble model over base models $\{F_i\}_{i=1}^N$. Let ϵ be a random variable. The EAS ensemble $H^{\epsilon}_{\mathcal{M}} : \mathbb{R}^d \mapsto [C]$

at input $\boldsymbol{x}_0 \in \mathbb{R}^d$ is defined as:

$$H^{\epsilon}_{\mathcal{M}}(\boldsymbol{x}_0) := G^{\epsilon}_{F_c}(\boldsymbol{x}_0) \quad \text{where} \quad c = \arg\max_{i \in [N]} g^{\epsilon}_{F_i}(\boldsymbol{x}_0)_{G^{\epsilon}_{F_i}(\boldsymbol{x}_0)}. \tag{D.86}$$

Here, c is the index of the smoothed base model selected.

Remark D.3. In EBS, we first construct a model ensemble $\mathcal{M}$ based on base models using WE or MME protocol, then apply randomized smoothing on top of the ensemble. The resulting smoothed ensemble predicts the most frequent class of $\mathcal{M}$ when the input follows distribution $\boldsymbol{x}_0 + \epsilon$.

In EAS, we use ϵ to construct smoothed classifiers for base models respectively. Then, for given input $\boldsymbol{x}_0$, the ensemble agrees on the base model which has the highest probability for its predicted class.

D.2.1 Certified Robustness

In this subsection, we characterize the certified robustness when using both strategies.

Ensemble-before-Smoothing

The following theorem gives an explicit method (first compute $g^{\epsilon}_{\mathcal{M}}(\boldsymbol{x}_0)_{G^{\epsilon}_{\mathcal{M}}(\boldsymbol{x}_0)}$ via sampling then compute r) to compute the certified robust radius r for EBS protocol. This method is used for computing the certified robust radius in our paper. All other baselines appeared in our paper also use this method.

Proposition D.5 (Certified Robustness for *Ensemble-before-Smoothing*)**.** Let $G^{\epsilon}_{\mathcal{M}}$ be an ensemble constructed by EBS strategy. The random variable $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$. Then the ensemble $G^{\epsilon}_{\mathcal{M}}$ is r-robust at point $\boldsymbol{x}_0$ where

$$r := \sigma \Phi^{-1}\left(g^{\epsilon}_{\mathcal{M}}(\boldsymbol{x}_0)_{G^{\epsilon}_{\mathcal{M}}(\boldsymbol{x}_0)}\right). \tag{D.87}$$

Here, $g^{\epsilon}_{\mathcal{M}}(\boldsymbol{x}_0)_j = \Pr_{\epsilon}(\mathcal{M}(\boldsymbol{x}_0 + \epsilon) = j)$.

The proposition is a direct application of Proposition 2.1.

Ensemble-after-Smoothing

The following theorem gives an explicit method to compute the certified robust radius r for EAS protocol.

Theorem D.3 (Certified robustness for *Ensemble-after-Smoothing*)**.** Let $H^{\epsilon}_{\mathcal{M}}$ be an ensemble constructed by EAS strategy over base models $\{F_i\}_{i=1}^N$. The random variable $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$. Let $y_0 = H^{\epsilon}_{\mathcal{M}}(\boldsymbol{x}_0)$. For each $i \in [N]$, define

$$r_i := \begin{cases} \sigma\Phi^{-1}\left(g^{\epsilon}_{F_i}(\boldsymbol{x}_0)_{G^{\epsilon}_{F_i}(\boldsymbol{x}_0)}\right), & \text{if } G^{\epsilon}_{F_i}(\boldsymbol{x}_0) = y_0 \\ -\sigma\Phi^{-1}\left(g^{\epsilon}_{F_i}(\boldsymbol{x}_0)_{G^{\epsilon}_{F_i}(\boldsymbol{x}_0)}\right). & \text{if } G^{\epsilon}_{F_i}(\boldsymbol{x}_0) \neq y_0 \end{cases} \tag{D.88}$$

Then the ensemble $H^{\epsilon}_{\mathcal{M}}$ is r-robust at point $\boldsymbol{x}_0$ where

$$r := \frac{\max_{i\in[N]} r_i + \min_{i\in[N]} r_i}{2}. \tag{D.89}$$

Remark D.4. The theorem appears to be a bit counter-intuitive — picking the best smoothed model in terms of certified robustness cannot give strong certified robustness for the ensemble. As long as the base models have different certified robust radius (i.e., r_i's are different), the r, certified robust radius for the ensemble, is strictly inferior to that of the best base model (i.e., $\max r_i$). Furthermore, if there exists a base model with wrong prediction (i.e., $r_i \leqslant 0$), the certified robust radius r is strictly smaller than *half* of the best base model.

Proof of Theorem D.3. Without loss of generality, we assume $r_1 > r_2 > \cdots > r_N$. Let the perturbation added to $\boldsymbol{x}_0$ has ℓ_2 length δ.

When $\delta \leqslant r_N$, since picking any model always gives the right prediction, the ensemble is robust.

When $r_N < \delta \leqslant \frac{r_1+r_N}{2}$, the highest robust radius with wrong prediction is $\delta - r_N$, and we can still guarantee that model F_1 has robust radius at least $r_1 - \delta$ from the smoothness of function $\boldsymbol{x} \mapsto g^{\epsilon}_{F_1}(\boldsymbol{x})_{G^{\epsilon}_{F_1}(\boldsymbol{x}_0)}$ [317]. Since $r_1 - \delta \geqslant \frac{r_1-r_N}{2} \geqslant \delta - r_N$, the ensemble will agree on F_1 or other base model with correct prediction and still gives the right prediction.

When $\delta > \frac{r_1+r_N}{2}$, suppose f_N is a linear model and only predicts two labels (which achieves the tight robust radius bound according to [77]), then f_N can have robust radius $\delta - r_N$ for the wrong prediction. At the same time, for any other model F_i which is linear and predicts correctly, the robust radius is at most $r_i - \delta$. Since $r_i - \delta < r_1 - \delta < \frac{r_1-r_N}{2} < \delta - r_N$, the ensemble can probably give wrong prediction.

In summary, as we have shown, the certified robust radius can be at most r. For any radius $\delta > r$, there exist base models which lead the ensemble $H^{\epsilon}_{\mathcal{M}}(\boldsymbol{x}_0 + \delta\boldsymbol{e})$ to predict the label other than y_0. QED.

D.2.2 Comparison of Two Strategies

In this subsection, we compare the two ensemble strategies when the ensembles are constructed from two base models.

Corollary D.4 (Smoothing Strategy Comparison)**.** Given $\mathcal{M}_{\text{MME}}$, a Max-Margin Ensemble constructed from base models $\{f_a, f_b\}$. Let $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$. Let $G^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ be the EBS ensemble, and $H^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ be the EAS ensemble. Suppose at point $\boldsymbol{x}_0$ with ground-truth label y_0, $G^{\epsilon}_{F_a}(\boldsymbol{x}_0) = G^{\epsilon}_{F_b}(\boldsymbol{x}_0) = y_0$, $g^{\epsilon}_{F_a}(\boldsymbol{x}_0) > 0.5$, $g^{\epsilon}_{F_b}(\boldsymbol{x}_0) > 0.5$.

Let δ be their probability difference for class y_0, i.e, $\delta := |g^{\epsilon}_{F_a}(\boldsymbol{x}_0)_{y_0} - g^{\epsilon}_{F_b}(\boldsymbol{x}_0)_{y_0}|$,. Let $p_{\min}$ be the smaller probability for class y_0 between them, i.e., $p_{\min} := \min\{g^{\epsilon}_{F_a}(\boldsymbol{x}_0)_{y_0}, g^{\epsilon}_{F_b}(\boldsymbol{x}_0)_{y_0}\}$. We denote p to the probability of choosing the correct class when the base models disagree with each other; denote p_{ab} to the probability of both base models agreeing on the correct class:

$$p := \Pr_{\epsilon}\left(\mathcal{M}_{\text{MME}}(\boldsymbol{x}_0 + \epsilon) = y_0 \,|\, F_a(\boldsymbol{x}_0 + \epsilon) \neq F_b(\boldsymbol{x}_0 + \epsilon) \,\text{and}\, (F_a(\boldsymbol{x}_0 + \epsilon) = y_0 \,\text{or}\, F_b(\boldsymbol{x}_0 + \epsilon) = y_0)\right), \tag{D.90}$$

$$p_{ab} := \Pr_{\epsilon}\left(F_a(\boldsymbol{x}_0 + \epsilon) = F_b(\boldsymbol{x}_0 + \epsilon) = y_0\right). \tag{D.91}$$

We have:

1. If $p > 1/2 + (2 + 4(p_{\min} - p_{ab})/\delta)^{-1}$, $r_G > r_H$.

2. If $p \leqslant 1/2$, $r_H \geqslant r_G$.

Here, r_G is the certified robust radius of $G^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ computed from Equation (D.87); and r_H is the certified robust radius of $H^{\epsilon}_{\mathcal{M}_{\text{MME}}}$ computed from Equation (D.89).

Remark D.5. Since p is the probability where the ensemble chooses the correct prediction between two base model predictions, with Max-Margin Ensemble, we think $p > 1/2$ with non-trivial margin.

The quantity $p_{\min} - p_{ab}$ and δ both measure the base model's diversity in terms of predicted label distribution, and generally they should be close. As a result, $1/2 + (2 + 4(p_{\min} - p_{ab})/\delta)^{-1} \approx 1/2 + 1/6 = 2/3$, and case (1) should be much more likely to happen than case (2). Therefore, *EBS usually yields higher robustness guarantee.* We remark that the similar tendency also holds with multiple base models.

Proof of Corollary D.4. For convenience, define $p_a := g^{\epsilon}_{F_a}(\boldsymbol{x}_0)_{y_0}, p_b := g^{\epsilon}_{F_b}(\boldsymbol{x}_0)_{y_0}$, where $p_a = p_b + \delta$ and $p_{\min} = p_b$.

From Proposition D.5 and Theorem D.3, we have

$$r_G := \frac{\sigma}{2} \cdot 2\Phi^{-1}\left(\Pr_{\epsilon}(\mathcal{M}_{\mathrm{MME}}(\boldsymbol{x}_0+\epsilon)=y_0)\right), \quad r_H := \frac{\sigma}{2}\left(\Phi^{-1}(p_a)+\Phi^{-1}(p_b)\right). \tag{D.92}$$

Notice that $\Pr_\epsilon(\mathcal{M}_{\mathrm{MME}}(\boldsymbol{x}_0+\epsilon)=y_0) = p_{ab}+p(p_a+p_b-2p_{ab})$, we can rewrite r_G as

$$r_G = \frac{\sigma}{2}\cdot 2\Phi^{-1}(p_{ab}+p(p_a+p_b-2p_{ab})). \tag{D.93}$$

1. When $p > 1/2 + (2+4(p_{\min}-p_{ab})/\delta)^{-1}$,
   since

$$p > \frac{1}{2}+\frac{1}{2+\frac{4(p_{\min}-p_{ab})}{\delta}} = \frac{1}{2}+\frac{\delta}{2\delta+4(p_b-p_{ab})} = \frac{p_a+p_b+\delta-2p_{ab}}{2(p_a+p_b-2p_{ab})} = \frac{p_a-p_{ab}}{p_a+p_b-2p_{ab}}, \tag{D.94}$$

   we have $p_{ab}+p(p_a+p_b-2p_{ab}) > p_a$. Therefore, $r_G > \sigma\Phi^{-1}(p_a)$. Whereas, $r_H \leqslant \sigma/2 \cdot 2\Phi^{-1}(p_a) = \sigma\Phi^{-1}(p_a)$. So $r_G > r_H$.

2. When $p \leqslant 1/2$,

$$p_{ab}+p(p_a+p_b-2p_{ab}) \leqslant p_{ab}+1/2\cdot(p_a+p_b-2p_{ab}) = (p_a+p_b)/2. \tag{D.95}$$

   Therefore, $r_G \leqslant \sigma\Phi^{-1}((p_a+p_b)/2)$. Notice that Φ^{-1} is convex in $[1/2,+\infty)$, so $\Phi^{-1}(p_a)+\Phi^{-1}(p_b) \geqslant 2\Phi^{-1}((p_a+p_b)/2)$, i.e., $r_H \geqslant r_G$.

QED.

D.3 ANALYSIS OF ALTERNATIVE DESIGN OF DRT

In Section 4.2, we design our DRT based on GD Loss

$$\mathcal{L}_{\mathrm{GD}}(\boldsymbol{x}_0)_{ij} = \left\|\nabla_{\boldsymbol{x}} f_i^{y_0/y_i^{(2)}}(\boldsymbol{x}_0) + \nabla_{\boldsymbol{x}} f_j^{y_0/y_j^{(2)}}(\boldsymbol{x}_0)\right\|_2 \tag{D.96}$$

and CM Loss

$$\mathcal{L}_{\mathrm{CM}}(\boldsymbol{x}_0)_{ij} = f_i^{y_i^{(2)}/y_0}(\boldsymbol{x}_0) + f_j^{y_j^{(2)}/y_0}(\boldsymbol{x}_0). \tag{D.97}$$

Following the convention, we apply Gaussian augmentation to train the models, i.e., replacing $\boldsymbol{x}_0$ by $\boldsymbol{x}_0+\epsilon$ where $\epsilon \sim \mathcal{N}(0,\sigma^2\boldsymbol{I}_d)$ in Equation (4.9) and Equation (4.10). We apply these two regularizers to every valid base model pair (F_i, F_j), where the valid pair means both base models predict the ground truth label y_0: $F_i(\boldsymbol{x}_0+\epsilon) = F_j(\boldsymbol{x}_0+\epsilon) = y_0$.

One may concern that in the worst case, there could be $O(N^2)$ valid pairs, i.e., $O(N^2)$ regularization terms in the training loss. However, we should notice that each base model F_i only appears in $O(N)$ valid pairs. Therefore, when N is large, we can optimize DRT by training iteratively, i.e., by regularizing each base model one by one to save the computational cost.

An alternative design inspired from the theorems (e.g., Theorem 4.1) is to use overall summation instead of pairwise summation, which directly correlates with I_{y_i} (Equation (4.6)):

$$\mathcal{L}'_{\text{GD}}(\boldsymbol{x}_0) = \Big\| \sum_{i=1}^{N} \nabla_{\boldsymbol{x}} f_i^{y_0/y_i^{(2)}}(\boldsymbol{x}_0) \Big\|_2, \tag{D.98}$$

$$\mathcal{L}'_{\text{CM}}(\boldsymbol{x}_0) = \sum_{i=1}^{N} f_i^{y_i^{(2)}/y_0}(\boldsymbol{x}_0). \tag{D.99}$$

Although this design appears to be more aligned with the theorem and more efficient with $O(N)$ regularization terms, it also requires all base models F_i to have the same runner-up prediction $y_i^{(2)}$ as observed from both theorem and intuition (otherwise diversified gradients and confidence margins are for different and independent labels that are meaningless to jointly optimize). It is less likely to have all base models having the same runner-up prediction than a pair of base models having the same runner-up prediction especially in the initial training phase. Therefore, this alternative design will cause fewer chances of meaningful optimization than the previous design and we use the previous design for our DRT in practice.

D.4 EXPERIMENT DETAILS

Baselines. We consider the following state-of-the-art baselines for certified robustness: (1) **Gaussian smoothing** [77] trains a smoothed classifier by applying Gaussian augmentation. 2. MACER [437]: Adding the regularization term to maximize the certified radius $R = \frac{\sigma}{2}(p_A - p_B)$ on training instances. (2) **SmoothAdv** [317] combines adversarial training with Gaussian augmentation. (3) **MACER** [437] improves a single model's certified robustness by adding regularization terms to minimize the Negative Log Likelihood (NLL) between smoothed classifier's output $g_F(\boldsymbol{x})$ and label y, and maximize the certified radius $R = \frac{\sigma}{2}(\Phi^{-1}(g_F^{\epsilon}(\boldsymbol{x})_y) - \Phi^{-1}(\max_{y' \neq y} g_F^{\epsilon}(\boldsymbol{x})_{y'}))$, where $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$ and g_F^{ϵ} is as defined in Equation (4.3). (4) **Stability** [218] maintains the stability of the smoothed classifier g_F by minimizing the Rényi Divergence between $g_F(\boldsymbol{x})$ and $g_F(\boldsymbol{x} + \epsilon)$ where $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$. (5) **SWEEN** [234] builds smoothed Weighted Ensemble (WE), which is the only prior work

computing certified robustness for ensemble to our knowledge.

Evaluation Metric. We report the standard *certified accuracy* under different ℓ_2 radii r's as our evaluation metric following Cohen et al. [77], which is defined as the fraction of the test set samples that the smoothed classifier can certify the robustness within the ℓ_2 ball of radius r. Since the computation of the accurate value of this metric is intractable, we report the *approximate certified test accuracy* [77] sampled through the Monte Carlo procedure. For each sample, the robustness certification holds with probability at least $1-\alpha$. Following the literature, we choose $\alpha = 0.001$, $n_0 = 100$ for Monte Carlo sampling during prediction phase, and $n = 10^5$ for Monte Carlo sampling during certification phase. On MNIST and CIFAR-10 we evaluated every 10-th image in the test set, for $1,000$ images total. On ImageNet we evaluated every 100-th image in the validation set, for 500 images total. This evaluation protocol is the same as prior work [77, 317].

D.4.1 MNIST

Baseline Configuration. Following the literature [165, 317, 437], in each batch, each training sample is Gaussian augmented twice (augmenting more times yields negligible difference as Salman et al. [317] show). We choose Gaussian smoothing variance $\sigma \in \{0.25, 0.5, 1.0\}$ for training and evaluation for all methods. For SmoothAdv, we consider the attack to be 10-step ℓ_2 PGD attack with perturbation scale $\delta = 1.0$ without pretraining and unlabelled data augmentation. We reproduced results similar to their paper by using their open-sourced code[21].

Training Details. First, we use LeNet architecture and train each base model for 90 epochs. For the training optimizer, we use the SGD-momentum with the initial learning rate $\alpha = 0.01$. The learning rate is decayed for every 30 epochs with decay ratio $\gamma = 0.1$ and the batch size equals to 256. Then, we apply DRT to finetune our model with small learning rate α for another 90 epochs. We explore different DRT hyper-parameters ρ_1, ρ_2 together with the initial learning rate α, and report the best certified accuracy on each radius r among all the trained ensemble models.

Efficiency Analysis. We regard the *execution time per mini-batch* as our efficiency criterion. For MNIST with batch size equals to 256, DRT with the Gaussian smoothing base model only requires 1.04s to finish one mini-batch training to achieve the comparable results

[21] `https://github.com/Hadisalman/smoothing-adversarial/`

to the SmoothAdv method which requires 1.86 s. Moreover, DRT with the SmoothAdv base model requires 2.52 s per training batch but achieves much better results. The evaluation is on single NVIDIA GeForce GTX 1080 Ti GPU.

D.4.2 CIFAR-10

Baseline Configuration. Following the literature [165, 317, 437], in each batch, each training sample is Gaussian augmented twice (augmenting more times yields negligible difference as Salman et al. [317] show). We choose Gaussian smoothing variance $\sigma \in \{0.25, 0.5, 1.0\}$ for training and evaluation for all methods. For SmoothAdv, we consider the attack to be 10-step ℓ_2 PGD attack with perturbation scale $\delta = 1.0$ without pretraining and unlabelled data augmentation. We also reproduced the similar results mentioned in baseline papers.

Training Details. First, we use ResNet-110 architecture and train each base model for 150 epochs. For the training optimizer, we use the SGD-momentum with the initial learning rate $\alpha = 0.1$. The learning rate is decayed for every 50-epochs with decay ratio $\gamma = 0.1$. Then, we use DRT to finetune our model with small learning rate α for another 150 epochs. We also explore different DRT hyper-parameters ρ_1, ρ_2 together with the initial learning rate α, and report the best certified accuracy on each radius r among all the trained ensemble models.

Efficiency Analysis. We also use the *execution time per mini-batch* as our efficiency criterion. For CIFAR-10 with batch size equals to 256, DRT with the Gaussian smoothing base model requires 3.82 s to finish one mini-batch training to achieve the competitive results to 10-step PGD attack based SmoothAdv method which requires 6.39 s. All the models are trained in parallel on 4 NVIDIA GeForce GTX 1080 Ti GPUs.

D.4.3 ImageNet

For ImageNet, we utilize ResNet-50 architecture and train each base model for 90 epochs using SGD-momentum optimizer. The initial learning rate α is set to 0.1. During training, the learning rate is decayed for every 30-epochs with decay ratio $\gamma = 0.1$. We tried different Gaussian smoothing parameter $\sigma \in \{0.50, 1.00\}$, and consider the best hyper-parameter configuration for each σ. Then, we use DRT to finetune base models with the learning rate $\alpha = 5 \times 10^{-3}$ for another 90 epochs. Due to the consideration of achieving high certified

accuracy on large radii, we choose large DRT training hyper-parameter ρ_1 and ρ_2 in practice, which lead to relatively low benign accuracy.

APPENDIX E: APPENDIX FOR CHAPTER 7

Below is an overview of this appendix.

- **Appendix E.1** contains the proofs for TSS general framework, which is introduced in Section 7.3.
- **Appendix E.2** contains the theorem statements and proofs for the robustness conditions derived for common smoothing distributions. These statements are instantiations of Theorem 7.1, and serve for both certifying resolvable transformations and differentially resolvable transformations.
- **Appendix E.3** contains the proofs of robustness conditions for various resolvable transformations.
- **Appendix E.4** contains the proofs of general theorems for certifying differentially resolvable transformations.
- **Appendix E.5** formally defines the rotation and scaling transformations, two typical differentially resolvable transformations.
- **Appendix E.6** proves our supporting theorems for computing the interpolation bound, which is used when certifying differentially resolvable transformations.
- **Appendix E.7** contains concrete algorithm descriptions for certifying differentially resolvable transformations.
- **Appendix E.8** presents the omitted details in experiments, including experiment settings, detailed discussion of baseline approaches, implementation details, and additional results.

E.1 PROOFS FOR GENERAL CERTIFICATION FRAMEWORK

Here we provide the proof for Theorem 7.1. For that purpose, recall the following definition from the main part of this chapter:

Definition 7.1 (restated)**.** Let $\phi\colon \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ be a transformation, $\varepsilon \sim \mathcal{P}_\varepsilon$ a random variable taking values in $\mathcal{Z}$ and let $h\colon \mathcal{X} \to \mathcal{Y}$ be a base classifier. We define the ε-smoothed

classifier $g\colon \mathcal{X} \to \mathcal{Y}$ as $g(\boldsymbol{x}; \varepsilon) = \arg\max_{y \in \mathcal{Y}} q(y|\, \boldsymbol{x};\, \varepsilon)$ where q is given by the expectation with respect to the smoothing distribution ε, i.e.,

$$q(y|\, \boldsymbol{x};\, \varepsilon) := \mathbb{E}(p(y|\, \phi(\boldsymbol{x},\, \varepsilon))). \tag{E.1}$$

Here, we additionally define the notion of level sets separately. These sets originate from statistical hypothesis testing correspond to rejection regions of likelihood ratio tests.

Definition E.1 (Lower level sets)**.** Let $\varepsilon_0 \sim \mathcal{P}_0$, $\varepsilon_1 \sim \mathcal{P}_1$ be $\mathcal{Z}$-valued random variables with probability density functions f_0 and f_1 with respect to a measure μ. For $t \geqslant 0$ we define lower and strict lower level sets as

$$\begin{gathered} \underline{S_t} := \left\{z \in \mathcal{Z}\colon\ \Lambda(z) < t\right\}, \quad \overline{S_t} := \left\{z \in \mathcal{Z}\colon\ \Lambda(z) \leqslant t\right\}, \\ \text{where} \quad \Lambda(z) := \frac{f_1(z)}{f_0(z)}. \end{gathered} \tag{E.2}$$

We also make the following definition in order to reduce clutter and simplify the notation. This definition will be used throughout the proofs presented here.

Definition E.2 ((p_A, p_B)-Confident Classifier)**.** Let $\boldsymbol{x} \in \mathcal{X}$, $y_A \in \mathcal{Y}$ and p_A, $p_B \in [0,\, 1]$ with $p_A > p_B$. We say that the ε-smoothed classifier q is $(p_A,\, p_B)$-confident at x if

$$q(y_A|\, \boldsymbol{x};\, \varepsilon) \geqslant p_A \geqslant p_B \geqslant \max_{y \neq y_A} q(y|\, \boldsymbol{x};\, \varepsilon). \tag{E.3}$$

E.1.1 Auxiliary Lemmas

Lemma E.1. Let ε_0 and ε_1 be random variables taking values in $\mathcal{Z}$ and with probability density functions f_0 and f_1 with respect to a measure μ. Denote by Λ the likelihood ratio $\Lambda(z) = f_1(z)/f_0(z)$. For $p \in [0,\, 1]$ let $\tau_p := \inf\{t \geqslant 0\colon\ \mathcal{P}_0(\overline{S_t}) \geqslant p\}$. Then, it holds that

$$\mathcal{P}_0\left(\underline{S}_{\tau_p}\right) \leqslant p \leqslant \mathcal{P}_0\left(\overline{S}_{\tau_p}\right). \tag{E.4}$$

Proof. We first show the RHS of inequality (E.4). This follows directly from the definition of τ_p if we show that the function $t \mapsto \mathcal{P}_0\left(\overline{S}_t\right)$ is right-continuous. For that purpose, let $t \geqslant 0$ and let $\{t_n\}_n$ be a sequence in $\mathbb{R}_{\geqslant 0}$ such that $t_n \downarrow t$. Define the sets $A_n := \{z\colon\ \Lambda(z) \leqslant t_n\}$ and note that $A_{n+1} \subseteq A_n$. Clearly, if $z \in \overline{S}_t$, then $\forall\, n\colon\ \Lambda(z) \leqslant t \leqslant t_n$, thus $z \in \cap_n A_n$ and hence $\overline{S}_t \subseteq \cap_n A_n$. If on the other hand $z \in \cap_n A_n$, then $\forall n\colon \Lambda(z) \leqslant t_n \to t$ as $n \to \infty$ and

thus $z \in \overline{S}_t$, yielding $\overline{S}_t = \cap_n A_n$. Hence for any $t \geqslant 0$ we have that

$$\lim_{n\to\infty} \mathcal{P}_0\left(A_n\right) = \mathcal{P}_0\left(\bigcap_n A_n\right) = \mathcal{P}_0\left(\underline{S}_t\right). \tag{E.5}$$

Thus, the function $t \mapsto \mathcal{P}_0\left(\overline{S}_t\right)$ is right continuous and in particular it follows that $\mathcal{P}_0\left(\overline{S}_{\tau_p}\right) \geqslant p$. We now show the LHS of inequality (E.4). Consider the sets $B_n := \{z\colon\ \Lambda(z) < \tau_p - 1/n\}$ and note that $B_n \subseteq B_{n+1}$. Clearly, if $z \in \cup_n B_n$, then $\exists\, n$ such that $\Lambda(z) < \tau_p - 1/n < \tau_p$ and thus $z \in \underline{S}_{\tau_p}$. If on the other hand $z \in \underline{S}_{\tau_p}$, then we can choose n large enough such that $\Lambda(z) < \tau_p - 1/n$ and thus $z \in \cup_n B_n$ yielding $\underline{S}_{\tau_p} = \cup_n B_n$. Furthermore, by the definition of τ_p and since for any $n \in \mathbb{N}$ we have that $\mathcal{P}_0(B_n) = \mathcal{P}_0\left(\underline{S}(\tau_p - 1/n)\right) < p$ it follows that

$$\mathcal{P}_0\left(\underline{S}_{\tau_p}\right) = \mathcal{P}_0\left(\bigcup_n B_n\right) = \lim_{n\to\infty} \mathcal{P}_0\left(B_n\right) \leqslant p \tag{E.6}$$

concluding the proof. QED.

Lemma E.2. Let ε_0 and ε_1 be random variables taking values in $\mathcal{Z}$ and with probability density functions f_0 and f_1 with respect to a measure μ. Let $h\colon\ \mathcal{Z} \to [0,\, 1]$ be a determinstic function. Then, for any $t \geqslant 0$ the following implications hold:

(i) For any $S \subseteq \mathcal{Z}$ with $\underline{S_t} \subseteq S \subseteq \overline{S_t}$ the following implication holds:

$$\mathbb{E}[h(\varepsilon_0)] \geqslant \mathcal{P}_0(S) \Rightarrow \mathbb{E}[h(\varepsilon_1)] \geqslant \mathcal{P}_1(S). \tag{E.7}$$

(i) For any $S \subseteq \mathcal{Z}$ with $\overline{S_t}^c \subseteq S \subseteq \underline{S_t}^c$ the following implication holds:

$$\mathbb{E}[h(\varepsilon_0)] \leqslant \mathcal{P}_0(S) \Rightarrow \mathbb{E}[h(\varepsilon_1)] \leqslant \mathcal{P}_1(S). \tag{E.8}$$

Proof. We first prove (i). For that purpose, consider

$$\begin{aligned}
\mathbb{E}[f(\varepsilon_1)] - \mathcal{P}_1(S) &= \int h f_1\, d\mu - \int_S f_1\, d\mu && \text{(E.9)}\\
&= \int_{S^c} h f_1\, d\mu - \left(\int_S (1-h) f_1\, d\mu\right) && \text{(E.10)}\\
&= \int_{S^c} h\Lambda f_0\, d\mu - \left(\int_S (1-h)\Lambda f_0\, d\mu\right) && \text{(E.11)}\\
&\geqslant t \cdot \int_{S^c} h f_0\, d\mu - t \cdot \left(\int_S (1-h) f_0\, d\mu\right) && \text{(E.12)}
\end{aligned}$$

$$= t \cdot \left(\int h f_0 \, d\mu - \int_S f_0 \, d\mu \right) \tag{E.13}$$

$$= t \cdot \big(\mathbb{E}[f(\varepsilon_0)] - \mathcal{P}_0(S) \big) \geqslant 0. \tag{E.14}$$

The inequality in (E.12) follows from the fact that whenever $z \in S^{\mathrm{c}}$, then $f_1(z) \geqslant t \cdot f_0(z)$ and if $z \in S$, then $f_1(z) \leqslant t \cdot f_0(z)$ since S is a lower level set. Finally, the inequality in (E.14) follows from the assumption. The proof of (ii) is analogous and omitted here. QED.

E.1.2 Proof of Theorem 7.1

Theorem 7.1 (restated)**.** Let $\varepsilon_0 \sim \mathcal{P}_0$ and $\varepsilon_1 \sim \mathcal{P}_1$ be $\mathcal{Z}$-valued random variables with probability density functions f_0 and f_1 with respect to a measure μ on $\mathcal{Z}$ and let $\phi \colon \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ be a semantic transformation. Suppose that $y_A = g(\boldsymbol{x}; \varepsilon_0)$ and let $p_A, p_B \in [0, 1]$ be bounds to the class probabilities, i.e.,

$$q(y_A | \, \boldsymbol{x}, \epsilon_0) \geqslant p_A > p_B \geqslant \max_{y \neq y_A} q(y | \, \boldsymbol{x}, \epsilon_0). \tag{E.15}$$

For $t \geqslant 0$, let $\underline{S}_t, \overline{S}_t \subseteq \mathcal{Z}$ be the sets defined as $\underline{S}_t := \{f_1/f_0 < t\}$ and $\overline{S}_t := \{f_1/f_0 \leqslant t\}$ and define the function $\xi \colon [0, 1] \to [0, 1]$ by

$$\begin{aligned} \xi(p) &:= \sup\{\mathcal{P}_1(S) \colon \underline{S}_{\tau_p} \subseteq S \subseteq \overline{S}_{\tau_p}\} \\ \text{where} \quad \tau_p &:= \inf\{t \geqslant 0 \colon \mathcal{P}_0(\overline{S}_t) \geqslant p\}. \end{aligned} \tag{E.16}$$

If the condition

$$\xi(p_A) + \xi(1 - p_B) > 1 \tag{E.17}$$

is satisfied, then it is guaranteed that $g(\boldsymbol{x}; \varepsilon_1) = g(\boldsymbol{x}; \varepsilon_0)$.

Proof. For ease of notation, let ζ be the function defined by

$$t \mapsto \zeta(t) := \mathcal{P}_0(\overline{S}_t) \tag{E.18}$$

and notice that $\tau_p = \zeta^{-1}(p)$ where ζ^{-1} denotes the generalized inverse of ζ. Furthermore, let $\tau_A := \tau_{p_A}$, $\tau_B := \tau_{1-p_B}$, $\underline{S}_A := \underline{S}(\tau_A)$, $\underline{S}_B := \underline{S}(\tau_B)$, $\overline{S}_A := \overline{S}(\tau_A)$ and $\overline{S}_B := \overline{S}(\tau_B)$. We first show that $q(y_A | \, x, \varepsilon_1)$ is lower bounded by $\xi(\tau_A)$. For that purpose, note that by Lemma E.1 we have that $\zeta(\tau_A) = \mathcal{P}_0(\overline{S}_A) \geqslant p_A \geqslant \mathcal{P}_0(\underline{S}_A)$. Thus, the collection of sets

$$\mathcal{S}_A := \{S \subseteq \mathcal{Z} \colon \underline{S}_A \subseteq S \subseteq \overline{S}_A, \, \mathcal{P}_0(S) \leqslant p_A\} \tag{E.19}$$

is not empty. Pick some $A \in \mathcal{S}_A$ arbitrary and note that, since by assumption $g(\cdot;\, \varepsilon_0)$ is $(p_A,\, p_B)$-confident at x it holds that

$$\mathbb{E}(p(y_A|\, \phi(\boldsymbol{x},\, \varepsilon_0))) = q(y_A|\, \boldsymbol{x};\, \varepsilon_0) \geqslant p_A \geqslant \mathcal{P}_0(A). \tag{E.20}$$

Since $\underline{S}_A \subseteq A \subseteq \overline{S}_A$ we can apply part (i) of Lemma E.2 and obtain the lower bound

$$q(y_A|\, \boldsymbol{x},\, \varepsilon_1) = \mathbb{E}(p(y_A|\, \phi(\boldsymbol{x},\, \varepsilon_1))) \geqslant \mathcal{P}_1(A). \tag{E.21}$$

Since $A \in \mathcal{S}_A$ was arbitrary, we take the sup over all $A \in \mathcal{S}_A$ and obtain

$$q(y_A|\, \boldsymbol{x};\, \varepsilon_1) \geqslant \sup_{A \in \mathcal{S}_A} \mathcal{P}_1(A) = \xi(p_A) \tag{E.22}$$

We now show that for any $y \neq y_A$ the prediction $q(y|\, \boldsymbol{x};\, \varepsilon_1)$ is upper bounded by $1-\xi(1-p_B)$. For that purpose, note that by Lemma E.1 we have that $\zeta(\tau_B) = \mathcal{P}_0(\overline{S}_A) \geqslant 1-p_B \geqslant \mathcal{P}_0(\underline{S}_B)$. Thus, the collection of sets

$$\mathcal{S}_B := \{S \subseteq \mathcal{Z} \colon \underline{S}_B \subseteq S \subseteq \overline{S}_B,\, \mathcal{P}_0(S) \leqslant 1 - p_B\} \tag{E.23}$$

is not empty. Pick some $B \in \mathcal{S}_A$ arbitrary and note that, since by assumption $g(\cdot;\, \varepsilon_0)$ is $(p_A,\, p_B)$-confident at x it holds that

$$\mathbb{E}(p(y|\, \phi(\boldsymbol{x},\, \varepsilon_0))) = q(y|\, \boldsymbol{x};\, \varepsilon_0) \leqslant p_B = 1 - (1 - p_B) \leqslant 1 - \mathcal{P}_0(B). \tag{E.24}$$

Since $\underline{S}_B^c \subseteq B^c \subseteq \overline{S}_B^c$ we can apply part (ii) of Lemma E.2 and obtain the upper bound

$$q(y|\, \boldsymbol{x};\varepsilon_1) = \mathbb{E}(p(y|\, \phi(\boldsymbol{x},\, \varepsilon_1))) \leqslant 1 - \mathcal{P}_1(B). \tag{E.25}$$

Since $B \in \mathcal{S}_B$ was arbitrary, we take the inf over all $B \in \mathcal{S}_B$ and obtain

$$q(y|\, \boldsymbol{x};\varepsilon_1) \leqslant \inf_{B \in \mathcal{S}_B} (1 - \mathcal{P}_1(B)) = 1 - \xi(1 - p_B). \tag{E.26}$$

Combining together (E.26) and (E.22), we find that, whenever

$$\xi(p_A) + \xi(1 - p_B) > 1 \tag{E.27}$$

it is guaranteed that

$$q(y_A|\, \boldsymbol{x};\, \varepsilon_1) > \max_{y \neq y_A} q(y|\, \boldsymbol{x};\, \varepsilon_1) \tag{E.28}$$

what concludes the proof. QED.

E.2 PROOFS FOR CERTIFICATION WITH DIFFERENT SMOOTHING DISTRIBUTIONS

Here, we instantiate Theorem 7.1 with different smoothing distributions and solve the robustness condition (7.6) for the case where the distribution of ε_1 results from shifting the distribution of ε_0, i.e., $\varepsilon_1 = \alpha + \varepsilon_0$. For ease of notation, let $\zeta\colon \mathbb{R}_{\geqslant 0} \to [0,\, 1]$ be the function defined by

$$t \mapsto \zeta(t) := \mathcal{P}_0\left(\overline{S}_t\right) \tag{E.29}$$

where $\mathcal{P}_0$ is the distribution of ε_0 and $\overline{S}_t$ is a lower level set; recall that the definitions of lower level sets is

$$\begin{gathered} \underline{S}_t := \left\{z \in \mathcal{Z}\colon\ \Lambda(z) < t\right\}, \quad \overline{S}_t := \left\{z \in \mathcal{Z}\colon\ \Lambda(z) \leqslant t\right\}, \\ \text{where} \quad \Lambda(z) := \frac{f_1(z)}{f_0(z)}. \end{gathered} \tag{E.30}$$

Note that the generalized inverse of ζ corresponds to τ_p, i.e.,

$$\zeta^{-1}(p) = \inf\{t \geqslant 0 |\, \zeta(t) \geqslant p\} = \tau_p \tag{E.31}$$

and the function ξ is correspondingly given by

$$\xi(p) = \sup\{\mathcal{P}_1(S) |\, \underline{S}(\zeta^{-1}(p)) \subseteq S \subseteq \overline{S}(\zeta^{-1}(p))\} \tag{E.32}$$

E.2.1 Gaussian Smoothing

Corollary E.1. Suppose $\mathcal{Z} = \mathbb{R}^m$, $\Sigma := \operatorname{diag}(\sigma_1^2,\, \ldots,\, \sigma_m^2)$, $\varepsilon_0 \sim \mathcal{N}(0,\, \Sigma)$ and $\varepsilon_1 := \alpha + \varepsilon_0$ for some $\alpha \in \mathbb{R}^m$. Suppose that the ε_0-smoothed classifier g is $(p_A,\, p_B)$-confident at $x \in \mathcal{X}$ for some $y_A \in \mathcal{Y}$. Then, it holds that $q(y_A |\, \boldsymbol{x};\, \varepsilon_1) > \max_{y \neq y_A} q(y |\, \boldsymbol{x};\, \varepsilon_1)$ if α satisfies

$$\sqrt{\sum_{i=1}^{m} \left(\frac{\alpha_i}{\sigma_i}\right)^2} < \frac{1}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right). \tag{E.33}$$

Proof. By Theorem 7.1 we know that if ε_1 satisfies

$$\xi(p_A) + \xi(1 - p_B) > 1, \tag{E.34}$$

then it is guaranteed that

$$q(y_A|\,\boldsymbol{x};\,\varepsilon_1) > \max_{y\neq y_A} q(y|\,\boldsymbol{x};\,\varepsilon_1). \tag{E.35}$$

The proof is thus complete if we show that (E.34) reduces to (E.33). For that purpose, denote by f_0 and f_1 density functions of ε_0 and ε_1, respectively. Let $A := \Sigma^{-1}$ and note that the bilinear form $(z_1,\, z_2) \mapsto z_1^T A z_2 =: \langle z_1,\, z_2\rangle_A$ defines an inner product on $\mathbb{R}^m$. Let $z \in \mathbb{R}^m$ and consider

$$\Lambda(z) = \frac{f_1(z)}{f_0(z)} = \frac{\exp\left(-\frac{1}{2}\langle z-\alpha,\, z-\alpha\rangle_A\right)}{\exp\left(-\frac{1}{2}\langle z,\, z\rangle_A\right)} = \exp\left(\langle z,\, \alpha\rangle_A - \frac{1}{2}\langle \alpha,\, \alpha\rangle_A\right) \tag{E.36}$$

and thus

$$\Lambda(z) \leqslant t \iff \langle z,\, \alpha\rangle_A \leqslant \log(t) + \frac{1}{2}\langle \alpha,\, \alpha\rangle. \tag{E.37}$$

Let $Z \sim \mathcal{N}(0,\, 1)$ and notice that

$$\frac{\langle \varepsilon_0,\, \alpha\rangle_A}{\sqrt{\langle \alpha,\, \alpha\rangle_A}} \overset{d}{=} Z \overset{d}{=} \frac{\langle \varepsilon_1,\, \alpha\rangle_A - \langle \alpha,\, \alpha\rangle_A}{\sqrt{\langle \alpha,\, \alpha\rangle_A}}. \tag{E.38}$$

Let $\partial_t := \overline{S}_t \backslash \underline{S}_t = \{z\colon\, \Lambda(z) = t\}$ and notice that $\mathcal{P}_0\left(\partial_t\right) = \mathcal{P}_1\left(\partial_t\right) = 0$ and $\mathcal{P}_0(\underline{S}_t) = \mathcal{P}_0(\overline{S}_t)$. Similarly, it holds that $\mathcal{P}_1(\underline{S}_t) = \mathcal{P}_1(\overline{S}_t)$. The function $p \mapsto \xi(p)$ is thus given by

$$\xi(p) = \mathcal{P}_1\left(\overline{S}_{\zeta^{-1}(p)}\right). \tag{E.39}$$

We compute ζ as

$$\zeta(t) = \Pr\left(\Lambda(\varepsilon_0) \leqslant t\right) = \Pr\left(\langle \varepsilon_0,\, \alpha\rangle_A \leqslant \log(t) + \frac{1}{2}\langle \alpha,\, \alpha\rangle_A\right) = \Phi\left(\frac{\log(t) + \frac{1}{2}\langle \alpha,\, \alpha\rangle_A}{\sqrt{\langle \alpha,\, \alpha\rangle_A}}\right) \tag{E.40}$$

and for $p \in [0,\, 1]$ its inverse is

$$\zeta^{-1}(p) = \exp\left(\Phi^{-1}(p)\sqrt{\langle \alpha,\, \alpha\rangle_A} - \frac{1}{2}\langle \alpha,\, \alpha\rangle_A\right). \tag{E.41}$$

Thus

$$\Pr\left(\Lambda(\varepsilon_1) \leqslant \zeta^{-1}(p)\right) \tag{E.42}$$

$$= \Pr\left(\frac{\langle \varepsilon_1,\, \alpha\rangle_A - \langle \alpha,\, \alpha\rangle_A}{\sqrt{\langle \alpha,\, \alpha\rangle_A}} \leqslant \frac{\log(\zeta^{-1}(p)) - \frac{1}{2}\langle \alpha,\, \alpha\rangle_A}{\sqrt{\langle \alpha,\, \alpha\rangle_A}}\right) \tag{E.43}$$

$$= \Phi\left(\frac{\left(\Phi^{-1}(p)\sqrt{\langle \alpha,\, \alpha\rangle_A} - \frac{1}{2}\langle \alpha,\, \alpha\rangle_A\right) - \frac{1}{2}\langle \alpha,\, \alpha\rangle_A}{\sqrt{\langle \alpha,\, \alpha\rangle_A}}\right) \tag{E.44}$$

$$= \Phi\left(\Phi^{-1}(p) - \sqrt{\langle \alpha,\, \alpha\rangle_A}\right). \tag{E.45}$$

Finally, algebra shows that

$$\Phi\left(\Phi^{-1}(p_A) - \sqrt{\langle \alpha,\, \alpha\rangle_A}\right) + \Phi\left(\Phi^{-1}(1-p_B) - \sqrt{\langle \alpha,\, \alpha\rangle_A}\right) > 1 \tag{E.46}$$

is equivalent to

$$\sqrt{\sum_{i=1}^{m}\left(\frac{\alpha_i}{\sigma_i}\right)^2} < \frac{1}{2}\left(\Phi^{-1}(p_A) - \Phi^{-1}(p_B)\right) \tag{E.47}$$

what concludes the proof. QED.

E.2.2 Exponential Smoothing

Corollary E.2. Suppose $\mathcal{Z} = \mathbb{R}^m_{\geqslant 0}$, fix some $\lambda > 0$ and let $\varepsilon_{0,i} \overset{\text{iid}}{\sim} \text{Exp}(1/\lambda)$, $\varepsilon_0 := (\varepsilon_{0,1},\, \ldots,\, \varepsilon_{0,m})^T$ and $\varepsilon_1 := \alpha + \varepsilon_0$ for some $\alpha \in \mathbb{R}^m_{\geqslant 0}$. Suppose that the ε_0-smoothed classifier g is $(p_A,\, p_B)$-confident at $\boldsymbol{x} \in \mathcal{X}$ for some $y_A \in \mathcal{Y}$. Then, it holds that $q(y_A|\, \boldsymbol{x};\, \varepsilon_1) > \max_{y \neq y_A} q(y|\, \boldsymbol{x};\, \varepsilon_1)$ if α satisfies

$$\|\alpha\|_1 < -\frac{\log(1 - p_A + p_B)}{\lambda}. \tag{E.48}$$

Proof. By Theorem 7.1 we know that if ε_1 satisfies

$$\xi(p_A) + \xi(1 - p_B) > 1, \tag{E.49}$$

then it is guaranteed that

$$q(y_A|\, \boldsymbol{x};\, \varepsilon_1) > \max_{y \neq y_A} q(y|\, \boldsymbol{x};\, \varepsilon_1). \tag{E.50}$$

The proof is thus complete if we show that (E.49) reduces to (E.48). For that purpose,

denote by f_0 and f_1 density functions of ε_0 and ε_1, respectively, and note that

$$f_1(z) = \begin{cases} \lambda \cdot \exp(-\lambda \|z - \alpha\|_1), & \min_i(z_i - \alpha_i) \geqslant 0, \\ 0, & \text{otherwise}, \end{cases} \tag{E.51}$$

$$f_0(z) = \begin{cases} \lambda \cdot \exp(-\lambda \|z\|_1), & \min_i(z_i) \geqslant 0, \\ 0, & \text{otherwise}, \end{cases} \tag{E.52}$$

and $\forall i$, $z_i - \alpha_i \leqslant z_i$ and hence $f_0(z) = 0 \Rightarrow f_1(z) = 0$. Thus

$$\Lambda(z) = \frac{f_1(z)}{f_0(z)} = \begin{cases} \exp\left(\lambda \cdot \|\alpha\|_1\right) & \min_i(z_i - \alpha_i) \geqslant 0, \\ 0, & \text{otherwise}. \end{cases} \tag{E.53}$$

Let $S_0 := \{z \in \mathbb{R}^m_{\geqslant 0} \colon \ \min_i(z_i - \alpha_i) < 0\}$ and note that due to independence

$$\mathcal{P}_0(S_0) = \Pr\left(\bigcup_{i=1}^m \{\varepsilon_{0,i} < \alpha_i\}\right) \tag{E.54}$$

$$= 1 - \Pr\left(\bigcap_{i=1}^m \{\varepsilon_{0,i} \geqslant \alpha_i\}\right) = 1 - \prod_{i=1}^m \Pr\left(\varepsilon_{0,i} \geqslant \alpha_i\right) \tag{E.55}$$

$$= 1 - \prod_{i=1}^m \Big(1 - \big(1 - \exp\left(-\lambda\, \alpha_i\right)\big)\Big) \tag{E.56}$$

$$= 1 - \exp\left(-\lambda \|\alpha\|_1\right). \tag{E.57}$$

Let $t_\alpha := \exp(\lambda \|\alpha\|_1)$ and compute ζ as

$$\zeta(t) = \Pr\left(\Lambda(\varepsilon_0) \leqslant t\right) \tag{E.58}$$

$$= \Pr\left(\{\min_i(\varepsilon_{0,i} - \alpha_i) \geqslant 0\} \leqslant t \cdot \exp\left(-\lambda \|\alpha\|_1\right)\right) \tag{E.59}$$

$$= \begin{cases} 1 - \exp\left(-\lambda \|\alpha\|_1\right) & t < t_\alpha, \\ 1 & t \geqslant t_\alpha. \end{cases} \tag{E.60}$$

Recall that $\zeta^{-1}(p) := \inf\{t \geqslant 0 \colon \ \zeta(t) \geqslant p\}$ for $p \in [0,\, 1]$ and hence

$$\zeta^{-1}(p) = \begin{cases} 0 & p \leqslant 1 - \exp(-\lambda \|\alpha\|_1), \\ \exp(\lambda \|\alpha\|_1) & p > 1 - \exp(-\lambda \|\alpha\|_1). \end{cases} \tag{E.61}$$

In order to evaluate ξ we compute the lower and strict lower level sets at $t = \zeta^{-1}(p)$. Recall that $\underline{S}_t = \{z \in \mathbb{R}^m_{\geqslant 0} \colon \Lambda(z) < t\}$ and $\overline{S}_t = \{z \in \mathbb{R}^m_{\geqslant 0} \colon \Lambda(z) \leqslant t\}$ and consider

$$\underline{S}_{\zeta^{-1}(p)} = \left(S_0^c \cap \left\{z \in \mathbb{R}^m_{\geqslant 0} \colon \exp(\lambda\|\alpha\|_1) < \zeta^{-1}(p)\right\}\right) \cup \left(S_0 \cap \left\{z \in \mathbb{R}^m_{\geqslant 0} \,|\, 0 < \zeta^{-1}(p)\right\}\right) \tag{E.62}$$

$$= \begin{cases} \varnothing & p \leqslant 1 - \exp(-\lambda\|\alpha\|_1), \\ S_0 & p > 1 - \exp(-\lambda\|\alpha\|_1) \end{cases} \tag{E.63}$$

and

$$\overline{S}_{\zeta^{-1}(p)} = \left(S_0^c \cap \left\{z \in \mathbb{R}^m_{\geqslant 0} \colon \exp(\lambda\|\alpha\|_1) \leqslant \zeta^{-1}(p)\right\}\right) \dot\cup \left(S_0 \cap \left\{z \in \mathbb{R}^m_{\geqslant 0} \colon 0 \leqslant \zeta^{-1}(p)\right\}\right) \tag{E.64}$$

$$= \begin{cases} S_0 & p \leqslant 1 - \exp(-\lambda\|\alpha\|_1), \\ \mathbb{R}^m_+ & p > 1 - \exp(-\lambda\|\alpha\|_1). \end{cases} \tag{E.65}$$

Suppose that $p \leqslant 1 - \exp(-\lambda\|\alpha\|_1)$. Then we have that $\underline{S}_{\zeta^{-1}(p)} = \varnothing$ and $\overline{S}_{\zeta^{-1}(p)} = S_0$ and hence

$$\begin{aligned} &p \leqslant 1 - \exp(-\lambda\|\alpha\|_1) \\ &\quad \Rightarrow \xi(p) = \sup\{\mathcal{P}_1(S) \colon S \subseteq S_0 \wedge \mathcal{P}_0(S) \leqslant p\} = 0. \end{aligned} \tag{E.66}$$

Condition (E.49) can thus be satisfied only if $p_A > 1 - \exp(-\lambda\|\alpha\|_1)$ and $1 - p_B > 1 - \exp(-\lambda\|\alpha\|_1)$. In this case $\underline{S}_{\zeta^{-1}(p)} = S_0$ and $\overline{S}_{\zeta^{-1}(p)} = \mathbb{R}^m_{\geqslant_0}$. For $p \in [0,\, 1]$ let $\mathcal{S}_p = \{S \subseteq \mathbb{R}^m_{\geqslant 0} \colon S_0 \subseteq S \subseteq \mathbb{R}^m_{\geqslant 0},\, \mathcal{P}_0(S) \leqslant p\}$. Then

$$p > 1 - \exp(-\lambda\|\alpha\|_1) \Rightarrow \xi(p) = \sup_{S \in \mathcal{S}_p} \mathcal{P}_1(S). \tag{E.67}$$

We can write any $S \in \mathcal{S}_p$ as the disjoint union $S = S_0 \dot\cup T$ for some $T \subseteq \mathbb{R}^m_{\geqslant 0}$ such that $\mathcal{P}_0(S_0 \dot\cup T) \leqslant p$. Note that $\mathcal{P}_1(S_0) = 0$ and since $S_0 \cap T = \varnothing$ any $z \in T$ satisfies $0 \leqslant \min_i (z_i - \alpha_i) \leqslant \min_i z_i$ and hence $\Lambda(z) = \exp(\lambda\|\alpha\|_1)$. Thus

$$\mathcal{P}_1(S) = \mathcal{P}_1(T) = \int_T f_1(z)\, dz \tag{E.68}$$

$$= \int_T \exp(\lambda\|\alpha\|_1) f_0(z)\, dz = \exp(\lambda\|\alpha\|_1) \cdot \mathcal{P}_0(T). \tag{E.69}$$

Thus, The supremum of the left hand side over all $S \in \mathcal{S}_p$ equals the supremum of the right hand side over all $T \in \{T' \subseteq S_0^c \colon \mathcal{P}_0(T') \leqslant 1 - \mathcal{P}_0(S_0)\}$

$$\sup_{S \in \mathcal{S}_p} \mathcal{P}_1(S) = \exp(\lambda\|\alpha\|_1) \cdot \sup\{\mathcal{P}_1(T') \colon T' \subseteq S_0^c,\, \mathcal{P}_0(T') \leqslant p - \mathcal{P}_0(S_0)\} \tag{E.70}$$

$$= \exp(\lambda\|\alpha\|_1) \cdot (p - \mathcal{P}_0(S_0)). \tag{E.71}$$

Computing ξ at p_A thus yields

$$\xi(p_A) = \sup_{S \in \mathcal{S}_{p_A}} \mathcal{P}_1(S) = \exp(\lambda\|\alpha\|_1) \cdot \left(p_A - \mathcal{P}_0(S_0)\right) \tag{E.72}$$

$$= \exp(\lambda\|\alpha\|_1) \cdot \left(p_A - \left(1 - \exp\left(-\lambda\|\alpha\|_1\right)\right)\right) \tag{E.73}$$

$$= \exp(\lambda\|\alpha\|_1) \cdot \left(p_A + \exp\left(-\lambda\|\alpha\|_1\right) - 1\right) \tag{E.74}$$

where the third equality follows from (E.57). Similarly, computing ξ at $1 - p_B$ yields

$$\xi(1 - p_B) = \sup_{S \in \mathcal{S}_{1-p_B}} \mathcal{P}_1(S) \tag{E.75}$$

$$= \exp(\lambda\|\alpha\|_1) \cdot \left(1 - p_B - \mathcal{P}_0(S_0)\right) \tag{E.76}$$

$$= \exp(\lambda\|\alpha\|_1) \cdot \left(1 - p_B - \left(1 - \exp\left(-\lambda\|\alpha\|_1\right)\right)\right) \tag{E.77}$$

$$= \exp(\lambda\|\alpha\|_1) \cdot \left(-p_B + \exp\left(-\lambda\|\alpha\|_1\right)\right). \tag{E.78}$$

Finally, condition (E.49) is satisfied whenever α satisfies

$$\exp(\lambda\|\alpha\|_1) \cdot \left(p_A + \exp\left(-\lambda\|\alpha\|_1\right) - 1\right) + \exp(\lambda\|\alpha\|_1) \cdot \left(-p_B + \exp\left(-\lambda\|\alpha\|_1\right)\right) > 1 \tag{E.79}$$

$$\iff \exp(-\lambda\|\alpha\|_1) + p_B - \exp(-\lambda\|\alpha\|_1) < p_A + \exp\left(-\lambda\|\alpha\|_1\right) - 1 \tag{E.80}$$

$$\iff 1 - p_A + p_B < \exp\left(-\lambda\|\alpha\|_1\right) \tag{E.81}$$

$$\iff \|\alpha\|_1 < -\frac{\log(1 - p_A + p_B)}{\lambda} \tag{E.82}$$

what completes the proof. QED.

E.2.3 Uniform Smoothing

Corollary E.3. Suppose $\mathcal{Z} = \mathbb{R}^m$, and $\varepsilon_0 \sim \mathcal{U}([a, b]^m)$ for some $a < b$. Set $\varepsilon_1 := \alpha + \varepsilon_0$ for $\alpha \in \mathbb{R}^m$. Suppose that the ε_0-smoothed classifier g is (p_A, p_B)-confident at $x \in \mathcal{X}$ for some $y_A \in \mathcal{Y}$. Then, it holds that $q(y_A|\,\boldsymbol{x};\, \varepsilon_1) > \max_{y \neq y_A} q(y|\,\boldsymbol{x};\, \varepsilon_1)$ if α satisfies

$$1 - \left(\frac{p_A - p_B}{2}\right) < \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b - a}\right)_+ \tag{E.83}$$

where $(\cdot)_+ := \max\{\cdot, 0\}$.

Proof. By Theorem 7.1 we know that if ε_1 satisfies

$$\xi(p_A) + \xi(1 - p_B) > 1, \tag{E.84}$$

then it is guaranteed that

$$q(y_A|\,\boldsymbol{x};\,\varepsilon_1) > \max_{y \neq y_A} q(y|\,\boldsymbol{x};\,\varepsilon_1). \tag{E.85}$$

The proof is thus complete if we show that (E.84) reduces to (E.83). For that purpose, denote by f_0 and f_1 density functions of ε_0 and ε_1, respectively, and let $I_0 = [a,\, b]^m$ and $I_1 := \prod_{i=1}^m [a + \alpha_i,\, b + \alpha_i]$ bet the support of ε_0 and ε_1. Consider

$$f_0(z) = \begin{cases} (b-a)^{-m} & z \in I_0, \\ 0 & \text{otherwise} \end{cases} \tag{E.86}$$

$$f_1(z) = \begin{cases} (b-a)^{-m} & z \in I_1, \\ 0 & \text{otherwise.} \end{cases} \tag{E.87}$$

Let $S_0 := I_0 \backslash I_1$. Then, for any $z \in I_0 \cup I_1$

$$\Lambda(z) = \frac{f_1(z)}{f_0(z)} = \begin{cases} 0 & z \in S_0, \\ 1 & z \in I_0 \cap I_1, \\ \infty & z \in I_1 \backslash I_0. \end{cases} \tag{E.88}$$

Note that

$$\mathcal{P}_0\left(S_0\right) = 1 - \mathcal{P}_0\left(I_1\right) \tag{E.89}$$

$$= 1 - \prod_{i=1}^m \mathcal{P}\left(a + \alpha_i \leqslant \varepsilon_{0,i} \leqslant b + \alpha_i\right) \tag{E.90}$$

$$= 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \tag{E.91}$$

where $(x)_+ = \max\{x,\, 0\}$. We then compute ζ for $t \geqslant 0$

$$\zeta(t) = \mathcal{P}\left(\Lambda(\varepsilon_0) \leqslant t\right) = \begin{cases} \mathcal{P}_0\left(S_0\right) & t < 1, \\ \mathcal{P}_0\left(I_0\right) & t \geqslant 1. \end{cases} = \begin{cases} 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ & t < 1, \\ 1 & t \geqslant 1. \end{cases} \tag{E.92}$$

Recall that $\zeta^{-1}(p) := \inf\{t \geqslant 0 \colon\ \zeta(t) \geqslant p\}$ for $p \in [0,\, 1]$ and hence

$$\zeta^1(p) = \begin{cases} 0 & p \leqslant 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \\ 1 & p > 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \end{cases}, \tag{E.93}$$

In order to evaluate ξ, we compute the lower and strict lower level sets at $t = \zeta^{-1}(p)$. Recall that $\underline{S}_t = \{z \in \mathbb{R}^m_{\geqslant 0} \colon\ \Lambda(z) < t\}$ and $\overline{S}_t = \{z \in \mathbb{R}^m_{\geqslant 0} \colon\ \Lambda(z) \leqslant t\}$ and consider

$$\underline{S}_{\zeta^{-1}(p)} = \begin{cases} \varnothing & p \leqslant 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \\ S_0 & p > 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \end{cases}, \tag{E.94}$$

and

$$\overline{S}_{\zeta^{-1}(p)} = \begin{cases} S_0 & p \leqslant 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \\ I_0 & p > 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \end{cases}, \tag{E.95}$$

Suppose $p \leqslant 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+$. Then $\underline{S}_{\zeta^{-1}(p)} = \varnothing$ and $\overline{S}_{\zeta^{-1}(p)} = S_0$ and hence

$$p \leqslant 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \Rightarrow \xi(p) = \sup\{\mathcal{P}_1(S) \colon\ S \subseteq S_0,\, \mathcal{P}_0(S) \leqslant p\} = 0. \tag{E.96}$$

Condition (E.84) can thus be satisfied only if $p_A > 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+$ and $1 - p_B > 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+$. In this case $\underline{S}_{\zeta^{-1}(p)} = S_0$ and $\overline{S}_{\zeta^{-1}(p)} = I_0$. For $p \in [0,\, 1]$ let $\mathcal{S}_p = \{S \subseteq \mathbb{R}^m \colon\ S_0 \subseteq S \subseteq I_0,\, \mathcal{P}_0(S) \leqslant p\}$. Then

$$p > 1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \Rightarrow \xi(p) = \sup_{S \in \mathcal{S}_p} \mathcal{P}_1(S). \tag{E.97}$$

We can write any $S \in \mathcal{S}_p$ as the disjoint union $S = S_0 \,\dot\cup\, T$ for some $T \subseteq I_0 \cap I_1$ such that $\mathcal{P}_0(S_0 \,\dot\cup\, T) \leqslant p$. Note that $\mathcal{P}_1(S_0) = 0$ and for any $z \in T$, we have $f_0(z) = f_1(z)$. Hence

$$\mathcal{P}_1(S) = \mathcal{P}_1(T) = \mathcal{P}_0(T) \tag{E.98}$$

$$\leqslant p - \mathcal{P}_0(S_0) = p - \left(1 - \prod_{i=1}^m \left(1 - \frac{|\alpha_i|}{b-a}\right)_+\right). \tag{E.99}$$

Thus, The supremum of the left hand side over all $S \in \mathcal{S}_p$ equals the supremum of the right hand side over all $T \in \{T' \subseteq I_0 \cap I_1 \colon \mathcal{P}_0(T') \leqslant 1 - \mathcal{P}_0(S_0)\}$

$$\sup_{S \in \mathcal{S}_p} \mathcal{P}_1(S) = \sup\{\mathcal{P}_1(T') \colon T' \subseteq I_0 \cap I_1, \mathcal{P}_0(T') \leqslant p - \mathcal{P}_0(S_0)\} \tag{E.100}$$

$$= p - \left(1 - \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+\right). \tag{E.101}$$

Hence, computing ξ at p_A and $1 - p_B$ yields

$$\xi(p_A) = p_A - \left(1 - \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+\right), \tag{E.102}$$

$$\xi(1 - p_B) = 1 - p_B - \left(1 - \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+\right). \tag{E.103}$$

Finally, condition (E.84) is satisfied whenever α satisfies

$$1 - \left(1 - p_B - \left(1 - \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+\right)\right) < p_A - \left(1 - \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+\right) \tag{E.104}$$

$$\iff$$

$$p_B + 1 - \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ < p_A - 1 + \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \tag{E.105}$$

$$\iff 2 - p_A + p_B < 2 \cdot \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \tag{E.106}$$

$$\iff 1 - \left(\frac{p_A - p_B}{2}\right) < \prod_{i=1}^{m} \left(1 - \frac{|\alpha_i|}{b-a}\right)_+ \tag{E.107}$$

what concludes the proof. QED.

E.2.4 Laplacian Smoothing

Corollary E.4. Suppose $\mathcal{Z} = \mathbb{R}$ and $\varepsilon_0 \sim \mathcal{L}(0, b)$ follows a Laplace distribution with mean 0 and scale parameter $b > 0$. Let $\varepsilon_1 := \alpha + \varepsilon_0$ for $\alpha \in \mathbb{R}$. Suppose that the ε_0-smoothed classifier g is (p_A, p_B)-confident at $x \in \mathcal{X}$ for some $y_A \in \mathcal{Y}$. Then, it holds that

$q(y_A|\,\boldsymbol{x};\,\varepsilon_1) > \max_{y\neq y_A} q(y|\,\boldsymbol{x};\,\varepsilon_1)$ if α satisfies

$$|\alpha| < \begin{cases} -b\cdot\log\left(4\,p_B\,(1-p_A)\right) & (p_A=\frac{1}{2}\ \wedge\ \ p_B<\frac{1}{2})\ \vee\ (p_A>\frac{1}{2}\ \wedge\ p_B=\frac{1}{2}), \\ -b\cdot\log\left(1-p_A+p_B\right) & p_A>\frac{1}{2}\ \wedge\ p_B<\frac{1}{2}. \end{cases} \tag{E.108}$$

Proof. By Theorem 7.1 we know that if ε_1 satisfies

$$\xi(p_A)+\xi(1-p_B)>1, \tag{E.109}$$

then it is guaranteed that

$$q(y_A|\,\boldsymbol{x};\,\varepsilon_1) > \max_{y\neq y_A} q(y|\,\boldsymbol{x};\,\varepsilon_1). \tag{E.110}$$

The proof is thus complete if we show that (E.109) reduces to (E.108). For that purpose denote by f_0 and f_1 density functions of ε_0 and ε_1, respectively, and consider

$$f_0(z)=\frac{1}{2b}\exp\left(-\frac{|z|}{b}\right),\quad f_1(z)=\frac{1}{2b}\exp\left(-\frac{|z-\alpha|}{b}\right). \tag{E.111}$$

Due to symmetry, assume without loss of generality that $\alpha\geqslant 0$. Then for $z\in\mathbb{R}$

$$\Lambda(z)=\frac{f_1(z)}{f_0(z)}=\exp\left(-\frac{|z-\alpha|-|z|}{b}\right) \tag{E.112}$$

$$=\begin{cases} \exp\left(-\frac{\alpha}{b}\right) & z<0, \\ \exp\left(\frac{2z-\alpha}{b}\right) & 0\leqslant z<\alpha, \\ \exp\left(\frac{\alpha}{b}\right) & z\geqslant\alpha. \end{cases} \tag{E.113}$$

Note that the CDFs for ε_0 and ε_1 are given by

$$F_0(z)=\begin{cases} \frac{1}{2}\exp\left(\frac{z}{b}\right) & z\leqslant 0, \\ 1-\frac{1}{2}\exp\left(-\frac{z}{b}\right) & z>0, \end{cases} \tag{E.114}$$

$$F_1(z)=\begin{cases} \frac{1}{2}\exp\left(\frac{z-\alpha}{b}\right) & z\leqslant\alpha, \\ 1-\frac{1}{2}\exp\left(-\frac{z-\alpha}{b}\right) & z>\alpha. \end{cases} \tag{E.115}$$

Note that for $\exp\left(-\frac{\alpha}{b}\right) \leqslant t < \exp\left(\frac{\alpha}{b}\right)$ we have

$$\begin{aligned}
&\mathcal{P}_0\left(\exp\left(\frac{2\varepsilon_0-\alpha}{b}\right) \leqslant t \ \wedge\ 0 \leqslant \varepsilon_0 < \alpha\right) \\
&\qquad = \mathcal{P}_0\left(\exp\left(-\frac{\alpha}{b}\right) \leqslant \exp\left(\frac{2\varepsilon_0-\alpha}{b}\right) \leqslant t\right) && \text{(E.116)}\\
&\qquad = \mathcal{P}_0\left(0 \leqslant \varepsilon_0 \leqslant \frac{b\log(t)+\alpha}{2}\right) && \text{(E.117)}\\
&\qquad = F_0\left(\frac{b\log(t)+\alpha}{2}\right) - F_0(0) && \text{(E.118)}\\
&\qquad = \frac{1}{2} - \frac{1}{2}\exp\left(-\frac{1}{b}\left(\frac{b\log(t)+\alpha}{2}\right)\right) && \text{(E.119)}\\
&\qquad = \frac{1}{2} - \frac{1}{2\sqrt{t}}\exp\left(-\frac{\alpha}{2b}\right). && \text{(E.120)}
\end{aligned}$$

Computing ζ yields

$$\begin{aligned}
\zeta(t) &= \Pr\left(\Lambda(\varepsilon_0) \leqslant t\right) && \text{(E.121)}\\
&= \Pr\left(\exp\left(-\frac{\alpha}{b}\right) \leqslant t \ \wedge\ \varepsilon_0 < 0\right) + \Pr\left(\exp\left(\frac{\alpha}{b}\right) \leqslant t \ \wedge\ \varepsilon_0 \geqslant \alpha\right) \\
&\qquad + \Pr\left(\exp\left(\frac{2\varepsilon_0-\alpha}{b}\right) \leqslant t \ \wedge\ 0 \leqslant \varepsilon_0 < \alpha\right) && \text{(E.122)}\\
&= \begin{cases} 0 & t < \exp\left(-\frac{\alpha}{b}\right), \\ 1 - \frac{1}{2\sqrt{t}}\exp\left(-\frac{\alpha}{2b}\right) & \exp\left(-\frac{\alpha}{b}\right) \leqslant t < \exp\left(\frac{\alpha}{b}\right), \\ 1 & t \geqslant \exp\left(\frac{\alpha}{b}\right). \end{cases} && \text{(E.123)}
\end{aligned}$$

The inverse is then given by

$$\zeta^{-1}(p) = \begin{cases} 0 & p < \frac{1}{2}, \\ \frac{1}{4(1-p)^2}\exp\left(-\frac{\alpha}{b}\right) & \frac{1}{2} \leqslant p < 1 - \frac{1}{2}\exp(-\frac{\alpha}{b}), \\ \exp\left(\frac{\alpha}{b}\right) & p \geqslant 1 - \frac{1}{2}\exp(-\frac{\alpha}{b}). \end{cases} \qquad \text{(E.124)}$$

In order to evaluate ξ, we compute the lower and strict lower level sets at $t = \zeta^{-1}(p)$. Recall

that $\underline{S}_t = \{z \in \mathbb{R}\colon\ \Lambda(z) < t\}$ and $\overline{S}_t = \{z \in \mathbb{R}\colon\ \Lambda(z) \leqslant t\}$ and consider

$$\underline{S}_{\zeta^{-1}(p)} = \begin{cases} \varnothing & p \leqslant \frac{1}{2}, \\ \left(-\infty,\, b \cdot \log\left(\frac{1}{2(1-p)}\right)\right) & \frac{1}{2} < p < 1 - \frac{1}{2}\exp\left(-\frac{\alpha}{b}\right), \\ (-\infty,\, \alpha\big], & p \geqslant 1 - \frac{1}{2}\exp\left(-\frac{\alpha}{b}\right) \end{cases} \tag{E.125}$$

and

$$\overline{S}_{\zeta^{-1}(p)} = \begin{cases} \varnothing & p < \frac{1}{2}, \\ \left(-\infty,\, b \cdot \log\left(\frac{1}{2(1-p)}\right)\right] & \frac{1}{2} \leqslant p < 1 - \frac{1}{2}\exp\left(-\frac{\alpha}{b}\right), \\ \mathbb{R} & p \geqslant 1 - \frac{1}{2}\exp\left(-\frac{\alpha}{b}\right). \end{cases} \tag{E.126}$$

Suppose $p < 1/2$. Then $\underline{S}_{\zeta^{-1}(p)} = \overline{S}_{\zeta^{-1}(p)} = \varnothing$ and hence $\xi(p) = 0$ and condition (E.109) cannot be satisfied. If $p = 1/2$, then $\underline{S}_{\zeta^{-1}(p)} = \varnothing$ and $\overline{S}_{\zeta^{-1}(p)} = (-\infty, 0]$. Note that for $z \leqslant 0$ we have $f_1(z) = f_0(z)\exp(-\alpha/b)$ and hence for any $S \subseteq \overline{S}_{\zeta^{-1}(1/2)}$ we have $\mathcal{P}_1(S) = \exp(-\alpha/b) \cdot \mathcal{P}_0(S)$. We can thus compute ξ at $1/2$ as

$$p = \frac{1}{2} \Rightarrow \xi\left(1/2\right) = \sup\left\{\mathcal{P}_1(S)\colon\ S \subseteq\ (-\infty,\, 0],\, \mathcal{P}_0(S) \leqslant \frac{1}{2}\right\} = \frac{1}{2}. \tag{E.127}$$

Now suppose $1/2 < p < 1 - 1/2\exp(-\alpha/b)$. In this case, $\underline{S}_{\zeta^{-1}(p)} = (-\infty,\, b \cdot \log(1/2(1-p)))$ and $\overline{S}_{\zeta^{-1}(p)} = \left(-\infty,\, b \cdot \log\left(1/2(1-p)\right)\right]$. Since the singleton $\{b \cdot \log(1/2(1-p))\}$ has no probability mass under both $\mathcal{P}_0$ and $\mathcal{P}_1$, the function ξ is straight forward to compute: if $\frac{1}{2} < p < 1 - \frac{1}{2}\exp(-\frac{\alpha}{b})$, then

$$\xi(p) = \mathcal{P}\left(\varepsilon_1 \leqslant b \cdot \log\left(\frac{1}{2(1-p)}\right)\right) \tag{E.128}$$

$$= \frac{1}{2}\exp\left(\frac{b \cdot \log\left(\frac{1}{2(1-p)}\right) - \alpha}{b}\right) = \frac{1}{4(1-p)}\exp\left(-\frac{\alpha}{b}\right). \tag{E.129}$$

Finally, consider the case where $p \geqslant 1 - 1/2\exp(-\alpha/b)$. Then $\underline{S}_{\zeta^{-1}(p)} = (-\infty,\, \alpha]$ and $\overline{S}_{\zeta^{-1}(p)} = \mathbb{R}$. Any $(-\infty,\, \alpha] \subseteq S \subseteq \mathbb{R}$ can then be written as $S = (-\infty,\, \alpha] \dot{\cup}\, T$ for some $T \subseteq (\alpha,\, \infty)$. Hence

$$\mathcal{P}_1(S) = \Pr(\varepsilon_1 \leqslant \alpha) + \mathcal{P}_1(T) = \frac{1}{2} + \exp\left(\frac{\alpha}{b}\right)\mathcal{P}_0(T), \tag{E.130}$$

$$\mathcal{P}_0(S) = \Pr(\varepsilon_0 \leqslant \alpha) + \mathcal{P}_0(T) = 1 - \frac{1}{2}\exp(-\frac{\alpha}{b}) + \mathcal{P}_0(T). \tag{E.131}$$

Thus, if $p \geqslant 1 - \frac{1}{2}\exp(-\frac{\alpha}{b})$, then

$$\begin{aligned}
\xi(p) &= \sup\left\{\mathcal{P}_1(S)\colon\ (-\infty,\, \alpha] \subseteq S \subseteq \mathbb{R},\, \mathcal{P}_0(S) \leqslant p\right\} && \text{(E.132)}\\
&= \frac{1}{2} + \sup\left\{\mathcal{P}_1(T)\colon\ T \subseteq (\alpha,\, \infty),\, \mathcal{P}_0(T) \leqslant p - 1 + \frac{1}{2}\exp\left(-\frac{\alpha}{b}\right)\right\} && \text{(E.133)}\\
&= \frac{1}{2} + \exp\left(\frac{\alpha}{b}\right)\left(p - 1 + \frac{1}{2}\exp\left(-\frac{\alpha}{b}\right)\right) && \text{(E.134)}\\
&= 1 - \exp\left(\frac{\alpha}{b}\right)(1-p)\,. && \text{(E.135)}
\end{aligned}$$

In order to evaluate condition (E.109), consider

$$1 - \xi(1 - p_B) = \begin{cases} 1 & p_B > \frac{1}{2} \\ \frac{1}{2} & p_B = \frac{1}{2} \\ 1 - \frac{1}{4p_B}\exp\left(-\frac{\alpha}{b}\right) & \frac{1}{2} > p_B > \exp\left(-\frac{\alpha}{b}\right) \\ \exp\left(\frac{\alpha}{b}\right) p_B & \exp\left(-\frac{\alpha}{b}\right) \geqslant p_B, \end{cases} \tag{E.136}$$

$$\xi(p_A) = \begin{cases} 0 & p_A < \frac{1}{2} \\ \frac{1}{2} & p_A = \frac{1}{2} \\ \dfrac{1}{4(1-p_A)}\exp\left(-\dfrac{\alpha}{b}\right) & \frac{1}{2} < p_A < 1 - \frac{1}{2}\exp(-\frac{\alpha}{b}) \\ 1 - \exp\left(\dfrac{\alpha}{b}\right)(1-p_A) & p_A \geqslant 1 - \frac{1}{2}\exp(-\frac{\alpha}{b}). \end{cases} \tag{E.137}$$

Note that the case $p_B > 1/2$ can be ruled out, since by assumption $p_A \geqslant p_B$. If $p_A = 1/2$, then we need $p_B < 1/2$. Thus, if $p_A = 1/2$, then condition (E.109) is satisfied if $p_B < 1/2$ and

$$\begin{aligned}
&\max\left\{1 - \frac{1}{4p_B}\exp\left(-\frac{\alpha}{b}\right),\, \exp\left(\frac{\alpha}{b}\right)\cdot p_B\right\} < \frac{1}{2} && \text{(E.138)}\\
&\qquad\Longleftrightarrow\ p_B \cdot \exp\left(\frac{\alpha}{b}\right) < \frac{1}{2} && \text{(E.139)}\\
&\qquad\Longleftrightarrow\ \alpha < -b \cdot \log(2p_B)\,. && \text{(E.140)}
\end{aligned}$$

Now consider the case where $p_A > 1/2$. If $p_B = 1/2$, then condition (E.109) is satisfied if

$$\frac{1}{2} < \min\left\{ \frac{1}{4(1-p_A)} \exp\left(-\frac{\alpha}{b}\right),\, 1 - \exp\left(\frac{\alpha}{b}\right)(1-p_A) \right\} \tag{E.141}$$

$$\iff \frac{1}{2} < 1 - \exp\left(\frac{\alpha}{b}\right)(1-p_A) \tag{E.142}$$

$$\iff \alpha < -b \cdot \log\big(2(1-p_A)\big)\,. \tag{E.143}$$

If on the other hand, $p_A > 1/2$ and $p_B < 1/2$, condition (E.109) is satisfied if

$$\begin{aligned} &\max\left\{ 1 - \frac{1}{4p_B} \exp\left(-\frac{\alpha}{b}\right),\, \exp\left(\frac{\alpha}{b}\right) \cdot p_B \right\} < \\ &\qquad \min\left\{ \frac{1}{4(1-p_A)} \exp\left(-\frac{\alpha}{b}\right),\, 1 - \exp\left(\frac{\alpha}{b}\right)(1-p_A) \right\} \end{aligned} \tag{E.144}$$

$$\iff p_B \cdot \exp\left(\frac{\alpha}{b}\right) < 1 - \exp\left(\frac{\alpha}{b}\right)(1-p_A) \tag{E.145}$$

$$\iff \alpha < -b \cdot \log\left(1 - p_A + p_B\right). \tag{E.146}$$

Finally, we get that condition (E.109) is satisfied, if

$$|\alpha| < \begin{cases} -b \cdot \log\big(4\, p_B\,(1-p_A)\big) & (p_A = \frac{1}{2} \,\wedge\, p_B < \frac{1}{2}) \vee (p_A > \frac{1}{2} \,\wedge\, p_B = \frac{1}{2}) \\ -b \cdot \log\left(1 - p_A + p_B\right) & p_A > \frac{1}{2} \,\wedge\, p_B < \frac{1}{2} \end{cases} \tag{E.147}$$

what concludes the proof. QED.

E.2.5 Folded Gaussian Smoothing

Corollary E.5. Suppose $\mathcal{Z} = \mathbb{R}_{\geqslant 0}$, $\varepsilon_0 \sim |\mathcal{N}(0,\, \sigma)|$ and $\varepsilon_1 := \alpha + \varepsilon_0$ for some $\alpha > 0$. Suppose that the ε_0-smoothed classifier g is $(p_A,\, p_B)$-confident at $x \in \mathcal{X}$ for some $y_A \in \mathcal{Y}$. Then, it holds that $q(y_A|\, \boldsymbol{x};\, \varepsilon_1) > \max_{y \neq y_A} q(y|\, \boldsymbol{x};\, \varepsilon_1)$ if α satisfies

$$\alpha < \sigma \cdot \left(\Phi^{-1}\left(\frac{1 + \min\{p_A,\, 1 - p_B\}}{2} \right) - \Phi^{-1}\left(\frac{3}{4}\right) \right). \tag{E.148}$$

Proof. By Theorem 7.1 we know that if ε_1 satisfies

$$\xi(p_A) + \xi(1 - p_B) > 1, \tag{E.149}$$

then it is guaranteed that

$$q(y_A|\,\boldsymbol{x};\,\varepsilon_1) > \max_{y\neq y_A} q(y|\,\boldsymbol{x};\,\varepsilon_1). \tag{E.150}$$

The proof is thus complete if we show that (E.149) reduces to (E.148). For that purpose denote by f_0 and f_1 density functions of ε_0 and ε_1, respectively, and consider

$$f_0(z) = \begin{cases} \frac{2}{\sqrt{2\pi}\sigma}\exp\left(-\frac{z^2}{2\sigma^2}\right) & z \geqslant 0 \\ 0 & z < 0 \end{cases} \tag{E.151}$$

$$f_1(z) = \begin{cases} \frac{2}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(z-\alpha)^2}{2\sigma^2}\right) & z \geqslant \alpha \\ 0 & z < \alpha. \end{cases} \tag{E.152}$$

Then, for $z \geqslant 0$,

$$\Lambda(z) = \frac{f_1(z)}{f_0(z)} = \begin{cases} 0 & z < \alpha, \\ \exp\left(\frac{z\alpha}{\sigma^2} - \frac{\alpha^2}{2\sigma^2}\right) & z \geqslant \alpha. \end{cases} \tag{E.153}$$

Let $t_\alpha := \exp\left(\frac{\alpha^2}{2\sigma^2}\right)$ and suppose $t < t_\alpha$. Then

$$\zeta(t) = \Pr\left(\Lambda(\varepsilon_0) \leqslant t\right) = \Pr\left(\varepsilon_0 < \alpha\right) \tag{E.154}$$

$$= \int_0^\alpha \frac{2}{\sqrt{2\pi}\sigma}\exp\left(-\frac{z^2}{2\sigma^2}\right)\,dz \tag{E.155}$$

$$= 2\cdot\int_0^{\alpha/\sigma} \frac{1}{\sqrt{2\pi}}\exp\left(-\frac{s^2}{2}\right)\,ds = 2\cdot\Phi\left(\frac{\alpha}{\sigma}\right) - 1. \tag{E.156}$$

If $t \geqslant t_\alpha$, then

$$\zeta(t) = \Pr\left(\Lambda(\varepsilon_0) \leqslant t\right) \tag{E.157}$$

$$= \Pr\left(\frac{\varepsilon_0\,\alpha}{\sigma^2} - \frac{\alpha^2}{2\sigma^2} \leqslant \log(t) \;\wedge\; \varepsilon_0 \geqslant \alpha\right) + \Pr\left(\varepsilon_0 < \alpha\right) \tag{E.158}$$

$$= \Pr\left(\varepsilon_0 \leqslant \frac{\sigma^2}{\alpha}\log(t) + \frac{1}{2}\alpha\right) = 2\cdot\Phi\left(\frac{\sigma}{\alpha}\log(t) + \frac{\alpha}{2\sigma}\right) - 1 \tag{E.159}$$

and hence

$$\zeta(t) = \begin{cases} 2 \cdot \Phi\left(\frac{\alpha}{\sigma}\right) - 1 & t < t_\alpha \\ 2 \cdot \Phi\left(\frac{\sigma}{\alpha} \log(t) + \frac{\alpha}{2\sigma}\right) - 1 & t \geqslant t_\alpha. \end{cases} \tag{E.160}$$

Note that $\zeta(t_\alpha) = 2 \cdot \Phi\left(\frac{\alpha}{\sigma}\right) - 1$ and let $p_\alpha := \zeta(t_\alpha)$. Recall that $\zeta^{-1}(p) := \inf\{t \geqslant 0 \colon\ \zeta(t) \geqslant p\}$, which yields

$$\zeta^{-1}(p) = \begin{cases} 0 & p \leqslant p_\alpha \\ \exp\left(\frac{\alpha}{\sigma}\Phi^{-1}\left(\frac{1+p}{2}\right) - \frac{\alpha^2}{2\sigma^2}\right) & p > p_\alpha. \end{cases} \tag{E.161}$$

In order to evaluate ξ we compute the lower and strict lower level sets at $t = \zeta^{-1}(p)$. Recall that $\underline{S}_t = \{z \in \mathbb{R}_{\geqslant 0} \colon\ \Lambda(z) < t\}$ and $\overline{S}_t = \{z \in \mathbb{R}_{\geqslant 0} \colon\ \Lambda(z) \leqslant t\}$. Let $S_0 := [0,\, \alpha)$ and note that if $p \leqslant p_\alpha$, we have $\zeta^{-1}(p) = 0$ and hence $\underline{S}_{\zeta^{-1}(p)} = \varnothing$ and $\overline{S}_{\zeta^{-1}(p)} = S_0$. If, on the other hand $p > p_\alpha$, then

$$\underline{S}_{\zeta^{-1}(p)} = \left\{z \geqslant 0 \colon\ \Lambda(z) < \zeta^{-1}(p)\right\} \tag{E.162}$$

$$= S_0 \cup \left\{z \geqslant \alpha \colon\ \frac{z\,\alpha}{\sigma^2} - \frac{\alpha^2}{2\sigma^2} < \frac{\alpha}{\sigma}\Phi^{-1}\left(\frac{1+p}{2}\right) - \frac{\alpha^2}{2\sigma^2}\right\} \tag{E.163}$$

$$= S_0 \cup \left\{z \geqslant \alpha \colon\ z < \sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right\} \tag{E.164}$$

$$= S_0 \cup \left[\alpha,\, \sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right) \tag{E.165}$$

and

$$\overline{S}_{\zeta^{-1}(p)} = \left\{z \geqslant 0 \colon\ \Lambda(z) \leqslant \zeta^{-1}(p)\right\} \tag{E.166}$$

$$= S_0 \cup \left\{z \geqslant \alpha \colon\ \frac{z\,\alpha}{\sigma^2} - \frac{\alpha^2}{2\sigma^2} \leqslant \frac{\alpha}{\sigma}\Phi^{-1}\left(\frac{1+p}{2}\right) - \frac{\alpha^2}{2\sigma^2}\right\} \tag{E.167}$$

$$= S_0 \cup \left\{z \geqslant \alpha \colon\ z \leqslant \sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right\} \tag{E.168}$$

$$= S_0 \cup \left[\alpha,\, \sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right] \tag{E.169}$$

$$= \underline{S}_{\zeta^{-1}(p)} \cup \left\{\sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right\}. \tag{E.170}$$

In other words

$$\underline{S}_{\zeta^{-1}(p)} = \begin{cases} \varnothing & p \leqslant p_\alpha, \\ S_0 \cup \left[\alpha,\, \sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right) & p > p_\alpha, \end{cases} \tag{E.171}$$

$$\overline{S}_{\zeta^{-1}(p)} = \begin{cases} S_0 & p \leqslant p_\alpha, \\ S_0 \cup \left[\alpha,\, \sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right] & p > p_\alpha. \end{cases} \tag{E.172}$$

Let $\mathcal{S}_p := \{S \subseteq \mathbb{R}_{\geqslant 0} \colon\ \underline{S}_{\zeta^{-1}(p)} \subseteq S \subseteq \overline{S}_{\zeta^{-1}(p)},\, \mathcal{P}_0(S) \leqslant p\}$ and recall that $\xi(p) = \sup_{S \in \mathcal{S}_p} \mathcal{P}_1(S)$. Note that for $p \leqslant p_\alpha$, we have $\mathcal{S}_p = \{S \subseteq \mathbb{R}_{\geqslant 0} \colon\ S \subseteq S_0 \ \wedge\ \mathcal{P}_0(S) \leqslant p\}$ and for $S \subseteq S_0$, it holds that $\mathcal{P}_1(S) = 0$. Hence

$$p \leqslant p_\alpha \Rightarrow \xi\,(p) = \sup_{S \in \mathcal{S}_p} \mathcal{P}_1(S) = 0. \tag{E.173}$$

If $p > p_\alpha$, then

$$\begin{aligned} \mathcal{S}_p = \Bigg\{ S \subseteq \mathbb{R}_{\geqslant 0} \colon\ & S_0 \cup \left[\alpha,\, \sigma \cdot \Phi^{-1}\left(1+p/2\right)\right) \subseteq S \\ & \subseteq S_0 \cup \left[\alpha,\, \sigma \cdot \Phi^{-1}\left(1+p/2\right)\right],\ \wedge\, \mathcal{P}_0(S) \leqslant p \Bigg\}. \end{aligned} \tag{E.174}$$

Since the singleton $\left\{\sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right\}$ has no mass under both $\mathcal{P}_0$ and $\mathcal{P}_1$, we find that if $p > p_\alpha$, then

$$\xi\,(p) = \mathcal{P}\left(0 \leqslant \varepsilon_1 \leqslant \sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right)\right) \tag{E.175}$$

$$= \mathcal{P}\left(0 \leqslant \varepsilon_0 \leqslant \sigma \cdot \Phi^{-1}\left(\frac{1+p}{2}\right) - \alpha\right) \tag{E.176}$$

$$= 2 \cdot \Phi\left(\Phi^{-1}\left(\frac{1+p}{2}\right) - \frac{\alpha}{\sigma}\right) - 1. \tag{E.177}$$

Condition (E.149) can thus be satisfied only if $p_B < p_A$ and

$$2 \cdot \Phi\left(\frac{\alpha}{\sigma}\right) - 1 < \min\{p_A,\, 1 - p_B\} \wedge\ \xi\,(p_A) + \xi\,(1 - p_b) > 1 \tag{E.178}$$

that is equivalent to

$$\begin{aligned}\alpha < \sigma \cdot \Phi^{-1}\left(\frac{1+\min\{p_A,\, 1-p_B\}}{2}\right) \\ \wedge\ \Phi\left(\Phi^{-1}\left(\frac{1+(1-p_B)}{2}\right)-\frac{\alpha}{\sigma}\right)+\Phi\left(\Phi^{-1}\left(\frac{1+p_A}{2}\right)-\frac{\alpha}{\sigma}\right) > \frac{3}{2}.\end{aligned} \tag{E.179}$$

Thus, the following is a sufficient condition for the two inequalities in (E.179) and hence (E.149) to hold

$$\alpha < \sigma \cdot \left(\Phi^{-1}\left(\frac{1+\min\{p_A,\, 1-p_B\}}{2}\right)-\Phi^{-1}\left(\frac{3}{4}\right)\right) \tag{E.180}$$

what completes the proof. QED.

E.3 PROOFS FOR CERTIFYING RESOLVABLE TRANSFORMATIONS

Here, we state the proofs and technical details concerning our results for resolvable transformations. Recall the definition of resolvable transformations from the main part of this chapter.

Definition 7.2 (restated)**.** A transformation $\phi\colon \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ is called resolvable if for any $\alpha \in \mathcal{Z}$ there exists a resolving function $\gamma_\alpha\colon \mathcal{Z} \to \mathcal{Z}$ that is injective, continuously differentiable, has non-vanishing Jacobian and for which

$$\phi(\phi(\boldsymbol{x},\, \alpha),\, \beta) = \phi(\boldsymbol{x},\, \gamma_\alpha(\beta)) \quad \boldsymbol{x} \in \mathcal{X},\, \beta \in \mathcal{Z}. \tag{E.181}$$

We say that ϕ is additive, if $\gamma_\alpha(\beta) = \alpha + \beta$.

E.3.1 Proof of Corollary 7.1

Corollary 7.1 (restated)**.** Suppose that the transformation ϕ in Theorem 7.1 is resolvable with resolving function γ_α. Let $\alpha \in \mathcal{Z}$ and set $\varepsilon_1 := \gamma_\alpha(\varepsilon_0)$ in the definition of the functions ζ and ξ. Then, if α satisfies condition (7.6), it is guaranteed that $g(\phi(\boldsymbol{x}, \alpha);\, \varepsilon_0) = g(\boldsymbol{x};\, \varepsilon_0)$.

Proof. Since ϕ is a resolvable transformation, by definition γ_α is injective, continuously differentiable and has non-vanishing Jacobian. By Jacobi's transformation formula (see e.g., [163]), it follows that the density of ε_1 vanishes outside the image of γ_α and is elsewhere

given by

$$f_1(z) = f_0(\gamma_\alpha^{-1}(z))|\det(J_{\gamma_\alpha^{-1}(z)})| \quad \text{for any } z \in \mathrm{Im}(\gamma_\alpha) \tag{E.182}$$

where $J_{\gamma_\alpha^{-1}(z)}$ is the Jacobian of $\gamma_\alpha^{-1}(z)$. Since f_1 is paramterized by α, it follows from Theorem 7.1 that if α satisfies (7.6) it is guaranteed that $\arg\max_y q(y|\, x,\, \varepsilon_1) = \arg\max_y q(y|\, x,\, \varepsilon_0)$. The statement of the corollary then follows immediately from the observation that for any $y \in \mathcal{Y}$ we have

$$\begin{aligned} q(y|\, \boldsymbol{x};\, \varepsilon_1) &= \mathbb{E}(p(y|\, \phi(\boldsymbol{x},\, \varepsilon_1))) && \text{(E.183)}\\ &= \mathbb{E}(p(y|\, \phi(\boldsymbol{x},\, \gamma_\alpha(\varepsilon_0)))) && \text{(E.184)}\\ &= \mathbb{E}(p(y|\, \phi(\phi(\boldsymbol{x},\, \alpha),\, \varepsilon_0))) && \text{(E.185)}\\ &= q(y|\, \phi(\boldsymbol{x},\, \alpha);\, \varepsilon_0). && \text{(E.186)} \end{aligned}$$

QED.

E.3.2 Gaussian Blur

Recall that the Gaussian blur transformation is given by a convolution with a Gaussian kernel

$$G_\alpha(k) = \frac{1}{\sqrt{2\pi\alpha}} \exp\left(-\frac{k^2}{2\alpha}\right) \tag{E.187}$$

where $\alpha > 0$ is the squared kernel radius. Here we show that the transformation $x \mapsto \phi_B(x) := \boldsymbol{x} * G$ is additive.

Lemma 7.1 (restated)**.** The Gaussian blur transformation is additive, i.e., for any $\alpha,\, \beta \geqslant 0$, we have $\phi_B(\phi_B(\boldsymbol{x},\, \alpha),\, \beta) = \phi_B(\boldsymbol{x},\, \alpha + \beta)$.

Proof. Note that associativity of the convolution operator implies that

$$\begin{aligned} \phi_B(\phi_B(\boldsymbol{x},\, \alpha),\, \beta) &= (\phi_B(\boldsymbol{x},\, \alpha) * G_\beta) && \text{(E.188)}\\ &= ((\boldsymbol{x} * G_\alpha) * G_\beta) && \text{(E.189)}\\ &= (\boldsymbol{x} * (G_\alpha * G_\beta)). && \text{(E.190)} \end{aligned}$$

The claim thus follows, if we can show that $(G_\alpha * G_\beta) = G_{\alpha+\beta}$. Let $\mathcal{F}$ denote the Fourier transformation and $\mathcal{F}^{-1}$ the inverse Fourier transformation and note that by the convolution theorem $(G_\alpha * G_\beta) = \mathcal{F}^{-1}\{\mathcal{F}(G_\alpha) \cdot \mathcal{F}(G_\beta)\}$. Therefore we have to show that $\mathcal{F}(G_\alpha) \cdot \mathcal{F}(G_\beta) =$

$\mathcal{F}(G_{\alpha+\beta})$. For that purpose, consider

$$\mathcal{F}(G_\alpha)(\omega) = \int_{-\infty}^{\infty} G_\alpha(y) \exp(-2\,\pi i\omega y)\, dy \tag{E.191}$$

$$= \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\alpha}} \exp\left(-\frac{y^2}{2\alpha}\right) \exp\left(-2\,\pi i\omega y\right)\, dy \tag{E.192}$$

$$= \frac{1}{\sqrt{2\pi\alpha}} \int_{-\infty}^{\infty} \exp\left(-\frac{y^2}{2\alpha}\right) \Big(\cos\left(2\pi\omega y\right) + i\sin\left(2\pi\omega y\right)\Big)\, dy \tag{E.193}$$

$$\overset{(i)}{=} \frac{1}{\sqrt{2\pi\alpha}} \int_{-\infty}^{\infty} \exp\left(-\frac{y^2}{2\alpha}\right) \cos\left(2\pi\omega y\right)\, dy \tag{E.194}$$

$$\overset{(ii)}{=} \exp\left(-\omega^2\pi^2 2\alpha\right), \tag{E.195}$$

where (i) follows from the fact that the second term is an integral of an odd function over a symmetric range and (ii) follows from $\int_{-\infty}^{\infty} \exp\left(-a\,y^2\right) \cos\left(2\pi\omega y\right)\, dy = \sqrt{\frac{\pi}{a}} \exp(\frac{-(\pi\omega)^2}{a})$ with $a = \frac{1}{2\alpha}$ (see p. 302, eq. 7.4.6 in [2]). This concludes our proof since

$$(\mathcal{F}(G_\alpha) \cdot \mathcal{F}(G_\beta))(\omega) = \exp\left(-\omega^2\pi^2 2\alpha\right) \cdot \exp\left(-\omega^2\pi^2 2\beta\right) \tag{E.196}$$

$$= \exp\left(-\omega^2\pi^2 2(\alpha+\beta)\right) = \mathcal{F}(G_{\alpha+\beta})(\omega) \tag{E.197}$$

and hence

$$(G_\alpha \,*\, G_\beta) = \mathcal{F}^{-1}\{\mathcal{F}(G_\alpha) \cdot \mathcal{F}(G_\beta)\} \tag{E.198}$$

$$= \mathcal{F}^{-1}\{\mathcal{F}(G_{\alpha+\beta})\} = G_{\alpha+\beta}. \tag{E.199}$$

QED.

Remark E.1. We notice that the preceding theorem naturally extends to higher dimensional Gaussian kernels of the form

$$G_\alpha(k) = \frac{1}{(2\pi\alpha)^{\frac{m}{2}}} \exp\left(-\frac{\|\boldsymbol{k}\|^2}{2\alpha}\right), \qquad \boldsymbol{k} \in \mathbb{R}^m. \tag{E.200}$$

Consider

$$\mathcal{F}(G_\alpha)(\omega) = \int_{\mathbb{R}^m} G_\alpha(\boldsymbol{y}) \exp\left(-2\pi i\langle\omega,\, \boldsymbol{y}\rangle\right)\, dy \tag{E.201}$$

$$= \frac{1}{(2\pi\alpha)^{\frac{m}{2}}} \int_{\mathbb{R}^m} \exp\left(-\frac{\|\boldsymbol{y}\|_2^2}{2\alpha} - 2\pi i\langle\omega,\, \boldsymbol{y}\rangle\right)\, d\boldsymbol{y} \tag{E.202}$$

$$= \prod_{j=1}^{m} \left(\frac{1}{\sqrt{2\pi\alpha}} \int_{\mathbb{R}} \exp\left(-\frac{y_j^2}{2\alpha} - 2\pi i \omega_j y_j \right) \, dy_j \right) \tag{E.203}$$

$$= \exp\left(- \|\omega\|_2^2 \, \pi^2 2\alpha \right) \tag{E.204}$$

that leads to $(G_\alpha * G_\beta) = G_{\alpha+\beta}$, and hence additivity.

E.3.3 Brightness and contrast

Recall that the brightness and contrast transformation is defined as

$$\phi_{BC} \colon \mathcal{X} \times \mathbb{R}^2 \to \mathcal{X}, \quad (\boldsymbol{x},\, \alpha) \mapsto e^{\alpha_1}(\boldsymbol{x} + \alpha_2). \tag{E.205}$$

Lemma 7.2 (restated)**.** Let $\boldsymbol{x} \in \mathcal{X}$, $k \in \mathbb{R}$, $\varepsilon_0 \sim \mathcal{N}(0,\, \mathrm{diag}(\sigma^2,\, \tau^2))$ and $\varepsilon_1 \sim \mathcal{N}(0,\, \mathrm{diag}(\sigma^2,\, e^{-2k}\tau^2))$. Suppose that $q(y|\, \boldsymbol{x};\, \varepsilon_0) \geqslant p$ for some $p \in [0,\, 1]$ and $y \in \mathcal{Y}$. Then

$$q(y|\, \boldsymbol{x};\, \varepsilon_1) \geqslant \begin{cases} 2\Phi\left(e^k \Phi^{-1}\left(\frac{1+p}{2} \right) \right) - 1 & k \leqslant 0 \\ 2\left(1 - \Phi\left(e^k \Phi^{-1}(1 - \frac{p}{2}) \right) \right) & k > 0. \end{cases} \tag{E.206}$$

Proof. Note that $\varepsilon_0 \sim \mathcal{N}(0,\, \Sigma)$ and $\varepsilon_1 = A\,\varepsilon_0 \sim \mathcal{N}(0,\, A^2\,\Sigma)$ where

$$A = \begin{pmatrix} 1 & 0 \\ 0 & e^{-k} \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \tau^2 \end{pmatrix} \tag{E.207}$$

and denote by f_0 and f_1 the probability density functions of ε_0 and ε_1, respectively, and denote by $\mathcal{P}_0$ and $\mathcal{P}_1$ the corresponding probability measures. Recall the Definition of Lower Level sets (Definition E.1: for $t \geqslant 0$, (strict) lower level sets are defined as

$$\begin{gathered} \underline{S_t} := \left\{ z \in \mathcal{Z} \colon\ \Lambda(z) < t \right\}, \quad \overline{S_t} := \left\{ z \in \mathcal{Z} \colon\ \Lambda(z) \leqslant t \right\}, \\ \text{where} \quad \Lambda(z) := \frac{f_1(z)}{f_0(z)}. \end{gathered} \tag{E.208}$$

Furthermore, recall that the function ζ is given by

$$t \mapsto \zeta(t) := \mathcal{P}_0\left(\overline{S}_t \right) \tag{E.209}$$

where $\mathcal{P}_0$ is the distribution of ε_0 and note that the generalized inverse of ζ corresponds to

τ_p, i.e.,

$$\zeta^{-1}(p) = \inf\{t \geqslant 0 |\, \zeta(t) \geqslant p\} = \tau_p \tag{E.210}$$

and the function ξ is correspondingly given by

$$\xi(p) = \sup\{\mathcal{P}_1(S) |\, \underline{S}(\zeta^{-1}(p)) \subseteq S \subseteq \overline{S}(\zeta^{-1}(p))\}. \tag{E.211}$$

By assumption we know that $\mathbb{E}(p(y|\, \phi(\boldsymbol{x},\, \varepsilon_0))) = q(y|\, \boldsymbol{x};\, \varepsilon_o) \geqslant p$. Note that by Lemma E.1, for any $p \in [0,\, 1]$ we have that

$$\mathcal{P}_0(\underline{S}_{\zeta^{-1}(p)}) \leqslant p. \tag{E.212}$$

Let $S \subseteq \mathcal{Z}$ be such that $\underline{S}_{\zeta^{-1}(p)} \subseteq S \subseteq \overline{S}_{\zeta^{-1}(p)}$ and $\mathcal{P}_0(S) \leqslant p$. Then, from part (i) of Lemma E.2, it follows that $\mathbb{E}(p(y|\, \phi(\boldsymbol{x},\, \varepsilon_1))) = q(y|\, \boldsymbol{x};\, \varepsilon_1) \geqslant \mathcal{P}_1(S)$. Note that

$$\Lambda(z) = \frac{f_1(z)}{f_0(z)} \tag{E.213}$$

$$= \frac{\left((2\pi)^2 |A^2\, \Sigma|\right)^{-\frac{1}{2}} \exp(-\frac{1}{2}(z^T (A^2 \Sigma)^{-1} z))}{\left((2\pi)^2 |\Sigma|\right)^{-\frac{1}{2}} \exp(-\frac{1}{2}(z^T (\Sigma)^{-1} z))} \tag{E.214}$$

$$= \frac{1}{|A|} \exp\left(-\frac{1}{2} z^T \left((A^2\, \Sigma)^{-1} - \Sigma^{-1}\right) z\right) \tag{E.215}$$

$$= \exp\left(k - \frac{z_2^2}{2\tau^2}\left(e^{2k} - 1\right)\right). \tag{E.216}$$

Note that, if $k = 0$, then $\varepsilon_1 = \varepsilon_0$ and hence the statement holds in this case. Suppose that $k > 0$ and consider

$$\zeta(t) = \mathcal{P}_0\left(\overline{S}_t\right) = \mathcal{P}\left(\exp\left(k - \frac{\varepsilon_{0,2}^2}{2\tau^2}\left(e^{2k} - 1\right)\right) \leqslant t\right) \tag{E.217}$$

$$= 1 - \mathcal{P}\left(\left(\frac{\varepsilon_{0,2}}{\tau}\right)^2 \leqslant 2 \cdot \frac{k - \log(t)}{e^{2k} - 1}\right) \tag{E.218}$$

$$= 1 - F_{\chi^2}\left(2 \cdot \frac{k - \log(t)}{e^{2k} - 1}\right) \tag{E.219}$$

$$= \begin{cases} 0 & t = 0, \\ 1 - F_{\chi^2}\left(2 \cdot \frac{k - \log(t)}{e^{2k} - 1}\right) & 0 < t < e^k, \\ 1 & t \geqslant e^k, \end{cases} \tag{E.220}$$

where F_{χ^2} denotes the CDF of the χ^2-distribution with one degree of freedom. Note that for

any $t \geqslant 0$ we have that $\mathcal{P}_0(\overline{S}_t) = \mathcal{P}_0(\underline{S}_t)$ and thus the inverse $\zeta^{-1}(p) = \inf\{t \geqslant 0 \colon \zeta(t) \geqslant p\}$ is given by

$$\zeta^{-1}(p) = \begin{cases} 0 & p = 0 \\ \exp\left(k - F^{-1}_{\chi^2}(1-p) \cdot \frac{e^{2k}-1}{2}\right) & 0 < p < 1 \\ e^k & p = 1. \end{cases} \tag{E.221}$$

Thus, for any $p \in [0,\, 1]$, we find that

$$\mathcal{P}_0(\overline{S}_{\zeta^{-1}(p)}) = \mathcal{P}_0(\underline{S}_{\zeta^{-1}(p)}) = \zeta(\zeta^{-1}(p)) = p \tag{E.222}$$

and

$$\mathop{\mathbb{E}}_{0}(p(y|\, \phi(\boldsymbol{x},\, \varepsilon_0))) = q(y|\, \boldsymbol{x};\, \varepsilon_0) \geqslant p = \mathcal{P}_0(\overline{S}_{\zeta^{-1}(p)}). \tag{E.223}$$

Part (i) of Lemma E.2 implies that $q(y|\, x;\, \varepsilon_1) \geqslant \mathcal{P}_1(\overline{S}_{\zeta^{-1}(p)})$. Computing $\mathcal{P}_1(\overline{S}_{\zeta^{-1}(p)})$ yields

$$\begin{aligned} q(y|\, x;\, \varepsilon_1) &\geqslant \mathcal{P}_1(\overline{S}_{\zeta^{-1}(p)}) && \text{(E.224)} \\ &= 1 - \mathcal{P}\left(\left(\frac{\varepsilon_{1,2}}{\tau^2}\right)^2 \leqslant (k - \log(\zeta^{-1}(p)))\frac{2}{e^{2k}-1}\right) && \text{(E.225)} \\ &= 1 - \mathcal{P}\left(\left(\frac{\varepsilon_{0,2}}{\tau^2}\right)^2 \leqslant (k - \log(\zeta^{-1}(p)))\frac{2e^{2k}}{e^{2k}-1}\right) && \text{(E.226)} \\ &= 1 - F_{\chi^2}\left((k - \log(\zeta^{-1}(p)))\frac{2e^{2k}}{e^{2k}-1}\right) && \text{(E.227)} \\ &= 1 - F_{\chi^2}\left(\left(k - \left(k - \frac{e^{2k}-1}{2}F^{-1}_{\chi^2}(1-p)\right)\right)\frac{2e^{2k}}{e^{2k}-1}\right) && \text{(E.228)} \\ &= 1 - F_{\chi^2}\left(e^{2k}F^{-1}_{\chi^2}(1-p)\right). && \text{(E.229)} \end{aligned}$$

If, on the other hand, $k < 0$, then

$$\begin{aligned} \zeta(t) &= \mathcal{P}_0\left(\overline{S}_t\right) && \text{(E.230)} \\ &= \mathcal{P}\left(\exp\left(k + \frac{\varepsilon^2_{0,2}}{2\tau^2}\left|e^{2k} - 1\right|\right) \leqslant t\right) && \text{(E.231)} \end{aligned}$$

$$= \mathcal{P}\left(\left(\frac{\varepsilon_{0,2}}{\tau}\right)^2 \leqslant 2 \cdot \frac{\log(t) - k}{\left|e^{2k} - 1\right|}\right) \tag{E.232}$$

$$= F_{\chi^2}\left(2 \cdot \frac{\log(t) - k}{\left|e^{2k} - 1\right|}\right) \tag{E.233}$$

$$= \begin{cases} 0 & t \leqslant e^k, \\ F_{\chi^2}\left(2 \cdot \frac{\log(t)-k}{\left|e^{2k}-1\right|}\right) & t > e^k. \end{cases} \tag{E.234}$$

A similar computation as in the case where $k > 0$ leads to an expression for the inverse $\zeta^{-1}(p) = \inf\{t \geqslant 0 |\, \zeta(t) \geqslant p\}$

$$\zeta^{-1}(p) = \begin{cases} 0 & p = 0, \\ \exp\left(k + F^{-1}_{\chi^2}(p) \cdot \frac{\left|e^{2k}-1\right|}{2}\right) & p > 0. \end{cases} \tag{E.235}$$

Thus, for any $p \in [0,\, 1]$, we find that

$$\mathcal{P}_0(\overline{S}_{\zeta^{-1}(p)}) = \mathcal{P}_0(\underline{S}_{\zeta^{-1}(p)}) = \zeta(\zeta^{-1}(p)) = p \tag{E.236}$$

and

$$\mathbb{E}(p(y |\, \phi(\boldsymbol{x},\, \varepsilon_0))) = q(y |\, \boldsymbol{x};\, \varepsilon_0) \geqslant p = \mathcal{P}_0(\overline{S}_{\zeta^{-1}(p)}). \tag{E.237}$$

Part (i) of Lemma E.2 implies that $g_c^{\varepsilon_1}(x) \geqslant \mathcal{P}_1(\overline{S}_{\zeta^{-1}(p)})$. Computing $\mathcal{P}_1(\overline{S}_{\zeta^{-1}(p)})$ yields

$$q(y |\, x;\, \varepsilon_1) \geqslant \mathcal{P}_1(\overline{S}_{\zeta^{-1}(p)}) \tag{E.238}$$

$$= \mathcal{P}\left(\left(\frac{\varepsilon_{1,2}}{\tau}\right)^2 \leqslant 2 \cdot \frac{\log(\zeta^{-1}(p)) - k}{|e^{2k} - 1|}\right) \tag{E.239}$$

$$= \mathcal{P}\left(\left(\frac{\varepsilon_{0,2}}{\tau}\right)^2 \leqslant 2e^{2k} \cdot \frac{\log(\zeta^{-1}(p)) - k}{|e^{2k} - 1|}\right) \tag{E.240}$$

$$= F_{\chi^2}\left(\left(\left(k + F^{-1}_{\chi^2}(p)\frac{|e^{2k} - 1|}{2}\right) - k\right)\frac{2\, e^{2k}}{|e^{2k} - 1|}\right) \tag{E.241}$$

$$= F_{\chi^2}\left(e^{2k} F^{-1}_{\chi^2}(p)\right). \tag{E.242}$$

Finally, note the following relation between the $\chi^2(1)$ and the standard normal distribution. Let $Z \sim \mathcal{N}(0, 1)$ and denote by Φ the CDF of Z. Then, for any $z \geqslant 0$, $F_{\chi^2}(z) = \Pr(Z^2 \leqslant z) = \Pr(-\sqrt{z} \leqslant Z \leqslant \sqrt{z}) = \Phi(\sqrt{z}) - \Phi(-\sqrt{z}) = 2\Phi(\sqrt{z}) - 1$ and the inverse is thus given

by $F^{-1}_{\chi^2}(p) = (\Phi^{-1}(\frac{1+p}{2}))^2$. It follows that

$$q(y|,\, x;\, \varepsilon_1) \geqslant \begin{cases} 2\Phi\left(e^k \Phi^{-1}\left(\frac{1+p}{2}\right)\right) - 1 & k \leqslant 0, \\ 2\left(1 - \Phi\left(e^k \Phi^{-1}(1 - \frac{p}{2})\right)\right) & k > 0, \end{cases} \tag{E.243}$$

what concludes the proof. QED.

The following lemma establishes another useful property of the distribution of ε_1.

Lemma E.3. Let $\varepsilon_0 \sim \mathcal{N}(0,\, \mathrm{diag}(\sigma^2,\, \tau^2))$, $\alpha = (k,\, b)^T \in \mathbb{R}^2$ and $\varepsilon_1 \sim \mathcal{N}(0,\, \mathrm{diag}(\sigma^2,\, e^{-2k}\tau^2))$. Then, for all $\boldsymbol{x} \in \mathcal{X}$, it holds that $g(\phi_{BC}(\boldsymbol{x},\, \alpha);\, \varepsilon_0) = g(\boldsymbol{x};\, \alpha + \varepsilon_1)$.

Proof. Let $x \in \mathcal{X}$, and write $\varepsilon_i = (\varepsilon_{i,1},\, \varepsilon_{i,2})^T$ for $i = 0,\, 1$. Note that

$$\phi_{BC}(\phi_{BC}(\boldsymbol{x},\, \alpha),\, \varepsilon_0) = e^{\varepsilon_{0,1}}\left(\phi_{BC}(\boldsymbol{x},\, \alpha) + \varepsilon_{0,2} \cdot \nVdash_d\right) = e^{\varepsilon_{0,1}}\left(e^k\left(\boldsymbol{x} + b \cdot \nVdash_d\right) + \varepsilon_{0,2} \cdot \nVdash_d\right) \tag{E.244}$$

$$= e^{\varepsilon_{0,1}+k}\left(x + \left(b + e^{-k}\varepsilon_{0,2}\right) \cdot \nVdash_d\right) = \phi_{BC}(x, \alpha + \tilde{\varepsilon}_0) \tag{E.245}$$

where $\tilde{\varepsilon}_0 = (\varepsilon_{0,1},\, e^{-k}\varepsilon_{0,2})^T$. Note that $\tilde{\varepsilon}_0$ follows a Gaussian distribution since

$$\tilde{\varepsilon}_0 = A \cdot \varepsilon_0, \quad A = \begin{pmatrix} 1 & 0 \\ 0 & e^{-k} \end{pmatrix} \tag{E.246}$$

and hence $\mathbb{E}\,(\tilde{\varepsilon}_0) = A \cdot \mathbb{E}\,(\varepsilon_0) = 0$ and

$$\mathrm{Cov}\,(\tilde{\varepsilon}_0) = \mathbb{E}\left(\varepsilon_0\, A\, A^T\, \varepsilon_0^T\right) = A^2 \cdot \begin{pmatrix} \sigma^2 & 0 \\ 0 & \tau^2 \end{pmatrix} = \begin{pmatrix} \sigma^2 & 0 \\ 0 & e^{-2k}\tau^2 \end{pmatrix}. \tag{E.247}$$

The choice $\varepsilon_1 := \tilde{\varepsilon}_0 \sim \mathcal{N}(0,\, \mathrm{diag}(\sigma_1^2,\, e^{-2k}\sigma_2^2))$ shows that for any $y \in \mathcal{Y}$

$$q(y|\, \phi_{BC}(\boldsymbol{x},\, \alpha);\, \varepsilon_0) = \mathbb{E}\left(p(y|\, \phi(\phi(\boldsymbol{x},\, \alpha),\, \varepsilon_0)\right) \tag{E.248}$$

$$= \mathbb{E}\left(p(y|\, \phi(x,\, \alpha + \varepsilon_1)\right) \tag{E.249}$$

$$= q(y|\, \boldsymbol{x};\, \alpha + \varepsilon_1) \tag{E.250}$$

what concludes the proof. QED.

These observations, together with the Gaussian robustness bound from Corollary E.1 allow us to prove Lemma 7.3.

Lemma 7.3 (restated)**.** Let ε_0 and ε_1 be as in Lemma 7.2 and suppose that

$$q(y_A|\,\boldsymbol{x};\,\varepsilon_1) \geqslant \tilde{p}_A > \tilde{p}_B \geqslant \max_{y \neq y_A} q(y|\,\boldsymbol{x};\,\varepsilon_1). \tag{E.251}$$

Then it is guaranteed that $y_A = g(\phi_{BC}(\boldsymbol{x},\,\alpha);\,\varepsilon_0)$ as long as $\alpha = (k,\,b)^T$ satisfies

$$\sqrt{\left(\frac{k}{\sigma}\right)^2 + \left(\frac{b}{e^{-k}\tau}\right)^2} < \frac{1}{2}\left(\Phi^{-1}\left(\tilde{p}_A\right) - \Phi^{-1}\left(\tilde{p}_B\right)\right). \tag{E.252}$$

Proof. Since $\varepsilon_1 \sim \mathcal{N}(0,\,\mathrm{diag}(\sigma^2,\,e^{-2k}\tau^2))$, it follows from Corollary E.1 that whenever $\alpha = (k,\,b)^T$ satisfies

$$\sqrt{\left(\frac{k}{\sigma}\right)^2 + \left(\frac{b}{e^{-k}\tau}\right)^2} < \frac{1}{2}\left(\Phi^{-1}\left(\tilde{p}_A\right) - \Phi^{-1}\left(\tilde{p}_B\right)\right), \tag{E.253}$$

then it is guaranteed that $y_A = g(\boldsymbol{x};\,\varepsilon_1)$. The statement now directly follows from Lemma E.3. QED.

E.3.4 Gaussian Blur, Brightness, Contrast, and Translation

Recall that the composition of Gaussian Blur, with brightness, contrast and translation is defined as

$$\phi_{BTBC}(\boldsymbol{x}, \alpha) := \phi_B(\phi_T(\phi_{BC}(\boldsymbol{x}, \alpha_k, \alpha_b), \alpha_{Tx}, \alpha_{Ty}), \alpha_B), \tag{E.254}$$

where ϕ_B, ϕ_T and ϕ_{BC} are Gaussian blur, translation, and brightness and contrast transformations respectively as defined before and $\alpha := (\alpha_k, \alpha_b, \alpha_{Tx}, \alpha_{Ty}, \alpha_B)^T \in \mathbb{R}^4 \times \mathbb{R}_{\geqslant 0}$ is the transformation parameter. It is easy to see that this transformation composition satisfies the following properties:

- **(P1)** For arbitrary $\alpha^{(1)}, \alpha^{(2)} \in \mathbb{R}^4 \times \mathbb{R}_{\geqslant 0}$,

$$\phi_{BTBC}(\phi_{BTBC}(\boldsymbol{x}, \alpha^{(1)}), \alpha^{(2)}) = \phi_{BTBC}(x, \alpha) \tag{E.255}$$

  where

$$\begin{aligned}\alpha = \Big(&\alpha_k^{(1)} + \alpha_k^{(2)}, \alpha_b^{(1)} + \alpha_b^{(2)}/e^{\alpha_k^{(1)}}, \\ &\alpha_{Tx}^{(1)} + \alpha_{Tx}^{(2)}, \alpha_{Tx}^{(1)} + \alpha_{Tx}^{(2)}, \alpha_B^{(1)} + \alpha_B^{(2)}\Big).\end{aligned} \tag{E.256}$$

- **(P2)** For an arbitrary $\alpha \in \mathbb{R}^4 \times \mathbb{R}_{\geqslant 0}$, define the parameterized operators:

$$\begin{gathered} \phi_B^\alpha := \phi_B(\cdot; \alpha_B), \quad \phi_T^\alpha := \phi_T(\cdot; \alpha_{Tx}, \alpha_{Ty}), \\ \phi_{BC}^\alpha := \phi_{BC}(\cdot; \alpha_k, \alpha_b) \end{gathered} \tag{E.257}$$

  and let $\phi_1^\alpha, \phi_2^\alpha, \phi_3^\alpha$ be an arbitrary permutation of the above three operators. Then, we have that

$$\phi_{BTBC}(x, \alpha) = \phi_1^\alpha \circ \phi_2^\alpha \circ \phi_3^\alpha(x). \tag{E.258}$$

The property **(P1)** states that ϕ_{BTBC} is almost additive where the exception happens only on the brightness dimension (α_b). The brightness dimension is subject to the same contrast effect implied and proved in Lemma 8 (in main part). The property **(P2)** states that all the three transformations ϕ_B, ϕ_T, and ϕ_{BC} are commutative. The reason is that: (1) ϕ_{BC} is a per-pixel color shift and independent of ϕ_B and ϕ_T; (2) ϕ_B, Gaussian blur, relies on relative position of pixels and the translation with reflection padding, ϕ_T, does not change it.

Based on these two properties, we prove the key results as follows.

Corollary 7.3 (restated)**.** Let $\boldsymbol{x} \in \mathcal{X}$, $k \in \mathbb{R}$ and let $\epsilon_0 := (\epsilon_0^a, \epsilon_0^b)^T$ be a random variable defined as

$$\epsilon_0^a \sim \mathcal{N}(0, \operatorname{diag}(\sigma_k^2,\, \sigma_b^2,\, \sigma_T^2,\, \sigma_T^2)) \operatorname{and} \epsilon_0^b \sim \operatorname{Exp}(\lambda_B). \tag{E.259}$$

Similarly, let $\epsilon_1 := (\epsilon_1^a, \epsilon_1^b)$ be a random variable with

$$\epsilon_1^a \sim \mathcal{N}(0, \operatorname{diag}(\sigma_k^2,\, e^{-2k}\sigma_b^2,\, \sigma_T^2,\, \sigma_T^2)) \operatorname{and} \epsilon_1^b \sim \operatorname{Exp}(\lambda_B). \tag{E.260}$$

For either random variable (denoted as ϵ), recall that $q(y|\boldsymbol{x};\, \epsilon) := \mathbb{E}(p(y|\phi_{BTBC}(\boldsymbol{x},\, \epsilon)))$. Suppose that $q(y|\boldsymbol{x};\, \epsilon_0) \geqslant p$ for some $p \in [0,\, 1]$ and $y \in \mathcal{Y}$. Then $q(y|\boldsymbol{x};\, \epsilon_1)$ satisfies Eq. (11).

Proof. According to the commutative property **(P2)**, we can view $q(y|x;\, \epsilon)$ as

$$q(y|\boldsymbol{x},\, \epsilon) = \mathop{\mathbb{E}}_{\epsilon} p(y|\phi_{BTBC}(\boldsymbol{x},\, \epsilon)) \tag{E.261}$$

$$= \mathop{\mathbb{E}}_{\epsilon_k, \epsilon_b} \underbrace{\mathop{\mathbb{E}}_{\epsilon_{Tx}, \epsilon_{Ty}, \epsilon_B} p(y|\phi_{BC}(\phi_T(\phi_B(\boldsymbol{x}, \epsilon_B), \epsilon_{Tx}, \epsilon_{Ty}), \epsilon_k, \epsilon_b))}_{=: q'(y|\boldsymbol{x};\, \epsilon_k,\, \epsilon_b)}. \tag{E.262}$$

Notice that $q'(y|\boldsymbol{x};\, \epsilon_k,\, \epsilon_b)$ is a deterministic value in $[0,\, 1]$. Its value is dependent on the distribtuion of $\epsilon_{Tx}, \epsilon_{Ty}, \epsilon_B$ and the underlying base classifier. Luckily, the random variables ϵ_0 and ϵ_1 have the same distribution over the components $\epsilon_{Tx}, \epsilon_{Ty}$ and ϵ_B. Thus, they share

the same q' and we write $q(y|\boldsymbol{x};\,\epsilon_0)$ and $q(y|\boldsymbol{x};\,\epsilon_1)$ as

$$q(y|\boldsymbol{x};\,\epsilon_0) = \mathop{\mathbb{E}}_{(\epsilon_k,\epsilon_b)\sim\mathcal{N}(0,\mathrm{diag}(\sigma_k^2,\sigma_b^2))} q'(y|\boldsymbol{x};\,\epsilon_k,\epsilon_b), \tag{E.263}$$

$$q(y|\boldsymbol{x};\,\epsilon_1) = \mathop{\mathbb{E}}_{(\epsilon_k,\epsilon_b)\sim\mathcal{N}(0,\mathrm{diag}(\sigma_k^2,e^{-2k}\sigma_b^2))} q'(y|\boldsymbol{x};\,\epsilon_k,\epsilon_b). \tag{E.264}$$

Now, we directly apply Lemma 7.2 and the desired lower bound for $q(y|\boldsymbol{x};\,\epsilon_1)$ follows. QED.

Lemma 7.6 (restated)**.** Let ϵ_0 and ϵ_1 be as in Corollary 7.3 and suppose that

$$q(y_A|\boldsymbol{x};\,\varepsilon_1) \geqslant \tilde{p}_A > \tilde{p}_B \geqslant \max_{y\neq y_A} q(y|\boldsymbol{x};\,\varepsilon_1). \tag{E.265}$$

Then it is guaranteed that $y_A = g(\phi_{BTBC}(\boldsymbol{x},\,\alpha);\varepsilon_0)$ as long as $p'_A > p'_B$,

$$p'_A = \begin{cases} 0, & \text{if } \tilde{p}_A \leqslant 1-\exp(-\lambda_B\alpha_B), \\ \Phi\left(\Phi^{-1}\left(1-(1-\tilde{p}_A)\exp(\lambda_B\alpha_B)\right)\right. & \\ \quad \left. -\sqrt{\alpha_k^2/\sigma_k^2 + \alpha_b^2/(e^{-2\alpha_k}\sigma_b^2) + (\alpha_{Tx}^2+\alpha_{Ty}^2)/\sigma_T^2}\right), & \text{otherwise} \end{cases} \tag{E.266}$$

and

$$p'_B = \begin{cases} 1, & \text{if } \tilde{p}_B \geqslant \exp(-\lambda_B\alpha_B), \\ 1-\Phi\left(\Phi^{-1}\left(1-\tilde{p}_B\exp(\lambda_B\alpha_B)\right)\right. & \\ \quad \left. -\sqrt{\alpha_k^2/\sigma_k^2 + \alpha_b^2/(e^{-2\alpha_k}\sigma_b^2) + (\alpha_{Tx}^2+\alpha_{Ty}^2)/\sigma_T^2}\right). & \text{otherwise} \end{cases} \tag{E.267}$$

Proof. We notice that for any $y \in \mathcal{Y}$,

$$\begin{aligned} &q(y|\phi_{BTBC}(\boldsymbol{x},\,\alpha);\,\epsilon_0) \\ &\quad = \mathop{\mathbb{E}}_{\epsilon_0} p(y|\phi_{BTBC}(\phi_{BTBC}(\boldsymbol{x},\,\alpha),\,\epsilon_0)) && \text{(E.268)} \\ &\quad \overset{(a.)}{=} \mathop{\mathbb{E}}_{\epsilon_0} p\Big(y\Big|\boldsymbol{x};\,\alpha + ((\epsilon_0)_k, (\epsilon_0)_b/e^{\alpha_k}, (\epsilon_0)_{Tx}, (\epsilon_0)_{Ty}, (\epsilon_0)_B)^T\Big) && \text{(E.269)} \\ &\quad \overset{(b.)}{=} E_{\epsilon_1} p\left(y|\boldsymbol{x};\alpha+\epsilon_1\right). && \text{(E.270)} \end{aligned}$$

The step $(a.)$ uses the property **(P1)** of transformation ϕ_{BTBC}, and the step $(b.)$ follows the definition of ε_1 in Corollary 7.3 (we define $k := \alpha_k$ hereinafter for simplicity). Thus, $g(\phi_{BTBC}(\boldsymbol{x},\,\alpha);\epsilon_0) = g(\boldsymbol{x};\alpha+\epsilon_1)$, and the robustness condition is equivalent to $g(\boldsymbol{x};\alpha+\epsilon_1) = g(\boldsymbol{x};\epsilon_1) = y_A$.

According to Theorem 7.1, to prove the robustness, we only need to show that $\xi(\tilde{p}_A) +$

$\xi(1-\tilde{p}_B) > 1$ given $p'_A > p'_B$. Note that in the definition of ξ, the density functions f_0 and f_1 are for distributions of $\epsilon_1 \sim \mathcal{P}_0$ and $(\alpha + \epsilon_1) \sim \mathcal{P}_1$ respectively.

In the proof below, we will compute the closed-form solution of $\xi(p)$ for any $0 \leqslant p \leqslant 1$, and show that $\xi(\tilde{p}_A) + \xi(1-\tilde{p}_B) > 1$ given $p'_A > p'_B$. To begin with, we write down f_0 and f_1.

$$f_0(z) = \frac{\lambda_B}{(2\pi)^2 \sigma_k \sigma_b \sigma_T^2} \exp\left(-\lambda_B z_B - {}^{(z_{Tx}^2+z_{Ty}^2)}\!/_{2\sigma_T^2} - {}^{z_k^2}\!/_{2\sigma_k^2} - {}^{z_b^2}\!/_{2e^{-2k}\sigma_b^2}\right), \tag{E.271}$$

$$f_1(z) = \begin{cases} \dfrac{\lambda_B \exp(\lambda_B \alpha_B)}{(2\pi)^2 \sigma_k \sigma_b \sigma_T^2} \exp\big(-\lambda_B z_B \\ \qquad - {}^{(z_{Tx}-\alpha_{Tx})^2}\!/_{2\sigma_T^2} - {}^{(z_{Ty}-\alpha_{Ty})^2}\!/_{2\sigma_T^2} \\ \qquad -{}^{(z_k-\alpha_k)^2}\!/_{2\sigma_k^2} - {}^{(z_b-\alpha_b)^2}\!/_{2e^{-2k}\sigma_b^2}\big), \end{cases} \quad \text{if } z_B \geqslant \alpha_B, \tag{E.272}$$

$$\qquad 0, \qquad \text{otherwise,}$$

where $z = (z_k, z_b, z_{Tx}, z_{Ty}, z_B)^T \in \mathbb{R}^4 \times R_{\geqslant 0}$. As a result, function $\Lambda = f_1/f_0$ in Theorem 7.1 writes as

$$\Lambda(z) = \begin{cases} \exp\left(\lambda_B \alpha_B - \dfrac{\alpha_{Tx}^2}{2\sigma_T^2} - \dfrac{\alpha_{Ty}^2}{2\sigma_T^2} - \dfrac{\alpha_k^2}{2\sigma_k^2} - \dfrac{\alpha_b^2}{2e^{-2k}\sigma_b^2}\right. \\ \left. + \dfrac{\alpha_{Tx} z_{Tx}}{\sigma_T^2} + \dfrac{\alpha_{Ty} z_{Ty}}{\sigma_T^2} + \dfrac{\alpha_k z_k}{\sigma_k^2} + \dfrac{\alpha_b z_b}{e^{-2k}\sigma_b^2}\right), \end{cases} \quad \text{if } z_B \geqslant \alpha_B, \tag{E.273}$$

$$\qquad 0 \qquad \text{otherwise.}$$

It turns out that for any $t > 0$,

$$\begin{aligned} &\underline{S}_t = \{f_1/f_0 < t\} \\ &= \Big\{(\hat{z}_k\sigma_k, \hat{z}_b e^{-k}\sigma_b, \hat{z}_{Tx}\sigma_T, \hat{z}_{Ty}\sigma_T)^T \\ &\quad | \, \hat{\alpha}_{Tx}\hat{z}_{Tx} + \hat{\alpha}_{Ty}\hat{z}_{Ty} + \hat{\alpha}_k\hat{z}_k + \hat{\alpha}_b\hat{z}_b \\ &\quad < \ln t + \hat{\alpha}_{Tx}^2/2 + \hat{\alpha}_{Ty}^2/2 + \hat{\alpha}_k^2/2 + \hat{\alpha}_b^2/2 - \lambda_B\alpha_B\Big\} \times [\alpha_B, +\infty) \\ &\quad \cup \mathbb{R}^4 \times [0, \alpha_B), \\ &\overline{S}_t = \{f_1/f_0 \leqslant t\} \\ &= \Big\{(\hat{z}_k\sigma_k, \hat{z}_b e^{-k}\sigma_b, \hat{z}_{Tx}\sigma_T, \hat{z}_{Ty}\sigma_T)^T \\ &\quad | \, \hat{\alpha}_{Tx}\hat{z}_{Tx} + \hat{\alpha}_{Ty}\hat{z}_{Ty} + \hat{\alpha}_k\hat{z}_k + \hat{\alpha}_b\hat{z}_b \\ &\quad \leqslant \ln t + \hat{\alpha}_{Tx}^2/2 + \hat{\alpha}_{Ty}^2/2 + \hat{\alpha}_k^2/2 + \hat{\alpha}_b^2/2 - \lambda_B\alpha_B\Big\} \times [\alpha_B, +\infty) \end{aligned} \tag{E.274}$$

$$\cup \mathbb{R}^4 \times [0, \alpha_B), \tag{E.275}$$

where $\hat{\alpha}_{Tx} = \alpha_{Tx}/\sigma_T, \hat{\alpha}_{Ty} = \alpha_{Ty}/\sigma_T, \hat{\alpha}_k = \alpha_k/\sigma_k, \hat{\alpha}_b = \alpha_b/(e^{-k}\sigma_b)$. When $t = 0$, $\underline{S}_t = \varnothing$ and $\overline{S}_t = \mathbb{R}^4 \times [0, \alpha_B)$. Then, the probability integration shows that

$$\begin{aligned}
\tau_p &= \inf\{t \geqslant 0 : \mathcal{P}_0(\overline{S}_t) \geqslant p\} \\
&= \begin{cases} 0, & \text{if } p \leqslant 1 - \exp(-\lambda_B \alpha_B), \\ \exp\left(\lambda_B \alpha_B + \|\hat{\alpha}_{:-1}\| \Phi^{-1}\left(1 - \exp(\lambda_B \alpha_B)(1-p)\right) - 1/2 \|\hat{\alpha}_{:-1}\|^2\right), & \\ & \text{otherwise,} \end{cases}
\end{aligned} \tag{E.276}$$

where $\|\hat{\alpha}_{:-1}\| = \sqrt{\hat{\alpha}_{Tx}^2 + \hat{\alpha}_{Ty}^2 + \hat{\alpha}_k^2 + \hat{\alpha}_b^2}$. Now we are ready to compute $\xi(p) = \sup\{\mathcal{P}_1(S) : \underline{S}_{\tau_p} \subset S \subset \overline{S}_{\tau_p}\}$. When $p \leqslant 1 - \exp(-\lambda_B \alpha_B)$, we have $S \subset \mathbb{R}^4 \times [0, \alpha_B)$ and $\mathcal{P}_1(S) = 0$ because $\mathcal{P}_1$ has zero mass for any $z_B < \alpha_B$ (see (E.272)). When $p > 1 - \exp(-\lambda_B \alpha_B)$, $\tau_p > 0$. Again, from probability integration, we get

$$\mathcal{P}_1(\underline{S}_{\tau_p}) = \mathcal{P}_1(\overline{S}_{\tau_p}) = \Phi\left(\frac{\ln \tau_p - \lambda_B \alpha_B}{\|\hat{\alpha}_{:-1}\|} - \frac{1}{2}\|\hat{\alpha}_{:-1}\|\right). \tag{E.277}$$

We inject the closed-form solution of τ_p in (E.276) and yield

$$\xi(p) = \mathcal{P}_1(S) = \Phi\left(\Phi^{-1}\left(1 - (1-p)\exp(\lambda_B \alpha_B)\right) - \|\hat{\alpha}_{:-1}\|\right) \tag{E.278}$$

for any S satisfying $\underline{S}_{\tau_p} \subset S \subset \overline{S}_{\tau_p}$. We summarize the above equations and write down the closed-form solution of $\xi(p)$ as such:

$$\xi(p) = \begin{cases} 0, & \text{if } p \leqslant 1 - \exp(-\lambda_B \alpha_B), \\ \Phi\left(\Phi^{-1}\left(1 - (1-p)\exp(\lambda_B \alpha_B)\right) - \|\hat{\alpha}_{:-1}\|\right), & \text{otherwise.} \end{cases} \tag{E.279}$$

We can easily observe that p'_A in lemma statement (E.266) is indeed $\xi(\tilde{p}_A)$, and p'_B (E.267) is indeed $1 - \xi(1 - \tilde{p}_B)$. Therefore,

$$\xi(\tilde{p}_A) + \xi(1 - \tilde{p}_B) > 1 \iff p'_A > p'_B \tag{E.280}$$

and using Theorem 7.1 concludes the proof. QED.

Finally, we simplify the statement of Lemma 7.6 with slight relaxation.

Corollary 7.4 (restated)**.** Let ϵ_0 and ϵ_1 be as in Corollary 7.3 and suppose that

$$q(y_A|\boldsymbol{x};\, \varepsilon_1) \geqslant \tilde{p}_A. \tag{E.281}$$

Then it is guaranteed that $y_A = g(\phi_{BTBC}(\boldsymbol{x},\, \alpha); \varepsilon_0)$ as long as

$$\tilde{p}_A > 1 - \exp(-\lambda_B \alpha_B) \left(1 - \Phi\left(\sqrt{\frac{\alpha_k^2}{\sigma_k^2} + \frac{\alpha_b^2}{e^{-2\alpha_k}\sigma_b^2} + \frac{\alpha_{Tx}^2 + \alpha_{Ty}^2}{\sigma_T^2}}\right)\right). \tag{E.282}$$

Proof. Since $q(y_A|\boldsymbol{x};\, \varepsilon_1) \geqslant \tilde{p}_A$, according to the complement rule, $\max_{y \neq y_A} q(y|\boldsymbol{x};\, \varepsilon_1) < 1 - \tilde{p}_A =: \tilde{p}_B$. Inject the $\tilde{p}_A$ and $\tilde{p}_B$ into Lemma 7.6 we find that $p'_A + p'_B = 1$ always hold. Therefore, $p'_A > 0.5$ guarantees that $p'_A > p'_B$ and thus the robustness. Indeed, from simple algebra, $p'_A > 0.5 \iff$ (E.282). QED.

E.4 PROOFS FOR DIFFERENTIALLY RESOLVABLE TRANSFORMATIONS

Here we provide proofs and technical details for theoretical results about certifying differentially resolvable transformations. First, let us recall the definition of differentially resolvable transformations.

Definition 7.3 (restated)**.** Let $\phi\colon \mathcal{X} \times \mathcal{Z}_\phi \to \mathcal{X}$ be a transformation with noise space $\mathcal{Z}_\phi$ and let $\psi\colon \mathcal{X} \times \mathcal{Z}_\psi \to \mathcal{X}$ be a resolvable transformation with noise space $\mathcal{Z}_\psi$. We say that ϕ can be resolved by ψ if for any $x \in \mathcal{X}$ there exists function $\delta_x\colon \mathcal{Z}_\phi \times \mathcal{Z}_\phi \to \mathcal{Z}_\psi$ such that for any $\beta \in \mathcal{Z}_\phi$

$$\phi(\boldsymbol{x},\, \alpha) = \psi(\phi(\boldsymbol{x},\, \beta),\, \delta_x(\alpha,\, \beta)). \tag{E.283}$$

Theorem 7.2 (restated)**.** Let $\phi\colon \mathcal{X} \times \mathcal{Z}_\phi \to \mathcal{X}$ be a transformation that is resolved by $\psi\colon \mathcal{X} \times \mathcal{Z}_\psi \to \mathcal{X}$. Let $\varepsilon \sim \mathcal{P}_\varepsilon$ be a $\mathcal{Z}_\psi$-valued random variable and suppose that the smoothed classifier $g : \mathcal{X} \to \mathcal{Y}$ given by $q(y|\,\boldsymbol{x}; \varepsilon) = \mathbb{E}(p(y|\,\psi(\boldsymbol{x},\, \varepsilon)))$ predicts $g(\boldsymbol{x};\, \varepsilon) = y_A = \arg\max_y q(y|\,\boldsymbol{x}; \varepsilon)$. Let $\mathcal{S} \subseteq \mathcal{Z}_\phi$ and $\{\alpha_i\}_{i=1}^N \subseteq \mathcal{S}$ be a set of transformation parameters such that for any i, the class probabilities satisfy

$$q(y_A|\,\phi(\boldsymbol{x},\, \alpha_i);\, \varepsilon) \geqslant p_A^{(i)} \geqslant p_B^{(i)} \geqslant \max_{y \neq y_A} q(y|\,\phi(\boldsymbol{x},\, \alpha_i);\, \varepsilon). \tag{E.284}$$

Then there exists a set $\Delta^* \subseteq \mathcal{Z}_\psi$ with the property that, if for any $\alpha \in \mathcal{S}$, $\exists \alpha_i$ with $\delta_{\boldsymbol{x}}(\alpha,\, \alpha_i) \in$

Δ^*, then it is guaranteed that

$$q(y_A|\,\phi(\boldsymbol{x},\,\alpha);\varepsilon) > \max_{y\neq y_A} q(y|\,\phi(\boldsymbol{x},\,\alpha);\varepsilon). \tag{E.285}$$

Proof. We prove the theorem by explicitly constructing a region Δ^* with the desired property by applying Theorem 7.1. For that purpose let $\delta \in \mathcal{Z}_\psi$ and denote by $\gamma_\delta\colon \mathcal{Z}_\psi \to \mathcal{Z}_\psi$ the resolving function of ψ, i.e.,

$$\psi(\psi(\boldsymbol{x},\,\delta),\,\delta') = \psi(\boldsymbol{x},\,\gamma_\delta(\delta')). \tag{E.286}$$

Let $\mathcal{P}_\gamma$ be the distribution of the random variable $\gamma := \gamma_\delta(\varepsilon)$ with density function f_γ and let

$$\underline{S}_t = \{z \in \mathcal{Z}_\psi\colon\ \Lambda(z) < t\}, \quad \overline{S}_t = \{z \in \mathcal{Z}_\psi\colon\ \Lambda(z) \leqslant t\} \tag{E.287}$$

$$\text{where} \quad \Lambda(z) = \frac{f_\gamma(z)}{f_\varepsilon(z)}. \tag{E.288}$$

Furthermore, recall the definition of the function $\zeta\colon \mathbb{R}_{\geqslant 0} \to [0,\,1]$ that is given by $t \mapsto \zeta(t) := \mathcal{P}_\varepsilon(\overline{S}_t)$ with generalized inverse $\zeta^{-1}(p) := \inf\{t \geqslant 0\colon\ \zeta(t) \geqslant p\}$. For $t \geqslant 0$ and the function $\xi\colon [0,\,1] \to [0,\,1]$ is given by by

$$\xi(p) := \sup\{\mathcal{P}_\gamma(S)\colon\ \underline{S}_{\zeta^{-1}(p)} \subseteq S \subseteq \overline{S}_\zeta^{-1}(p),\, \mathcal{P}_\varepsilon(S) \leqslant p\}. \tag{E.289}$$

By assumption, for every $i = 1,\,\ldots,\,n$, the ε-smoothed classifier g is $(p_A^{(i)},\,p_B^{(i)})$-confident at $\phi(x,\,\alpha_i)$. Identify $\Delta_i \subseteq \mathcal{Z}_\psi$ with the set of perturbations that satisfy the robustness condition (7.6) in Theorem 7.1, i.e.,

$$\Delta_i := \{\delta \in \mathcal{Z}_\psi\colon\ 1 - \xi(1 - p_B^{(i)}) < \xi(p_A^{(i)})\}. \tag{E.290}$$

Thus, by Theorem 7.1, we have that for any $\delta \in \Delta_i$

$$q(y_A|\,\psi(\phi(\boldsymbol{x},\,\alpha_i),\,\delta);\,\varepsilon) > \max_{y\neq y_A} q(y|\,\psi(\phi(\boldsymbol{x},\,\alpha_i),\,\delta);\,\varepsilon). \tag{E.291}$$

Finally, note that for the set

$$\Delta^* := \bigcap_{i=1}^{N} \Delta_i \tag{E.292}$$

it holds that, if for $\alpha \in \mathcal{S}$ there exists α_i with $\delta_{\boldsymbol{x}}(\alpha,\,\alpha_i) \in \Delta^*$, then in particular $\delta_{\boldsymbol{x}}(\alpha,\,\alpha_i) \in \Delta_i$

and hence, by Theorem 7.1 it is guaranteed that

$$q(y_A|\,\phi(\boldsymbol{x},\,\alpha);\,\varepsilon) = q(y_A|\,\psi(\phi(\boldsymbol{x},\,\alpha_i),\,\delta_x(\alpha,\,\alpha_i));\,\varepsilon) \tag{E.293}$$

$$> \max_{y \neq y_A} q(y|\,\psi(\phi(\boldsymbol{x},\,\alpha_i),\,\delta_x(\alpha,\,\alpha_i));\,\varepsilon) \tag{E.294}$$

$$= \max_{y \neq y_A} q(y|\,\phi(\boldsymbol{x},\,\alpha);\varepsilon) \tag{E.295}$$

what concludes the proof. QED.

Corollary 7.2 (restated)**.** Let $\psi(\boldsymbol{x},\,\delta) = \boldsymbol{x}+\delta$ and let $\varepsilon \sim \mathcal{N}(0,\,\sigma^2\boldsymbol{I}_d)$. Furthermore, let ϕ be a transformation with parameters in $\mathcal{Z}_\phi \subseteq \mathbb{R}^m$ and let $\mathcal{S} \subseteq \mathcal{Z}_\phi$ and $\{\alpha_i\}_{i=1}^N \subseteq \mathcal{S}$. Let $y_A \in \mathcal{Y}$ and suppose that for any i, the ε-smoothed classifier defined by $q(y|\,\boldsymbol{x};\varepsilon) := \mathbb{E}(p(y|\,\boldsymbol{x}+\varepsilon))$ has class probabilities that satisfy

$$q(y_A|\,\phi(\boldsymbol{x},\,\alpha_i);\,\varepsilon) \geqslant p_A^{(i)} \geqslant p_B^{(i)} \geqslant \max_{y \neq y_A} q(y|\,\phi(\boldsymbol{x},\,\alpha_i);\,\varepsilon). \tag{E.296}$$

Then it is guaranteed that $\forall \alpha \in \mathcal{S}\colon\ y_A = \arg\max_y q(y|\,\phi(\boldsymbol{x},\,\alpha);\varepsilon)$ if the maximum interpolation error

$$M_{\mathcal{S}} := \max_{\alpha \in \mathcal{S}} \min_{1 \leqslant i \leqslant N} \left\|\phi(\boldsymbol{x},\,\alpha) - \phi(\boldsymbol{x},\,\alpha_i)\right\|_2 \tag{E.297}$$

satisfies

$$M_{\mathcal{S}} < R := \frac{\sigma}{2} \min_{1 \leqslant i \leqslant N} \left(\Phi^{-1}\left(p_A^{(i)}\right) - \Phi^{-1}\left(p_B^{(i)}\right)\right). \tag{E.298}$$

Proof. Since the resolvable transformation ψ is given by $\psi(\boldsymbol{x},\,\delta) = \boldsymbol{x} + \delta$ we can write

$$\phi(\boldsymbol{x},\,\alpha) = \phi(\boldsymbol{x},\,\alpha_i) + \underbrace{\left(\phi(\boldsymbol{x},\,\alpha) - \phi(\boldsymbol{x},\,\alpha_i)\right)}_{=:\delta_{\boldsymbol{x}}(\alpha,\,\alpha_i)}. \tag{E.299}$$

Furthermore, by assumption $\varepsilon \sim \mathcal{N}(0,\,\sigma^2\boldsymbol{I}_d)$ and $g(\cdot;\,\varepsilon)$ is $(p_A^{(i)},\,p_B^{(i)})$-confident at $\phi(\boldsymbol{x},\,\alpha_i)$ for y_A and for all i. Thus, by Corollary E.1, if δ satisfies

$$\|\delta\|_2 < R_i := \frac{\sigma}{2}\left(\Phi^{-1}\left(p_A^{(i)}\right) - \Phi^{-1}\left(p_B^{(i)}\right)\right) \tag{E.300}$$

then it is guaranteed that $y_A = \arg\max_y q(y|\,\phi(\boldsymbol{x},\,\alpha_i) + \delta;\,\varepsilon)$. Let $\Delta_i := B_{R_i}(0)$ and notice that $R := \min_i R_i$ and thus

$$\bigcap_{i=1}^{N} B_{R_i}(0) = B_R(0) = \Delta^*. \tag{E.301}$$

To see that Δ^* has the desired property, consider

$$\forall \alpha \in \mathcal{S}\, \exists \alpha_i \colon\ \delta_{\boldsymbol{x}}(\alpha,\, \alpha_i) \in \Delta^* \tag{E.302}$$

$$\iff\ \forall \alpha \in \mathcal{S}\, \exists \alpha_i \colon\ \left\|\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i)\right\|_2 < R. \tag{E.303}$$

Since $R \leqslant R_i$ it follows that for $\delta_i = \phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i)$ it is guaranteed that

$$y_A = \arg\max_y q(y|\, \phi(\boldsymbol{x},\, \alpha_i) + \delta_i;\, \varepsilon) \tag{E.304}$$

$$= \arg\max_y q(y|\, \phi(\boldsymbol{x},\, \alpha);\, \varepsilon). \tag{E.305}$$

Thus, the set Δ^* has the desired property. In particular, since

$$\forall \alpha \in \mathcal{S}\, \exists \alpha_i \colon\ \left\|\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i)\right\|_2 < R \tag{E.306}$$

$$\iff \max_{\alpha \in \mathcal{S}} \min_{1 \leqslant i \leqslant N} \left\|\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i)\right\|_2 < R \tag{E.307}$$

the statement follows. QED.

Corollary 7.5 (restated)**.** Let $\psi_B(\boldsymbol{x},\, \delta,\, b) = x + \delta + b \cdot \mathbb{1}_d$ and let $\varepsilon \sim \mathcal{N}(0,\, \sigma^2 \boldsymbol{I}_d)$, $\varepsilon_b \sim \mathcal{N}(0,\, \sigma_b^2)$. Furthermore, let ϕ be a transformation with parameters in $\mathcal{Z}_\phi \subseteq \mathbb{R}^m$ and let $\mathcal{S} \subseteq \mathcal{Z}_\phi$ and $\{\alpha_i\}_{i=1}^N \subseteq \mathcal{S}$. Let $y_A \in \mathcal{Y}$ and suppose that for any i, the $(\varepsilon, \varepsilon_b)$-smoothed classifier $q(y|\, \boldsymbol{x};\, \varepsilon, \varepsilon_b) := \mathbb{E}(p(y|\, \psi_B(\boldsymbol{x},\, \varepsilon,\, \varepsilon_b))$ satisfies

$$q(y_A|\, \boldsymbol{x};\, \varepsilon, \varepsilon_b) \geqslant p_A^{(i)} > p_B^{(i)} \geqslant \max_{y \neq y_A} q(y|\, \boldsymbol{x};\, \varepsilon, \varepsilon_b). \tag{E.308}$$

for each i. Let

$$R := \frac{\sigma}{2} \min_{1 \leqslant i \leqslant N} \left(\Phi^{-1}\left(p_A^{(i)}\right) - \Phi^{-1}\left(p_B^{(i)}\right) \right) \tag{E.309}$$

Then, $\forall \alpha \in \mathcal{S}$ and $\forall b \in [-b_0, b_0]$ it is guaranteed that $y_A = \arg\max_y q(y|\, \phi(\boldsymbol{x}, \alpha) + b \cdot \mathbb{1}_d;\, \varepsilon, \varepsilon_b)$ as long as

$$R > \sqrt{M_{\mathcal{S}}^2 + \frac{\sigma^2}{\sigma_b^2} b_0^2}, \tag{E.310}$$

where $M_{\mathcal{S}}$ is defined as in Corollary 7.2.

Proof. Since the resolvable transformation ψ_B is given by

$$\psi_B(\boldsymbol{x},\, \delta,\, b) = \boldsymbol{x} + \delta + b \cdot \mathbb{1}_d, \tag{E.311}$$

we can write

$$\phi(\boldsymbol{x},\,\alpha) + b \cdot \mathbb{1}_d = \phi(\boldsymbol{x},\,\alpha_i) + \underbrace{\big(\phi(\boldsymbol{x},\,\alpha) - \phi(\boldsymbol{x},\,\alpha_i)\big) + b \cdot \mathbb{1}_d}_{=:\delta_{\boldsymbol{x}}((\alpha,b),\,(\alpha_i,0))}. \tag{E.312}$$

Furthermore, by assumption $\varepsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I}_d)$, $\varepsilon_b \sim \mathcal{N}(0, \sigma_b^2)$ and $g(\cdot;\, \varepsilon, \varepsilon_b)$ is $(p_A^{(i)}, p_B^{(i)})$-confident at $\phi(\boldsymbol{x}, \alpha_i)$ for y_A and all i. Thus, by Corollary E.1, if δ and b satisfy

$$\sqrt{\frac{\|\delta\|_2^2}{\sigma^2} + \frac{b^2}{\sigma_b^2}} < \frac{1}{2}\left(\Phi^{-1}(p_A^{(i)}) - \Phi^{-1}(p_B^{(i)})\right), \tag{E.313}$$

then it is guaranteed that

$$y_A = \arg\max_y q(y|\phi(\boldsymbol{x}, \alpha_i) + \delta + b \cdot \mathbb{1}_d;\, \varepsilon, \varepsilon_b). \tag{E.314}$$

Let

$$R_i := \frac{\sigma}{2}\left(\Phi^{-1}(p_A^{(i)}) - \Phi^{-1}(p_B^{(i)})\right) \tag{E.315}$$

and note that without loss of generality we can assume that $R_i > \sigma/\sigma_b b_0$, because otherwise the robustness condition is violated. Rearranging terms in (E.313) leads to the condition

$$\|\delta\|_2 < \sqrt{R_i^2 - \frac{\sigma^2}{\sigma_b^2} b^2} \tag{E.316}$$

that can be turned into a sufficient robustness condition holding for any $b \in [-b_0,\, b_0]$ simultaneously

$$\|\delta\|_2 < \sqrt{R_i^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2} \tag{E.317}$$

Note that, without loss of generality For each i let Δ_i be the set defined as

$$\Delta_i := \left\{ \delta + b \cdot \mathbf{1}_d \in \mathbb{R}^d \colon\ \|\delta\|_2 < \sqrt{R_i^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2},\ |b| \leqslant b_0 \right\} \tag{E.318}$$

and note that

$$\Delta^* := \bigcap_{i=1}^{N} \Delta_i = \left\{ \delta + b \cdot \mathbf{1}_d \in \mathbb{R}^d \colon\ \|\delta\|_2 < \sqrt{R^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2},\ |b| \leqslant b_0 \right\} \tag{E.319}$$

with $R := \min_i R_i$. Clearly, if $\forall \alpha \in \mathcal{S},\ \forall b \in [-b_0,\, b_0]\, \exists i$ such that

$$\delta_{\boldsymbol{x}}((\alpha, b),\, (\alpha_i, 0)) \in \Delta^* \tag{E.320}$$

then it is guaranteed that

$$y_A = \arg\max_y q(y|\, \phi(\boldsymbol{x}, \alpha_i) + \delta_{\boldsymbol{x}}((\alpha, b),\, (\alpha_i, 0))) \tag{E.321}$$

$$= \arg\max_y q(y|\, \phi(\boldsymbol{x},\, \alpha) + b \cdot \boldsymbol{I}_d). \tag{E.322}$$

We can thus reformulate the robustness condition as

$$\forall \alpha \in \mathcal{S},\ \forall b \in [-b_0,\, b_0]\, \exists i \text{ s.t. } \delta_{\boldsymbol{x}}((\alpha, b),\, (\alpha_i, 0)) \in \Delta^* \tag{E.323}$$

$$\iff \forall \alpha \in \mathcal{S},\ \forall b \in [-b_0,\, b_0]\, \exists i \text{ s.t. } \|\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i)\|_2 < \sqrt{R^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2}$$

$$\iff \max_{\alpha \in \mathcal{S}} \min_{1 \leqslant i \leqslant N} \|\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i)\|_2 < \sqrt{R^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2}$$

that, written in terms of the maximum ℓ_2 interpolation error $M_{\mathcal{S}}$, is equivalent to

$$R > \sqrt{M_{\mathcal{S}}^2 + \frac{\sigma^2}{\sigma_b^2} b_0^2} \tag{E.324}$$

what concludes the proof. QED.

Corollary 7.6 (restated)**.** Under the same setting as in Corollary 7.5, for $\forall \alpha \in \mathcal{S},\ \forall b \in [-b_0, b_0]$ and $\forall \delta \in \mathbb{R}^d$ such that $\|\delta\|_2 \leqslant r$, it is guaranteed that $y_A = \arg\max_k q(y|\, \phi(\boldsymbol{x}, \alpha) + b \cdot \mathbb{1}_d + \delta;\, \varepsilon, \varepsilon_d)$ as long as

$$R > \sqrt{(M_{\mathcal{S}} + r)^2 + \frac{\sigma^2}{\sigma_b^2} b_0^2}, \tag{E.325}$$

where $M_{\mathcal{S}}$ is defined as in Corollary 7.2.

Proof. Note that we can write the transformed input as

$$\begin{aligned} &\phi(\boldsymbol{x}, \alpha) + b \cdot \boldsymbol{I}_d + \delta \\ &\quad = \phi(\boldsymbol{x}, \alpha_i) + \underbrace{(\phi(\boldsymbol{x}, \alpha) - \phi(\boldsymbol{x}, \alpha_i) + \delta) + b \cdot \mathbb{1}_d}_{=: \delta_{\boldsymbol{x}}((\alpha, b, \delta),\, (\alpha_i, 0, 0))}. \end{aligned} \tag{E.326}$$

Since we use the same smoothing protocol as in Corollary 7.5, the general proof idea is

similar to Corollary 7.5 — we use the same resolvable transformation ψ_B and define the same set Δ_i, namely

$$\Delta_i := \left\{ \delta' + b \cdot \mathbf{1}_d + \delta \in \mathbb{R}^d \colon \|\delta' + \delta\|_2 < \sqrt{R_i^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2},\, |b| \leqslant b_0,\, \|\delta\|_2 \leqslant r \right\}. \tag{E.327}$$

and set

$$\Delta^* := \bigcap_{i=1}^{N} \Delta_i = \left\{ \delta' + b \cdot \mathbf{1}_d + \delta \in \mathbb{R}^d \colon \|\delta' + \delta\|_2 < \sqrt{R^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2},\, |b| \leqslant b_0,\, \|\delta\|_2 \leqslant r \right\} \tag{E.328}$$

with $R := \min_i R_i$. Clearly, if $\forall \alpha \in \mathcal{S}$, $\forall b \in [-b_0,\, b_0]$, $\|\delta\|_2 \leqslant r$ $\exists i$ such that

$$\delta_{\boldsymbol{x}}((\alpha, b,\, \delta),\, (\alpha_i, 0)) \in \Delta^* \tag{E.329}$$

then it is guaranteed that

$$y_A = \arg\max_y q(y|\, \phi(\boldsymbol{x}, \alpha_i) + \delta_{\boldsymbol{x}}((\alpha, b),\, (\alpha_i, 0))) \tag{E.330}$$

$$= \arg\max_y q(y|\, \phi(\boldsymbol{x},\, \alpha) + b \cdot \mathbb{1}_d). \tag{E.331}$$

We can thus reformulate the robustness condition as

$$\begin{aligned} &\forall \alpha \in \mathcal{S},\, \forall b \in [-b_0,\, b_0],\, \|\delta\|_2 \leqslant r\, \exists i \,\text{s.t.}\, \delta_{\boldsymbol{x}}((\alpha, b,\, \delta),\, (\alpha_i,\, 0,\, 0)) \in \Delta^* \\ \iff &\forall \alpha \in \mathcal{S},\, \forall b \in [-b_0,\, b_0],\, \|\delta\|_2 \leqslant r\, \exists i \text{ s.t. } \|\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i) + \delta\|_2 < \sqrt{R^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2} \\ \iff &\max_{\alpha \in \mathcal{S}} \min_{1 \leqslant i \leqslant N} \|\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i) + \delta\|_2 < \sqrt{R^2 - \frac{\sigma^2}{\sigma_b^2} b_0^2}. \end{aligned} \tag{E.332}$$

Note that by the triangle inequality have

$$\max_{\alpha \in \mathcal{S}} \min_{1 \leqslant i \leqslant N} \|\phi(\boldsymbol{x},\, \alpha) - \phi(\boldsymbol{x},\, \alpha_i) + \delta\|_2 \leqslant M_{\mathcal{S}} + \|\delta\|_2 \leqslant M_{\mathcal{S}} + r \tag{E.333}$$

and thus, robustness is implied by

$$R > \sqrt{(M_{\mathcal{S}} + r)^2 + \frac{\sigma^2}{\sigma_b^2} b_0^2} \tag{E.334}$$

what concludes the proof.

QED.

E.5 TRANSFORMATION DETAILS

In this section, we provide detailed definitions of rotation and scaling transformation.

E.5.1 Bilinear Interpolation

Let $\Omega_K := \{0, \ldots, K-1\}$ and $\Omega := [0, W-1] \times [0, H-1]$. We define bilinear interpolation to be the map $Q\colon \mathbb{R}^{K\times W\times H} \to L^2(\Omega_K \times \mathbb{R}^2, \mathbb{R})$, $x \mapsto Q(x) =: Q_x$ where Q_x is given by

$$(k, i, j) \mapsto Q_x(k, i, j) := \begin{cases} 0 & (i, j) \notin \Omega \\ x_{k,i,j} & (i, j) \in \Omega \cap \mathbb{N}^2 \\ \tilde{x}_{k,i,j} & (i, j) \in \Omega \backslash \mathbb{N}^2. \end{cases} \tag{E.335}$$

and where

$$\begin{aligned} \tilde{x}_{k,i,j} := \big(1 - (i - \lfloor i \rfloor)\big) \cdot \Big(\big(1 - (j - \lfloor j \rfloor)\big) \cdot x_{k,\lfloor i \rfloor,\lfloor j \rfloor} + (j - \lfloor j \rfloor) \cdot x_{k,\lfloor i \rfloor,\lfloor j \rfloor+1}\Big) \\ + (i - \lfloor i \rfloor) \cdot \Big(\big(1 - (j - \lfloor j \rfloor)\big) \cdot x_{k,\lfloor i \rfloor+1,\lfloor j \rfloor} + (j - \lfloor j \rfloor) \cdot x_{k,\lfloor i \rfloor+1,\lfloor j \rfloor+1}\Big). \end{aligned} \tag{E.336}$$

E.5.2 Rotation

The rotation transformation is denoted as $\phi_R\colon \mathbb{R}^{K\times W\times H} \times \mathbb{R} \to \mathbb{R}^{K\times W\times H}$ and acts on an image in three steps that we will highlight in greater detail. First, it rotates the image by α degrees counter-clockwise. After rotation, pixel values are determined using bilinear interpolation (E.335). Finally, we apply black padding to all pixels (i, j) whose ℓ_2-distance to the center pixel is larger than half of the length of the shorter side, and denote this operation by P. Let c_W and c_H be the center pixels

$$c_W := \frac{W-1}{2}, \qquad c_H := \frac{H-1}{2}. \tag{E.337}$$

and

$$d_{i,j} = \sqrt{(i - c_W)^2 + (j - c_H)^2}, \quad g_{i,j} = \operatorname{arctan2}\left(j - c_H, i - c_W\right). \tag{E.338}$$

We write $\tilde{\phi}_R$ for the rotation transformation before black padding and decompose ϕ_R as $\phi_R = P \circ \tilde{\phi}_R$, where $\tilde{\phi}_R \colon \mathbb{R}^{K\times W\times H} \times \mathbb{R} \to \mathbb{R}^{K\times W\times H}$ is defined by

$$\tilde{\phi}_R(\boldsymbol{x},\, \alpha)_{k,i,j} := Q_x(k,\, c_W + d_{i,j}\, \cos(g_{i,j} - \alpha),\, c_H + d_{i,j}\, \sin(g_{i,j} - \alpha)) \tag{E.339}$$

and $P \colon \mathbb{R}^{K\times W\times H} \to \mathbb{R}^{K\times W\times H}$ by

$$f \mapsto P(f)_{k,i,j} = \begin{cases} f(k,\, i,\, j) & d_{i,j} < \min\{c_W,\, c_H\} \\ 0 & \text{otherwise} \end{cases}. \tag{E.340}$$

The rotation transformation in practice may use different padding mechanisms. For example, the rotation in the physical world may fill in boundary pixels with real elements captured by the camera. We remark that our TSS against the transformation ϕ_R implies the defense against rotation with *any other* padding mechanisms, because we first apply black-padding P to any such rotated input and then feed into TSS models so that TSS models always receive black-padded inputs.

E.5.3 Scaling

The scaling transformation is denoted as $\phi_S \colon \mathbb{R}^{K\times W\times H} \times \mathbb{R} \to \mathbb{R}^{K\times W\times H}$. Similar as for rotations, ϕ_S acts on an image in three steps. First, it stretches height and width by a fixed ratio $\alpha \in \mathbb{R}$. Second, we determine missing pixel values with bilinear interpolation. Finally, we apply black padding to regions with missing pixel values if the image is scaled by a factor smaller than 1. Let c_W and c_H be the center pixels

$$c_W := \frac{W-1}{2}, \qquad c_H := \frac{H-1}{2}. \tag{E.341}$$

We notice that black padding is naturally applied during bilinear interpolation in cases where the scaling factor is smaller than 1 (that is, when we make images smaller). We can thus write the scaling operation as $\phi_S \colon \mathbb{R}^{K\times W\times H} \times \mathbb{R}_{>0} \to \mathbb{R}^{K\times W\times H}$, $(\boldsymbol{x},\, \alpha) \mapsto \phi(\boldsymbol{x},\, \alpha)$ where

$$\phi_S(\boldsymbol{x},\, \alpha)_{k,i,j} := Q_x\left(k,\, c_W + \frac{i - c_W}{\alpha},\, c_H + \frac{j - c_H}{\alpha}\right). \tag{E.342}$$

When the scaling transformation in practice uses different padding mechanisms, we can simply apply black padding to the outer pixels during preprocessing. For example, if we know the semantic attacker could choose 0.7 as the smallest scaling ratio, we can apply black padding to all pixels that are out of canvas after 0.7 scaling. Therefore, we overwrite

all different padding mechanisms and ensure the generalizability. As a trade-off, the classifier has a narrower reception field that affects the clean accuracy.

E.6 PROOFS FOR INTERPOLATION BOUND COMPUTATION

In this section we state the proofs for the theoretical results governing our approach to certifying rotations and scaling transformations using randomized smoothing. We first define the maximum ℓ_2 interpolation error. First, let us recall the following definitions from the main part of this chapter.

Definition E.3 (ℓ_2 interpolation error)**.** Let $\boldsymbol{x} \in \mathcal{X}$, $\phi\colon \mathcal{X} \times \mathcal{Z} \to \mathcal{X}$ a transformation, $\mathcal{S} = [a, b]$, $N \in \mathbb{N}$ and suppose $\{\alpha_i\}_{i=1}^N \subseteq \mathcal{S}$. The maximum ℓ_2 interpolation error is defined as

$$M_{\mathcal{S}} := \max_{a \leqslant \alpha \leqslant b} \min_{1 \leqslant i \leqslant N} \|\phi(\boldsymbol{x}, \alpha) - \phi(\boldsymbol{x}, \alpha_i)\|_2. \tag{E.343}$$

Definition 7.4 (restated)**.** For pixels $(i, j) \in \Omega$, we define the grid pixel generator G_{ij} as

$$G_{ij} := \{(i, j), (i+1, j), (i, j+1), (i+1, j+1)\}. \tag{E.344}$$

Definition 7.5 (restated)**.** We define the operator that extracts the channel-wise maximum pixel wise on a grid $S \subseteq \Omega$ as the map $\overline{m}\colon \mathbb{R}^{K \times W \times H} \times \{0, \ldots, K-1\} \times 2^{\Omega} \to \mathbb{R}$ with

$$\overline{m}(\boldsymbol{x}, k, S) := \max_{(i,j) \in S} \left(\max_{(r,s) \in G_{ij}} x_{k,r,s} \right) \tag{E.345}$$

Definition 7.6 (restated)**.** We define the operator that extracts the channel-wise maximum change in color on a grid $S \subseteq \Omega$ as the map $m_{\Delta}\colon \mathbb{R}^{K \times W \times H} \times \{0, \ldots, K-1\} \times 2^{\Omega} \to \mathbb{R}$ with

$$m_{\Delta}(\boldsymbol{x}, k, S) := \max_{(i,j) \in S} \left(\max_{(r,s) \in G_{ij}} x_{k,r,s} - \min_{(r,s) \in G_{ij}} x_{k,r,s} \right) \tag{E.346}$$

The following auxiliary lemma is used for both rotation and scaling:

Lemma E.4. Let $\boldsymbol{x} \in \mathbb{R}^{K \times W \times H}$, $-\infty < t_1 < t_2 < \infty$ and suppose $\rho\colon [t_1, t_2] \to [0, W-1] \times [0, H-1]$ is a curve of class C^1. Let

$$\psi_k\colon [t_1, t_2] \to \mathbb{R}, \;\; \psi_k(t) := Q_{\boldsymbol{x}}(k, \rho_1(t), \rho_2(t)) \tag{E.347}$$

where $k \in \Omega_K$ and $Q_{\boldsymbol{x}}$ denotes bilinear interpolation. Then ψ_k is L_k-Lipschitz continuous with constant

$$L_k = \max_{t \in [t_1, t_2]} \left(\sqrt{2} \left\| \dot{\rho}(t) \right\|_2 \cdot m_\Delta(\boldsymbol{x},\, k,\, \lfloor \rho(t) \rfloor) \right) \tag{E.348}$$

Proof. Note that the function $t \mapsto \lfloor \rho(t) \rfloor$ is piecewise constant and let $t_1 =: u_1 < u_2 < \ldots < u_{N_0} := t_2$ such that $\lfloor \rho(t) \rfloor$ is constant on $[u_i,\, u_{i+1})$ for all $1 \leqslant i \leqslant N_0 - 1$ and $\dot{\bigcup}_{i=1}^{N_0} [u_i,\, u_{i+1}) = [t_1,\, t_2)$. We notice that ψ_k is a continuous real-valued function since it is the composition of the continuous $Q_{\boldsymbol{x}}$ and C^1-curve ρ. L_k-Lipschitz continuity on $[t_1,\, t_2)$ thus follows if we show that ψ_k is L_k-Lipschitz on each interval in the partition. For that purpose, let $1 \leqslant i \leqslant N_0$ be arbitrary and fix some $t \in [u_i,\, u_{i+1})$. Let $(w,\, h) := \lfloor \rho(t) \rfloor$ and $\gamma(t) := \rho(t) - \lfloor \rho(t) \rfloor$ and notice that $\gamma(t) \in [0,\, 1)^2$. Let

$$V_1 := x_{k,w,h},\; V_2 := x_{k,w,h+1}, \tag{E.349}$$

$$V_3 := x_{k,w+1,h},\; V_4 := x_{k,w+1,h+1}, \tag{E.350}$$

Then, for any $u \in [u_i,\, u_{i+1})$

$$\psi_k(u) = Q_{\boldsymbol{x}}(k,\, \rho_1(u),\, \rho_2(u)) \tag{E.351}$$

$$= (1 - \gamma_1(u)) \cdot ((1 - \gamma_2(u)) \cdot V_1 + \gamma_2(u) \cdot V_2) + \gamma_1(u) \cdot ((1 - \gamma_2(u) \cdot V_3 + \gamma_2(u) \cdot V_4). \tag{E.352}$$

Let $m_\Delta := m_\Delta(x,\, k \lfloor \rho(t) \rfloor)$ and notice that by definition

$$m_\Delta = \max_i V_i - \min_i V_i \tag{E.353}$$

and in particular

$$\left| V_i - V_j \right| \leqslant m_\Delta \quad \forall\, i, j. \tag{E.354}$$

Since V_i is constant for each i and γ is differentiable, ψ_k is differentiable on $[u_i,\, u_{i+1})$ and hence

$$\dot{\psi}_k(u) = (\dot{\gamma}_1(u)\gamma_2(u) + \gamma_1(u)\dot{\gamma}_2(u))(V_1 - V_2 - V_3 + V_4) \tag{E.355}$$

$$+ \dot{\gamma}_1(u)(V_3 - V_1) + \dot{\gamma}_2(u)(V_2 - V_1). \tag{E.356}$$

Note that the derivative $\dot{\psi}_k$ is linear in γ_1 and γ_2 and hence its extreme values are bounded when evaluated at extreme values of γ, that is $(\gamma_1,\, \gamma_2) \in \{0,\, 1\}^2$. We treat each case sepa-

rately:

- $\gamma_1 = \gamma_2 = 0$. Then,

$$\left|\dot{\psi}_k\right| \leqslant \left|\dot{\gamma}_1(V_3 - V_1) + \dot{\gamma}_2(V_2 - V_1)\right| \tag{E.357}$$

$$\leqslant |\dot{\gamma}_1| \cdot |V_3 - V_1| + |\dot{\gamma}_2| \cdot |V_2 - V_1| \leqslant m_\Delta(|\dot{\gamma}_1| + |\dot{\gamma}_2|) \tag{E.358}$$

- $\gamma_1 = \gamma_2 = 1$. Then,

$$\left|\dot{\psi}_k\right| \leqslant \left|\dot{\gamma}_1(V_4 - V_2) + \dot{\gamma}_2(V_4 - V_3)\right| \tag{E.359}$$

$$\leqslant |\dot{\gamma}_1| \cdot |V_4 - V_2| + |\dot{\gamma}_2| \cdot |V_4 - V_3| \leqslant m_\Delta(|\dot{\gamma}_1| + |\dot{\gamma}_2|) \tag{E.360}$$

- $\gamma_1 = 0$, $\gamma_2 = 1$. Then,

$$\left|\dot{\psi}_k\right| \leqslant \left|\dot{\gamma}_1(V_4 - V_2) + \dot{\gamma}_2(V_2 - V_1)\right| \tag{E.361}$$

$$\leqslant |\dot{\gamma}_1| \cdot |V_4 - V_2| + |\dot{\gamma}_2| \cdot |V_2 - V_1| \leqslant m_\Delta(|\dot{\gamma}_1| + |\dot{\gamma}_2|) \tag{E.362}$$

- $\gamma_1 = 1$, $\gamma_2 = 0$. Then,

$$\left|\dot{\psi}_k\right| \leqslant \left|\dot{\gamma}_1(V_3 - V_1) + \dot{\gamma}_2(V_4 - V_3)\right| \tag{E.363}$$

$$\leqslant |\dot{\gamma}_1| \cdot |V_3 - V_1| + |\dot{\gamma}_2| \cdot |V_4 - V_3| \leqslant m_\Delta(|\dot{\gamma}_1| + |\dot{\gamma}_2|) \tag{E.364}$$

Hence, for any $u \in [u_i,\, u_{i+1})$, the modulus of the derivative is bounded by $m_\Delta(|\dot{\gamma}_1| + |\dot{\gamma}_2|)$. We can further bound this by observing the following connection between ℓ_1 and ℓ_2 distance

$$\forall\, \boldsymbol{x} \in \mathbb{R}^n : \quad \|x\|_1 = \left|\langle |\boldsymbol{x}|\,,\, \mathbf{1}\rangle\right| \leqslant \|\boldsymbol{x}\|_2 \, \|\mathbf{1}\|_2 = \sqrt{n}\, \|\boldsymbol{x}\|_2 \tag{E.365}$$

and hence $\forall\, u \in [u_i,\, u_{i+1})$

$$\left|\psi_k(u)\right| \leqslant m_\Delta \left\|\dot{\gamma}(u)\right\|_1 \leqslant m_\Delta \sqrt{2} \left\|\dot{\gamma}(u)\right\|_2 = m_\Delta \sqrt{2} \left\|\dot{\rho}(u)\right\|_2 . \tag{E.366}$$

Since ψ_k is differentiable on $[u_i,\, u_{i+1})$, its Lipschitz constant is bounded by the maximum absolute value of its derivative. Hence

$$\max_{u \in [u_i,\, u_{i+1})} m_\Delta \sqrt{2} \left\|\dot{\rho}(u)\right\|_2 = \max_{u \in [u_i,\, u_{i+1})} m_\Delta(\boldsymbol{x},\, k,\, \lfloor \rho(u) \rfloor) \sqrt{2} \left\|\dot{\rho}(u)\right\|_2 \tag{E.367}$$

$$\leqslant \max_{u \in [t_1,\, t_2)} m_\Delta(\boldsymbol{x},\, k,\, \lfloor \rho(u) \rfloor) \sqrt{2} \left\|\dot{\rho}(u)\right\|_2 = L_k \tag{E.368}$$

is a Lipschitz constant for ψ_k on $[u_i,\, u_{i+1})$. Note that L_k does not depend on i. Furthermore, i was chosen arbitrarily and hence L_k is a Lipschitz constant for ψ_k on $[t_1,\, t_2)$ and due to continuity on $[t_1,\, t_2]$, concluding the proof. QED.

E.6.1 Rotation

Lemma 7.4 (restated)**.** Let $\boldsymbol{x} \in \mathbb{R}^{K\times W\times H}$ be a K-channel image and let $\phi_R = P \circ I \circ \tilde{\phi}_R$ be the rotation transformation. Then, a global Lipschitz constant L for the functions $\{g_i\}_{i=1}^N$ is given by

$$L_r = \max_{1\leqslant i\leqslant N-1} \sum_{k=0}^{K-1} \sum_{r,s\in V} 2d_{r,s} \cdot m_\Delta(\boldsymbol{x}, k, \mathcal{P}_{r,s}^{(i)}) \cdot \overline{m}(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}^{(i)}) \tag{E.369}$$

where $V = \big\{(r,s) \in \mathbb{N}^2 |\, d_{r,s} < \frac{1}{2}(\min\{W,H\} - 1)\big\}$. The set $\mathcal{P}_{r,s}^{(i)}$ is given by all integer grid pixels that are covered by the trajectory of source pixels of (r,s) when rotating from angle α_i to α_{i+1}.

Proof. Recall that ϕ_R acts on images $x \in \mathbb{R}^{K\times W\times H}$ and that g_i is defined as

$$g_i(\alpha) = \big\|\phi_R(\boldsymbol{x},\, \alpha) - \phi_R(\boldsymbol{x},\, \alpha_i)\big\|_2^2 = \sum_{k=0}^{K-1}\sum_{r=0}^{W-1}\sum_{s=0}^{H-1} \big(\phi_R(\boldsymbol{x},\, \alpha)_{k,r,s} - \phi_R(\boldsymbol{x},\, \alpha_i)_{k,r,s}\big)^2 \tag{E.370}$$

Let c_W and c_H denote the center pixels

$$c_W := \frac{W-1}{2}, \qquad c_H := \frac{H-1}{2}. \tag{E.371}$$

and recall the following quantities from the definition of ϕ_R (Appendix E.5.2):

$$d_{r,s} = \sqrt{(r-c_W)^2 + (s-c_H)^2}, \quad g_{r,s} = \arctan 2\,(s-c_H,\, r-c_W) \tag{E.372}$$

Note that

$$d_{r,s} \geqslant \min\{c_W,\, c_H\} \;\;\Rightarrow\;\; \phi_R(\boldsymbol{x},\, \alpha)_{k,r,s} = 0. \tag{E.373}$$

We thus only need to consider pixels that lie inside the centered disk. We call the collection of such pixels *valid* pixels, denoted by V:

$$V := \big\{(r,\, s) \in \mathbb{N}^2 \mid d_{r,s} < \min\{c_W,\, c_H\}\big\}. \tag{E.374}$$

Let $f_1^{r,s}\colon \mathbb{R} \to \mathbb{R}$ and $f_2^{r,s}\colon \mathbb{R} \to \mathbb{R}$ be functions defined as

$$f_1^{r,s}(\alpha) = c_W + d_{r,s}\cos(g_{r,s} - \alpha), f_2^{r,s}(\alpha) = c_H + d_{r,s}\sin(g_{r,s} - \alpha). \tag{E.375}$$

Then for any valid pixel $(r,\, s) \in V$, the value of the rotated image $\phi_R(x,\, \alpha)$ is given by

$$\phi_R(\boldsymbol{x},\, \alpha)_{k,r,s} = Q_x(k,\, f_1^{r,s}(\alpha),\, f_2^{r,s}(\alpha)) \tag{E.376}$$

where Q_x denotes bilinear interpolation. We define the shorthand

$$g_i^{k,r,s}(\alpha) := \left(\phi_R(\boldsymbol{x},\, \alpha)_{k,r,s} - \phi_R(\boldsymbol{x},\, \alpha_i)_{k,r,s}\right)^2 \tag{E.377}$$

and denote by $L_i^{k,r,s}$ and $L_{i+1}^{k,r,s}$ the Lipschitz constants of $g_i^{k,r,s}$ and $g_{i+1}^{k,r,s}$ on $[\alpha_i,\, \alpha_{i+1}]$. We can write (E.370) as

$$\begin{aligned} g_i(\alpha) &= \sum_{k=0}^{K-1} \sum_{(r,\, s)\in V} g_i^{k,r,s}(\alpha), \\ g_{i+1}(\alpha) &= \sum_{k=0}^{K-1} \sum_{(r,\, s)\in V} g_{i+1}^{k,r,s}(\alpha) \end{aligned} \tag{E.378}$$

and note that Lipschitz constants of g_i and g_{i+1} on $[\alpha_i,\, \alpha_{i+1}]$ are given by

$$\max_{c,\, d\in[\alpha_i,\, \alpha_{i+1}]} \frac{\left|g_i(c) - g_i(d)\right|}{|c-d|} \leqslant \left(\sum_{k=0}^{K-1} \sum_{(r,\, s)\in V} L_i^{k,r,s}\right) =: L_i \tag{E.379}$$

$$\max_{c,\, d\in[\alpha_i,\, \alpha_{i+1}]} \frac{\left|g_{i+1}(c) - g_{i+1}(d)\right|}{|c-d|} \leqslant \left(\sum_{k=0}^{K-1} \sum_{(r,\, s)\in V} L_{i+1}^{k,r,s}\right) =: L_{i+1} \tag{E.380}$$

We can hence determine L according to equation (7.29) as

$$L = \max_i \left\{\max\left\{L_i,\, L_{i+1}\right\}\right\}. \tag{E.381}$$

Without loss of generality, consider $L_i^{k,r,s}$ and note that

$$\max_{c,d\in[\alpha_i,\, \alpha_{i+1}]} \left|\frac{g_i^{k,r,s}(c) - g_i^{k,r,s}(d)}{c-d}\right| \tag{E.382}$$

$$= \max_{c,d\in[\alpha_i,\, \alpha_{i+1}]} \left|\frac{\phi_R(\boldsymbol{x},\, c)_{k,r,s} - \phi_R(\boldsymbol{x},\, d)_{k,r,s}}{c-d}\right| \cdot \left|\phi_R(\boldsymbol{x},\, c)_{k,r,s} + \phi_R(\boldsymbol{x},\, d)_{k,r,s} - 2\phi_R(\boldsymbol{x},\, \alpha_i)_{k,r,s}\right| \tag{E.383}$$

$$\leqslant \max_{c,d\in[\alpha_i,\,\alpha_{i+1}]} \underbrace{\left|\frac{\phi_R(\boldsymbol{x},\,c)_{k,r,s}-\phi_R(\boldsymbol{x},\,d)_{k,r,s}}{c-d}\right|}_{\text{(I)}} \cdot 2 \max_{\theta\in[\alpha_i,\,\alpha_{i+1}]} \underbrace{\left|\phi_R(\boldsymbol{x},\,\theta)_{k,r,s}-\phi_R(\boldsymbol{x},\,\alpha_i)_{k,r,s}\right|}_{\text{(II)}}. \tag{E.384}$$

To compute a Lipschitz constant for $g_i^{k,r,s}$ on the interval $[\alpha_i,\,\alpha_{i+1}]$ we thus only need to compute a Lipschitz constant for $\phi_R(\boldsymbol{x},\,\cdot)$ on $[\alpha_i,\,\alpha_{i+1}]$ and an upper bound on (II). For that purpose, note that ϕ_R takes only positive values and consider

$$\text{(II)} \leqslant \max_{\theta\in[\alpha_i,\,\alpha_{i+1}]} \left\{\phi_R(\boldsymbol{x},\,\theta)_{k,r,s},\,\phi_R(\boldsymbol{x},\,\alpha_i)_{k,r,s}\right\} = \max_{\theta\in[\alpha_i,\,\alpha_{i+1}]} \phi_R(\boldsymbol{x},\,\theta)_{k,r,s} \tag{E.385}$$

Notice that now both $L_i^{k,r,s}$ and $L_{i+1}^{k,r,s}$ share the same upper bound. Recall (E.376), i.e.,

$$\phi_R(\boldsymbol{x},\,\theta)_{k,r,s} = Q_{\boldsymbol{x}}(k,\,f_1^{r,s}(\theta),\,f_2^{r,s}(\theta)). \tag{E.386}$$

Now, we upper bound (E.385) by finding all integer grid pixels that are covered by the trajectory $(f_1^{r,s}(\theta),\,f_2^{r,s}(\theta))$. Specifically, let

$$\mathcal{P}_{r,s}^{(i)} := \bigcup_{\theta\in[\alpha_i,\,\alpha_{i+1}]} \left(\lfloor f_1^{r,s}(\theta)\rfloor,\,\lfloor f_2^{r,s}(\theta)\rfloor\right). \tag{E.387}$$

Since ϕ_R is interpolated from integer pixels, we can consider the maximum over $\mathcal{P}_{r,s}^{(i)}$ in order to upper bound (E.385):

$$\max_{\theta\in[\alpha_i,\,\alpha_{i+1}]} \phi_R(\boldsymbol{x},\,\theta)_{k,r,s} = \max_{\theta\in[\alpha_i,\,\alpha_{i+1}]} Q_x(k,\,f_1^{r,s}(\theta),\,f_2^{r,s}(\theta)) \tag{E.388}$$

$$\leqslant \max_{(i,j)\in\mathcal{P}_{r,s}} \max\left\{x(k,\,i,\,j),\,x(k,\,i+1,\,j),\,x(k,\,i,\,j+1),\,x(k,\,i+1,\,j+1)\right\} \tag{E.389}$$

$$= \bar{m}(\boldsymbol{x},\,k,\,\mathcal{P}_{r,s}^{(i)}). \tag{E.390}$$

We now have to find an upper bound of (I), that is, a Lipschitz constant of $\phi_R(\boldsymbol{x},\,\cdot)_{k,r,s}$ on the interval $[\alpha_i,\,\alpha_{i+1}]$. For that purpose, consider the following. Note that the curve $\rho\colon [\alpha_i,\,\alpha_{i+1}] \to \mathbb{R}^2$, $\rho(t) := (f_1^{r,s}(t),\,f_2^{r,s}(t))$ is of class C^1 and

$$\frac{df_1^{r,s}(t)}{dt} = \frac{d}{dt}\left(c_W + d_{r,s}\,\cos(g_{r,s}-t)\right) = d_{r,s}\,\sin(g_{r,s}-t) \tag{E.391}$$

$$\frac{df_2^{r,s}(t)}{dt} = \frac{d}{dt}\left(c_H + d_{r,s}\,\sin(g_{r,s}-t)\right) = -d_{r,s}\,\cos(g_{r,s}-t) \tag{E.392}$$

and hence

$$\left\|\dot{\rho}(t)\right\|_2 = \sqrt{\left(\frac{df_1^{r,s}(t)}{dt}\right)^2 + \left(\frac{df_2^{r,s}(t)}{dt}\right)^2} = \sqrt{2}\, d_{r,s}. \tag{E.393}$$

By Lemma E.4 a Lipschitz constant for the function $\phi_R(\boldsymbol{x},\, \cdot)_{k,r,s}$ is thus given by

$$\max_{c,d\in[\alpha_i,\,\alpha_{i+1}]} \left| \frac{\phi_R(\boldsymbol{x},\, c)_{k,r,s} - \phi_R(\boldsymbol{x},\, d)_{k,r,s}}{c-d} \right| \leqslant 2\, d_{r,s} \cdot m_\Delta(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}^{(i)}). \tag{E.394}$$

We can thus upper bound (I) and (II) in (E.384) yielding a Lipschitz constant for $g_i^{k,r,s}$ and $g_{i+1}^{k,r,s}$ on $[\alpha_i,\, \alpha_{i+1}]$

$$\max_{c,d\in[\alpha_i,\,\alpha_{i+1}]} \left| \frac{g_i^{k,r,s}(c) - g_i^{k,r,s}(d)}{c-d} \right| \leqslant 2\, d_{r,s} \cdot m_\Delta(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}^{(i)}) \cdot \bar{m}(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}^{(i)}) \tag{E.395}$$

$$= L_i^{k,r,s} (= L_{i+1}^{k,r,s}). \tag{E.396}$$

Finally, we can compute L_r as

$$L = \max_{1\leqslant i\leqslant N-1} \sum_{k=0}^{K-1} \sum_{(r,s)\in V} L_i^{k,r,s} = \max_{1\leqslant i\leqslant N-1} \sum_{k=0}^{K-1} \sum_{r,s\in V} 2d_{r,s} \cdot m_\Delta(\boldsymbol{x}, k, \mathcal{P}_{r,s}^{(i)}) \cdot \overline{m}(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}^{(i)})) \tag{E.397}$$

what concludes the proof. QED.

E.6.2 Scaling

Lemma 7.5 (restated)**.** Let $\boldsymbol{x} \in \mathbb{R}^{K\times W\times H}$ be a K-channel image and let ϕ_S be the scaling transformation. Then, a global Lipschitz constant L for the functions $\{g_i\}_{i=1}^N$ is given by

$$L_{\mathrm{s}} = \max_{1\leqslant i\leqslant N-1} \sum_{k=0}^{K-1} \sum_{r,s\in\Omega\cap\mathbb{N}^2} \frac{\sqrt{2}d_{r,s}}{a^2} \cdot m_\Delta(\boldsymbol{x}, k, \mathcal{P}_{r,s}^{(i)}) \cdot \overline{m}(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}^{(i)}) \tag{E.398}$$

where $\Omega = [0,\, W-1] \times [0,\, H-1]$ and a is the lower boundary value in $\mathcal{S} = [a,\, b]$. The set $\mathcal{P}_{r,s}^{(i)}$ is given by all integer grid pixels that are covered by the trajectory of source pixels of (r, s) when scaling with factors from α_{i+1} to α_i.

Proof. Recall the Definition of the Scaling transformation ϕ_S given by $\phi_S \colon \mathbb{R}^{K\times W\times H} \times \mathbb{R} \to$

$\mathbb{R}^{K\times W\times H}$, where

$$\phi_S(\boldsymbol{x},\,\alpha)_{k,r,s} := Q_{\boldsymbol{x}}\left(k,\,c_W+\frac{r-c_W}{s},\,c_H+\frac{s-c_H}{s}\right). \tag{E.399}$$

Recall that the set Ω is given by $\Omega = [0,\,W-1]\times[0,\,H-1] = \{1,\,\ldots,\,K\}$ and let

$$\Omega_{\mathbb{N}} := \Omega\cap\mathbb{N}^2 \tag{E.400}$$

be the set of integers in Ω. Let $f_1^r\colon[a,\,b]\to\mathbb{R}$ and $f_2^{r,s}\colon[a,\,b]\to\mathbb{R}$ be functions defined as

$$f_1^r(\alpha) := c_W+\frac{r-c_W}{\alpha},\quad f_2^s(\alpha)\ := c_H+\frac{s-c_H}{\alpha}. \tag{E.401}$$

Then, the value of the scaled image $\phi_S(\boldsymbol{x},\,\alpha)$ is given by

$$\phi_S(\boldsymbol{x},\,\alpha)_{k,r,s} = Q_{\boldsymbol{x}}(k,\,f_1^r(\alpha),\,f_2^s(\alpha)) \tag{E.402}$$

where $Q_{\boldsymbol{x}}$ denotes bilinear interpolation. Let

$$\psi_k\colon[a,\,b]\to\mathbb{R},\ \alpha\mapsto Q_{\boldsymbol{x}}(k,\,f_1^r(\alpha),\,f_2^s(\alpha)). \tag{E.403}$$

We notice that, in contrast to rotations, ψ_k is *not* continuous at every $\alpha\in\mathbb{R}_{>0}$. Namely, when considering scaling factors in $(0,\,1)$, bilinear interpolation applies black padding to some $(r,\,s)\in\Omega$ resulting in discontinuities of ψ_k. To see this, consider the following. The interval $[\alpha_{i+1},\,\alpha_i]$ contains a discontinuity of ψ_k, if

$$\begin{cases}\alpha_{i+1}<\dfrac{r-c_W}{c_W}<\alpha_i, & r>c_W,\\[2ex] \alpha_{i+1}<\dfrac{c_W-r}{c_W}<\alpha_i, & r<c_W,\end{cases} \tag{E.404}$$

because then $\exists\,\alpha_0\in[\alpha_{i+1},\,\alpha_i]$ such that $f_1^r(\alpha_0)\in\{0,\,W-1\}\subseteq\Omega$ and hence

$$\phi_S(\boldsymbol{x},\,\alpha_0)_{k,r,s}\neq 0 \tag{E.405}$$

but, for $r>c_W$,

$$\phi_S(\boldsymbol{x},\,\alpha_0+\varepsilon)_{k,r,s}=0\quad\forall\varepsilon>0 \tag{E.406}$$

or, when $r<c_W$,

$$\phi_S(\boldsymbol{x},\,\alpha_0-\varepsilon)_{k,r,s}=0\quad\forall\varepsilon>0. \tag{E.407}$$

A similar reasoning leads to a discontinuity in the s-coordinates. We can thus define the set of discontinuities of ψ_k as

$$\mathcal{D} := \left(\bigcup_{r=0}^{W-1} \mathcal{D}_1^r \right) \cup \left(\bigcup_{s=0}^{H-1} \mathcal{D}_2^s \right) \tag{E.408}$$

where

$$\begin{aligned} \mathcal{D}_1^r &:= \left\{ \alpha_0 \in [a,\, b] \mid f_1^r(\alpha_0) \in \{0,\, W-1\} \right\} \\ \mathcal{D}_2^s &:= \left\{ \alpha_0 \in [a,\, b] \mid f_2^s(\alpha_0) \in \{0,\, H-1\} \right\}. \end{aligned} \tag{E.409}$$

We notice that $|\mathcal{D}| \leqslant H + W$ and hence for large enough N, each interval $[\alpha_i,\, \alpha_{i+1}]$ contains at most 1 discontinuity.

Due to these continuities, we need to modify the general upper bound M of the interpolation error $M_{\mathcal{S}}$ Recall that for $a < b$ and $\{\alpha_i\}_{i=1}^N$, the maximum L_2-sampling error $M_{a,b}$ is given by

$$M_{\mathcal{S}} := \max_{a \leqslant \alpha \leqslant b} \min_{1 \leqslant i \leqslant N} \left|\left| \phi_S(\boldsymbol{x},\, \alpha) - \phi_S(\boldsymbol{x},\, \alpha_i) \right|\right|_2. \tag{E.410}$$

In order to compute an upper bound on (E.410) for scaling, we are interested in finding $M \geqslant 0$ such that

$$M_{\mathcal{S}}^2 \leqslant M \tag{E.411}$$

For scaling, similar as in the case for rotations, we sample α_i uniformly from $[a,\, b]$:

$$\alpha_i = a + \frac{b-a}{N-1}(i-1) \text{ for } 1 \leqslant i \leqslant N. \tag{E.412}$$

and note that $\alpha_1 = b$ and $\alpha_N = a$. For $1 \leqslant i \leqslant N$ Let g_i be the functions $g_i \colon [a,\, b] \to \mathbb{R}_{\geqslant 0}$ defined by

$$g_i(\alpha) := \left\| \phi_S(\boldsymbol{x},\, \alpha) - \phi_S(\boldsymbol{x},\, \alpha_i) \right\|_2^2. \tag{E.413}$$

Note that $\forall\, \alpha \in [a,\, b]$, $\exists\, i$ such that $\alpha \in [\alpha_{i+1},\, \alpha_i]$. Suppose that N is large enough such that $\forall\, i \colon |\mathcal{D} \cap [\alpha_{i+1},\, \alpha_i]| \leqslant 1$ and denote the discontinuity in interval $[\alpha_{i+1},\, \alpha_i]$ by t_i if it exists. Let

$$M_i := \begin{cases} \max\limits_{\alpha_i \leqslant \alpha \leqslant \alpha_{i+1}} \min\{g_i(\alpha),\, g_{i+1}(\alpha)\} & [\alpha_i,\, \alpha_{i+1}] \cap \mathcal{D} = \varnothing \\ \max\left\{ \max\limits_{\alpha_i \leqslant \alpha \leqslant t_i} g_{i+1}(\alpha),\, \max\limits_{t_i \leqslant \alpha \leqslant \alpha_{i+1}} g_i(\alpha) \right\} & [\alpha_i,\, \alpha_{i+1}] \cap \mathcal{D} = \{t_i\} \end{cases} \tag{E.414}$$

Similarly as in the case for rotations, we find

$$M_{\mathcal{S}}^2 \leqslant \max_{1 \leqslant i \leqslant N-1} M_i. \tag{E.415}$$

For simplicity, we assume for the sequel that $\mathcal{D} = \varnothing$. The case where discontinuities exist can be treated analogously. We further divide each interval $[\alpha_i,\, \alpha_{i+1}]$ by sampling $n \in \mathbb{N}$ points $\{\gamma_{i,j}\}_{j=1}^n$ according to

$$\gamma_{i,j} := \alpha_i + \frac{\alpha_{i+1} - \alpha_i}{n-1}(j-1) \text{ for } 1 \leqslant j \leqslant n \tag{E.416}$$

and define

$$m_{i,j} := \max_{\gamma_{i,j} \leqslant \gamma \leqslant \gamma_{i,j+1}} \min\left\{g_i(\gamma),\, g_{i+1}(\gamma)\right\}. \tag{E.417}$$

We can thus upper bound each M_i by

$$M_i \leqslant \max_{1 \leqslant j \leqslant n-1} m_{i,j}. \tag{E.418}$$

In order to find an upper bound on $M_{\mathcal{S}}^2$, we thus need to find an upper bound on $m_{i,j}$ and can proceed analogously to rotations. Namely, setting

$$\begin{aligned} M := \max_{1 \leqslant i \leqslant N-1} \Bigg\{ \max_{1 \leqslant j \leqslant n-1} \Bigg\{ \frac{1}{2} \cdot \Big(\min\big\{ & g_i(\gamma_{i,j}) + g_i(\gamma_{i,j+1}), \\ & g_{i+1}(\gamma_{i,j}) + g_{i+1}(\gamma_{i,j+1}) \big\} \Big) + L \cdot \frac{\gamma_{i,\,j+1} - \gamma_{i,\,j}}{2} \Bigg\} \Bigg\} \end{aligned} \tag{E.419}$$

yields a computable upper bound of the maximum ℓ_2 interpolation error. Computing a Lipschitz constant for g_i and g_{i+1} is also analogous to rotations. The difference lies only in computing a Lipschitz constant for ϕ_S what we will explain in greater detail.

Recall that Lemma E.4 provides a Lipschitz constant for the function $t \mapsto \psi_k(t) := Q_x(k,\, \rho_1(t),\, \rho_2(t))$ where ρ is a differentiable curve with values in $\mathbb{R}^2$. Namely, a Lipschitz constant for ψ_k is given by

$$L_k = \max_{t \in [t_1, t_2]} \left(\sqrt{2}\, \left\| \dot{\rho}(t) \right\|_2 \cdot m_\Delta(\boldsymbol{x},\, k,\, \lfloor \rho(t) \rfloor) \right). \tag{E.420}$$

Consider the curve

$$\rho(t) := (f_1^r(t),\, f_2^s(t)), \quad t > 0 \tag{E.421}$$

and note that it is differentiable with derivatives

$$\frac{df_1^r(t)}{dt} = \frac{d}{dt}\left(c_W + \frac{r - c_W}{t}\right) = \frac{c_W - r}{t^2} \tag{E.422}$$

$$\frac{df_2^s(t)}{dt} = \frac{d}{dt}\left(c_H + \frac{s - c_H}{t}\right) = \frac{c_H - s}{t^2} \tag{E.423}$$

and

$$\left\|\dot{\rho}(t)\right\|_2 = \frac{1}{t^2}\sqrt{(c_W - r)^2 + (c_H - s)^2}. \tag{E.424}$$

A Lipschitz constant for $\phi_S(x, \cdot)_{k,r,s}$ is thus given by

$$L_k^{r,s} = \max_{t\in[t_1,\,t_2]}\left(\frac{\sqrt{(c_W - r)^2 + (c_H - s)^2}}{t^2} \cdot \sqrt{2} m_\Delta(\boldsymbol{x},\, k,\, \lfloor\rho(t)\rfloor)\right) \tag{E.425}$$

$$\leqslant \frac{\sqrt{(c_W - r)^2 + (c_H - s)^2}}{t_1^2} \cdot \sqrt{2} \cdot m_\Delta(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}) \tag{E.426}$$

$$\leqslant \frac{\sqrt{(c_W - r)^2 + (c_H - s)^2}}{a^2} \cdot \sqrt{2} \cdot m_\Delta(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}) \tag{E.427}$$

where

$$\mathcal{P}_{r,s} = \bigcup_{\alpha\in[t_1,\,t_2]} \{(\lfloor f_1^r(t)\rfloor,\, \lfloor f_2^s(t)\rfloor)\}. \tag{E.428}$$

Finally, setting

$$L_i^{k,r,s} := L_k^{r,s} \cdot \bar{m}(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}^{(i)}) \tag{E.429}$$

and

$$\begin{aligned} L_s &= \max_{1\leqslant i\leqslant N-1} \sum_{k=0}^{K-1} \sum_{(r,s)\in\Omega_{\mathbb{N}}} L_i^{k,r,s} \\ &= \max_{1\leqslant i\leqslant N-1} \sum_{k=0}^{K-1} \sum_{r,s\in\Omega\cap\mathbb{N}^2} \frac{\sqrt{2} d_{r,s}}{a^2} \cdot m_\Delta(\boldsymbol{x}, k, \mathcal{P}_{r,s}^{(i)}) \cdot \overline{m}(\boldsymbol{x},\, k,\, \mathcal{P}_{r,s}^{(i)}) \end{aligned} \tag{E.430}$$

yields the desired Lipschitz constant. QED.

E.7 ALGORITHM DESCRIPTION FOR DIFFERENTIALLY RESOLVABLE TRANSFORMATIONS

Algorithm E.1 presents a pseudo-code for interpolation error M computation, taking rotation transformation as the example. It corresponds to the description in Section 7.5.2. Algorithm E.2 presents a pseudo-code for progressive sampling. It corresponds to the de-

Algorithm E.1: Interpolation Error M Computation for Rotation Transformation.

Input: clean input image x;
interval of rotation angle to certify $[a, b]$;
number of first-level samples N;
number of second-level samples n
Output: rotation angle samples $\{\alpha_i\}_{i=1}^N$;
upper bound M of squared ℓ_2-interpolation error

$$M_{\mathcal{S}}^2 = \arg\max_{\alpha\in[a,b]} \min_{1\leqslant i\leqslant N} \|\tilde{\phi}_R(x,\alpha) - \tilde{\phi}_R(x,\alpha_i)\|_2^2.$$

/ Compute Lipschitz constant L_r (7.35) */*
$\alpha_1 \leftarrow a$
for $i = 1, \ldots, N-1$ **do**
  $\alpha_{i+1} \leftarrow a + (b-a)\cdot \frac{i}{N-1}$ (7.24)
  for all $(r,s) \in V$ **do**
    / V and $\mathcal{P}_{r,s}^{(i)}$ are defined in Lemma 7.4 */*
    Compute trajectory covered grid pixels $\mathcal{P}_{r,s}^{(i)}$
    for $k = 0, \ldots, K-1$ **do**
      Compute $2d_{r,s}\cdot m_\Delta(x,k,\mathcal{P}_{r,s}^{(i)})\cdot \bar{m}(x,k,\mathcal{P}_{r,s}^{(i)})$ (7.35)
    end for
  end for
  $L_{r,i} \leftarrow \sum_{k=0}^{K-1}\sum_{(r,s)\in V} 2d_{r,s}\cdot m_\Delta(x,k,\mathcal{P}_{r,s}^{(i)})\cdot \bar{m}(x,k,\mathcal{P}_{r,s}^{(i)})$.
end for
$L_r \leftarrow \max_{1\leqslant i\leqslant N-1} L_{r,i}$ (7.35)
/ Compute interpolation error bound M (7.27) from stratified sampling */*
for $i = 1, \ldots, N-1$ **do**
  for $j = 1, \ldots, n$ **do**
    / Second-level sampling */*
    $\gamma_{i,j} \leftarrow \alpha_i + (\alpha_{i+1} - \alpha_i)\cdot \frac{j-1}{n-1}$ (7.28)
  end for
  $M_i \leftarrow 0$
  for $j = 1, \ldots, n-1$ **do**
    Compute $g_i(\gamma_{i,j})$, $g_i(\gamma_{i,j+1})$, $g_{i+1}(\gamma_{i,j})$, and $g_{i+1}(\gamma_{i,j+1})$ (7.25)
    $M_i \leftarrow \max\Big\{M_i, \min\big\{g_i(\gamma_{i,j}) + g_i(\gamma_{i,j+1}),$
      $g_{i+1}(\gamma_{i,j}) + g_{i+1}(\gamma_{i,j+1})\big\}\Big\}$
  end for
  $M_i \leftarrow \frac{1}{2}M_i + L\cdot \frac{b-a}{(N-1)(n-1)}$ (7.30)
end for
Return: $M \leftarrow \max_{1\leqslant i\leqslant N-1} M_i$ (7.27)

scription in Section 7.5.7. We remark that in practice, we sample in mini-batches with batch size B. The error tolerance T is set to $M_{\mathcal{S}}$ (7.23) if certifying rotation or scaling, is set to $\sqrt{M_{\mathcal{S}}^2 + \sigma^2/\sigma_b^2\cdot b_0^2}$ (7.50) if certifying the composition of rotation or scaling with brightness

Algorithm E.2: Progressive Sampling for Certification.

Input: clean input image x with true class k_A; first-level parameter samples $\{\alpha_i\}_{i=1}^N$; perturbation random variable ε with variance σ^2; ℓ_2 error tolerance T; batch size B; sampling size limit n_s; confidence level p.
Output: with probability $1-p$, whether $g(\cdot;\varepsilon)$ is certifiably robust at $\phi(x,\, \alpha)$.
for $i = 1, \ldots, N$ **do**
  $x^{(i)} \leftarrow \phi(x, \alpha_i)$
  $j \leftarrow 0$
  while $j \leqslant n_s$ **do**
    Sample B instances of $\phi(x^{(i)}, \varepsilon)$, and use them to update empirical mean $\hat{q}(y_A|\, x^{(i)};\, \varepsilon)$.
    $j \leftarrow j + B$.
    / Lower confidence interval bound with these j samples */*
    $\underline{p_A}^{(i)} = \text{LowerConfBound}(\hat{q}(y_A|\, x^{(i)}; \varepsilon), j, 1 - p/N)$.
    if $\underline{R_i} = \sigma\Phi^{-1}\left(\underline{p_A}^{(i)}\right) > T$ **then**
      / Already get the certification that $R_i > T$, break */*
      `Break`
    end if
  end while
  if $\underline{R_i} = \sigma\Phi^{-1}\left(\underline{p_A}^{(i)}\right) \leqslant T$ **then**
    / Cannot ensure that $R_i > T$. So cannot ensure that $R = \min R_i > T$. Early halt */*
    Return: `false`
  end if
end for
Return: `true`

change within $[-b_0,\, b_0]$; and is set to $\sqrt{(M_{\mathcal{S}} + r)^2 + \sigma^2/\sigma_b^2 \cdot b_0^2}$ (7.51) if certifying the composition of rotation or scaling, brightness change $[-b_0,\, b_0]$, and ℓ_2 bounded perturbations within r. The two algorithms jointly constitute our pipeline **TSS-DR** for certifying against differentially resolvable transformations as shown in Figure 7.5(a).

E.8 EXPERIMENT DETAILS

Here we provide all omitted details about experiment setup, implementation, discussion about baselines, evaluation protocols, results, findings, and analyses.

E.8.1 Model Preparation

As previous work shows, an undefended model is very vulnerable even under simple random semantic attacks. Therefore, to obtain nontrivial certified robustness, we require the model itself to be trained to be robust against semantic transformations. We apply data

augmentation training [77] combined with Consistency regularization [165] to train the base classifiers. The data augmentation training randomly transforms the input by the specified transformation using parameters drawn from the specified smoothing distribution/strategy. The Consistency regularization further enhances the consistency of the base classifiers' prediction among the drawn parameters. Then, the base classifiers are used to construct smoothed classifiers by the specified smoothing distribution/strategy, and we compute its robustness certification with our approach.

On relatively small datasets MNIST and CIFAR-10, the models are trained from scratch. On MNIST, we use a convolutional neural network (CNN) composed of four convolutional layers and three fully connected layers. On CIFAR-10, we use the neural network ResNet-110, a 110-layer ResNet model [142]. These model structures are the same as in the literature [77, 317, 427] for direct comparison. On MNIST, we train 100 epochs; on CIFAR-10, we train 150 epochs. The batch sizes (B) are 400 and 256 on MNIST and CIFAR-10, respectively. The learning rate on both datasets is initialized to 0.01, and after every 50 epochs, the learning rate is multiplied by 0.1. For resolvable transformations, the data augmentation usually uses the same smoothing distribution/strategy as we will use to construct the smoothed classifier. In particular, for brightness and contrast transformation, we empirically observe that a larger variance during inference time helps to improve the certified accuracy under large attack radius, and for the composition of Gaussian blur, brightness, contrast, and translation, we additionally add small additive Gaussian noise to improve its ability to defend against other unforeseen attacks as we will discuss in Appendix E.8.8. For differentially resolvable transformations, since Gaussian noise is required in constructing the smoothed classifier, the data augmentation jointly adds Gaussian noise and the transformation to certify against. The detailed hyperparameters such as distribution type and variance are listed in Table E.2. The weight of Consistency regularization is set to 10 throughout the training.

On the large ImageNet dataset, we finetune the existing trained models. For resolvable transformations, we finetune from ResNet-50 model in `torchvision` library [300]. For differentially resolvable transformations, since the base classifier should also be robust under Gaussian noise, we finetune from Resnet-50 model in [317] that achieves state-of-the-art robustness under Gaussian noise. In either case, we follow the same data augmentation scheme as on MNIST and CIFAR-10, and we finetune for two epochs with batch size (B) 128, learning rate 0.001, and Consistency regularization weight 10. During certification (e.g., Algorithm E.2), we use the same batch sizes as during training on these datasets.

The channel-wise normalization is used for all models on these three datasets as in [77, 317]. On all three datasets, in each training epoch, we feed in the whole training dataset without random shuffle.

We remark that since our approach focuses on robustness certification and the smoothing strategy to improve certified robustness, we did not fully explore the potential of improving certified robustness from the training side, nor did we file-tune the training hyperparameters. Therefore, though we already achieved the state of the art by our effective robustness certification and smoothing strategies, we believe the results could be further improved by more effective training approaches.

E.8.2 Implementation Details

We implement the whole approach along with the training scripts in a tool based on `PyTorch`. For resolvable transformations, we extend the smoothing module from Cohen et al [77] to accommodate various smoothing strategies and smoothing distributions. The predict and certify modules are kept the same. For differentially resolvable transformations, since the stratified sampling requires $N \times n$ times of transformation to compute the interpolation error bound (where N is the number of first-level samples and n the number of second-level samples), we implement a fast `C` module and integrate it to our `Python`-based tool. It empirically achieves $3-5$x speed gain compared with `OpenCV`[283]-based transformation. For Lipschitz upper bound computation, since the loop in `Python` is slow, we reformulate the computation by loop-free tensor computations using `numpy`. It empirically achieves $20-40$x speed gain compared to the plain loop-based implementation. The full code implementation of our TSS tool along with all trained models are publicly available at `https://github.com/AI-secure/semantic-randomized-smoothing`.

E.8.3 Details on Attacks

We use the following three attacks to evaluate the empirical accuracy of both TSS models and vanilla models: Random Attack, Random+ Attack, and PGD Attack. The Random Attack is used in previous work [22, 118] but does not consider the intrinsic characteristics of semantic transformations. Thus, we propose Random+ Attack and PGD Attack as the alternatives since they are adaptively designed for our smoothed TSS models and also consider the intrinsic characteristics of these transformations.

Random Attack

The random attack is used to evaluate the empirical robust accuracy, which is an upper bound of the certified robust accuracy. The random attack reads in the clean input, and

uniformly samples N parameters from the pre-defined transformation parameter space to transform the input following uniform distribution. If the model gives a wrong prediction on any of these N transformed inputs, we treat this sample as being successfully attacked; otherwise, the sample counts toward the empirical robust accuracy. We denote by N the "number of initial starts". In the main experiments, we set $N = 100$, and in the following ablation study (Appendix E.8.7), we also compare the behaviors of the three attacks under $N = 10/20/50$.

For transformations with a hyper-rectangle parameter space, including brightness, contrast, scaling, rotation, Gaussian blur, and their compositions, we uniformly sample transformation parameters for each coordinate. For transformations with discrete parameter space, such as translation, we draw the parameter with equal probability. When the transformation is composed with ℓ_p-bounded perturbations, we additionally generate the perturbation vector using FGSM attack [359], where the precise gradient is used for vanilla models, and the empirical mean gradient over 100 samples is used for smoothed TSS models.

Adaptive Attack: Random+

The Random+ attack follows the same procedure as the Random attack. The only difference is that, instead of using uniform distribution for sampling transformation parameters, we use the Beta distribution $\text{Beta}(0.5, 0.5)$.

Formally, suppose the transformation space is $[a, b]$. In Random attack, we generate the attack parameter ε randomly as follows:

$$\varepsilon' \sim \text{Unif}(0, 1), \quad \varepsilon \leftarrow a + (b - a) \times \varepsilon'. \tag{E.431}$$

In Random+ attack, we generate the attack parameter δ randomly as follows:

$$\delta' \sim \text{Beta}(0.5, 0.5), \quad \delta \leftarrow a + (b - a) \times \delta'. \tag{E.432}$$

We choose the Beta distribution because, intuitively, an adversarial example would be more likely to exist at the boundary, i.e., closer to a or b. For example, suppose the rotation attacker has permitted angles in $[-r, r]$, then the adversarial samples may be more likely to have large rotation angle. As shown in Figure E.1, the Beta distribution helps to assign more mass when the parameter becomes closer to the boundary. Choosing other Beta distribution hyperparameters could control the trade-off on sampling weights over the boundary or over center, and we empirically find $\text{Beta}(0.5, 0.5)$ already works very well as shown in

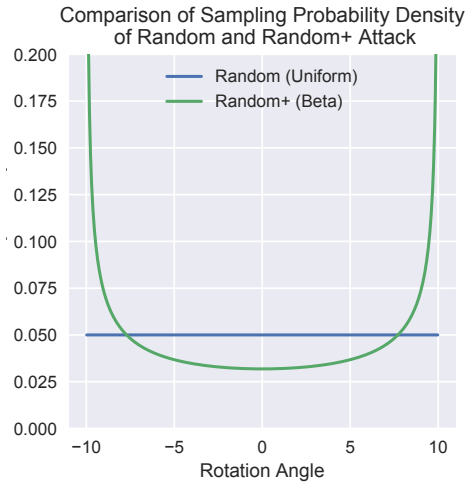


Figure E.1: Comparison of probability density of Random and Random+ attack when attacking the rotation transformation with rotation angle between -10° and $+10^\circ$.

experiments (Appendix E.8.7).

Adaptive Attack: PGD

We propose the semantic transformation version of PGD attack as follows: (1) Initialize the transformation parameter following the same process as in Random+ Attack; (2) Suppose the current parameter is $(\alpha_1, \cdots, \alpha_z)$. The attack slightly perturbs each coordinate from α_i to $\alpha_i \pm \tau_i$ respectively and obtain $2z$ perturbed candidates. (3) The attack clips each coordinate to be within the specified range, and then choose the candidate that yields the largest gain of cross-entropy loss (for vanilla models) or empirical mean cross-entropy loss (for TSS models) to update the current parameter. Then go to step (2). The loop repeats for 10 iterations for each sample obtained in step (1). The perturbed step size $\tau_i = l_i/10$ where l_i is the length of the specified interval on the i-th coordinate. Finally, if the transformation is composed with ℓ_p-bounded perturbations, we additionally generate the perturbation vector in the same way as in Random attack.

For each Step (1) sample we would have an output and thus there are N outputs. If any of these outputs fool the target model, we treat this sample as being successfully attacked; otherwise, the sample counts toward the empirical robust accuracy. Note that the translation transformation has a discrete parameter space, thus PGD attack is not applicable.

We refer to the attack as the semantic transformation version of PGD attack because: (1) It involves multiple initial starts; (2) It leverages the local landscape information to maximize the loss function iteratively. (3) It clips (i.e., "projects") the parameters to be

within the perturbation range. Compared to the classical PGD attack under ℓ_p-norm constraint, we use coordinate-wise perturbations to probe the local landscape to circumvent the hardness of obtaining the gradient with respect to transformation parameters.

E.8.4 Details on Baseline Approaches

DeepG [22] is based on linear relaxations. The code is open-sourced, and we utilize it to provide a direct comparison. The code provides trained models on MNIST and CIFAR-10, while on ImageNet the method is too slow and memory-consuming to run. On both MNIST and CIFAR-10, we use the provided trained models from the code. In terms of computation time, since our approach uses far less than 1000 s for certification per input on MNIST and CIFAR-10, we tune the hyperparameters to let the code spend roughly 1000 s for certification.

Interval [343] is based on interval bound propagation. We also utilize the open-sourced code to provide a direct comparison. The settings are the same as in DeepG.

VeriVis [292] provides an enumeration-based solution when the number of possible transformation parameters or the number of possible transformed images is finite. In our evaluation, only translation satisfies this property. Therefore, as the baseline, we implement the enumeration-based robustness certification algorithm for our trained robust models.

Semantify-NN [268] proposes to insert a preprocessing layer to reduce the verification against semantic transformations to the verification against classical ℓ_p noises. To our knowledge, the code has not been open-sourced yet. Therefore, we directly compare with the numbers reported in their paper. Since they report the average of certified robust magnitude, we apply Markov's inequality to obtain an upper bound of their certified robust accuracy. For example, they report 46.24 degrees as the average certified robust rotation angles. It means that $\Pr[r \geqslant 50^\circ] \leqslant \mathbb{E}[r]/50 = 92.48\%$, i.e., the certified robust accuracy is no larger than 92.48% when fixing the rotation angle to be 50°.

For brightness and contrast changes, Semantify-NN considers first applying the change and then clipping to $[0, 1]$, while our TSS considers only brightness and contrast changes. This makes a one-to-one comparison with [268] difficult, but since other baselines (e.g., [22]) consider the same setting as we do, and to align with most baselines, we slightly sacrifice comparability in this special case. For interested readers who would like to have an absolutely fair comparison with Semantify-NN on brightness and contrast changes, they can extend our TSS by modeling Semantify-NN's tranformation by $\phi_{BC}(\boldsymbol{x}, (b, c)) \circ \phi_{clip}(\boldsymbol{x}, t_l, t_h)$, where ϕ_{clip} clips the pixel intensities lower than t_l and higher than t_h. Applying TSS-R on transformation parameters (b, c, t_l, t_h) then derives the robustness certification under the

same threat model as Semantify-NN.

DistSPT [118] combines randomized smoothing and interval bound propagation to provide certified robustness against semantic transformations. Concretely, the approach leverages interval bound propagation to compute the upper bound of interpolation error and then applies randomized smoothing. On small datasets such as MNIST and CIFAR-10, the approach is able to provide nontrivial robustness certification. Though the certified robust accuracy is inferior than TSS as reflected in Table 7.3. We use their reported numbers in [118, Table 4] for DistSPT^x for comparison, since the certification goal and evaluation protocol are the same as ours. On ImageNet, as described in [118, Section 7.4], the interval bound propagation is computationally expensive and loose. Therefore, they use sampling to estimate the interpolation error, which makes the robustness certification no longer hold against arbitrary attacks but just a certain random attack ("worst-of-10" attack).

IndivSPT [118] provides a different certification goal from the above approaches. At a high level, the approach uses a transformed image as the input where the transformation parameter is within predefined threshold. Then the approach certifies whether the prediction for the transformed image and the prediction for the original image are the same. In contrast, TSS and other baseline approaches take original image as the input and certifies whether there exists no transformed image that can mislead the model. Due to different certification goals, TSS is not comparable with IndivSPT.

E.8.5 Benign Accuracy

Table E.1 shows the benign accuracy of our models corresponding to Table 7.3. For comparison, the vanilla trained models have benign accuracy 98.6% on MNIST, 88.6% on CIFAR-10, and 74.4% on ImageNet.

We observe that though the trade-off between accuracy and (certified) robustness is widely reported both theoretically [269, 427, 442] and empirically (e.g., [77, 165, 437]) in classical ℓ_p threat model, it does not always exist in our semantic defense setting. Specifically, for resolvable transformations, we do not observe an apparent loss of benign accuracy in our certifiably robust models; while for differentially resolvable transformations (those involving scaling and rotation), there is no loss on MNIST, slight losses on CIFAR-10, and apparent losses on ImageNet. When there does exist a trade-off between benign accuracy and certified robust accuracy, we show that smoothing variance levels control it in Appendix E.8.10.

Table E.1: Benign accuracy of our TSS models corresponding to those in Table 7.3. Certified robust accuracy shown as reference.

Transformation	Dataset	Attack Radius	Certified Robust Acc.	Benign Acc.
Gaussian Blur	MNIST	Squared Radius $\alpha \leqslant 36$	90.6%	96.8%
	CIFAR-10	Squared Radius $\alpha \leqslant 16$	63.6%	76.2%
	ImageNet	Squared Radius $\alpha \leqslant 36$	51.6%	59.2%
Translation (Reflection Pad.)	MNIST	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 8$	99.6%	99.6%
	CIFAR-10	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 20$	80.8%	87.0%
	ImageNet	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 100$	50.0%	73.0%
Brightness	MNIST	$b \pm 50\%$	98.2%	98.2%
	CIFAR-10	$b \pm 40\%$	87.0%	87.8%
	ImageNet	$b \pm 40\%$	70.0%	72.2%
Contrast and Brightness	MNIST	$c \pm 50\%, b \pm 50\%$	97.6%	98.0%
	CIFAR-10	$c \pm 40\%, b \pm 40\%$	82.4%	86.8%
	ImageNet	$c \pm 40\%, b \pm 40\%$	61.4%	72.2%
Gaussian Blur, Translation, Brightness, and Contrast	MNIST	$\alpha \leqslant 1, c, b \pm 10\%, \sqrt{\Delta x^2 + \Delta y^2} \leqslant 5$	90.2%	98.2%
	CIFAR-10	$\alpha \leqslant 1, c, b \pm 10\%, \sqrt{\Delta x^2 + \Delta y^2} \leqslant 5$	58.2%	77.6%
	ImageNet	$\alpha \leqslant 10, c, b \pm 20\%, \sqrt{\Delta x^2 + \Delta y^2} \leqslant 10$	32.8%	61.6%
Rotation	MNIST	$r \pm 50°$	97.4%	99.4%
	CIFAR-10	$r \pm 10°$	70.6%	83.2%
		$r \pm 30°$	63.6%	82.6%
	ImageNet	$r \pm 30°$	30.4%	46.2%
Scaling	MNIST	$s \pm 30\%$	99.0%	99.4%
	CIFAR-10	$s \pm 30\%$	58.8%	79.8%
	ImageNet	$s \pm 30\%$	26.4%	50.8%
Rotation and Brightness	MNIST	$r \pm 50°, b \pm 20\%$	97.0%	99.4%
	CIFAR-10	$r \pm 10°, b \pm 10\%$	70.2%	83.0%
		$r \pm 30°, b \pm 20\%$	61.4%	82.6%
	ImageNet	$r \pm 30°, b \pm 20\%$	26.8%	45.8%
Scaling and Brightness	MNIST	$s \pm 50\%, b \pm 50\%$	96.6%	99.4%
	CIFAR-10	$s \pm 30\%, b \pm 30\%$	54.2%	79.6%
	ImageNet	$s \pm 30\%, b \pm 30\%$	23.4%	50.8%
Rotation, Brightness, and ℓ_2	MNIST	$r \pm 50°, b \pm 20\%, \\|\delta\\|_2 \leqslant .05$	96.6%	99.4%
	CIFAR-10	$r \pm 10°, b \pm 10\%, \\|\delta\\|_2 \leqslant .05$	64.2%	83.0%
		$r \pm 30°, b \pm 20\%, \\|\delta\\|_2 \leqslant .05$	55.2%	82.6%
	ImageNet	$r \pm 30°, b \pm 20\%, \\|\delta\\|_2 \leqslant .05$	26.6%	45.8%
Scaling, Brightness, and ℓ_2	MNIST	$s \pm 50\%, b \pm 50\%, \\|\delta\\|_2 \leqslant .05$	96.4%	99.4%
	CIFAR-10	$s \pm 30\%, b \pm 30\%, \\|\delta\\|_2 \leqslant .05$	51.2%	79.6%
	ImageNet	$s \pm 30\%, b \pm 30\%, \\|\delta\\|_2 \leqslant .05$	22.6%	50.8%

E.8.6 Smoothing Distributions and Running Time Statistics

In Table E.2, we present the smoothing distributions with concrete parameters and average certification computing time per sample for results in main table (Table 7.3). In the table, α is for squared kernel radius for Gaussian blur; Δx and Δy are for translation displacement on horizontal and vertical direction; b and c are for brightness shift and contrast change respectively as in $x \mapsto (1 + c)x + b$; r is for rotation angle; s is for size scaling ratio; ε is for additive noise vector; and $\|\delta\|_2$ for ℓ_2 norm of permitted additional perturbations. Specifically, "Training Distribution" stands for the distributions for data augmentation during training the base classifiers; and "Smoothing Distribution" stands for the distributions for constructing the smoothed classifiers for certification.

We select these distributions according to the principles in Appendix E.8.1.

Table E.2: Detailed smoothing distributions and running time statistics for our TSS. $\mathcal{N}(\mu, \Sigma)$ is the normal distribution, $\exp(\lambda)$ is the exponential distribution, $\mathcal{U}([a, b])$ is the uniform distribution. Random variable ϵ is the elementwise noise as in Corollary 7.2. "Cert." means certification.

Transformation	Dataset	Attack Radius	Training Distribution	Smoothing Distribution	Avg. Cert. Time per Sample
Gaussian Blur	MNIST	Squared Radius $\alpha \leqslant 36$	$\alpha \sim \text{Exp}(1/10)$		7.9 s
	CIFAR-10	Squared Radius $\alpha \leqslant 16$	$\alpha \sim \text{Exp}(1/5)$		30.9 s
	ImageNet	Squared Radius $\alpha \leqslant 36$	$\alpha \sim \text{Exp}(1/10)$		45.7 s
Translation (Reflection Pad.)	MNIST	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 8$	$(\Delta x, \Delta y) \sim \mathcal{N}(0, 10^2 I)$		10.2 s
	CIFAR-10	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 20$	$(\Delta x, \Delta y) \sim \mathcal{N}(0, 15^2 I)$		39.4 s
	ImageNet	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 100$	$(\Delta x, \Delta y) \sim \mathcal{N}(0, 30^2 I)$		161.9 s
Brightness	MNIST	$b \pm 50\%$	$b \sim \mathcal{N}(0, 0.6^2)$		2.1 s
	CIFAR-10	$b \pm 40\%$	$b \sim \mathcal{N}(0, 0.3^2)$		4.4 s
	ImageNet	$b \pm 40\%$	$b \sim \mathcal{N}(0, 0.4^2)$		45.1 s
Contrast and Brightness	MNIST	$c \pm 50\%, b \pm 50\%$	$(c, b) \sim \mathcal{N}(0, 0.6^2 I)$	$(c, b) \sim \mathcal{N}(0, 1.0^2 I)$	9.8 s
	CIFAR-10	$c \pm 40\%, b \pm 40\%$	$(c, b) \sim \mathcal{N}(0, 0.4^2 I)$	$(c, b) \sim \mathcal{N}(0, 0.6^2 I)$	45.0 s
	ImageNet	$c \pm 40\%, b \pm 40\%$	$(c, b) \sim \mathcal{N}(0, 0.4^2 I)$		325.6 s
Gaussian Blur, Translation, Brightness, and Contrast	MNIST	$\alpha \leqslant 5, c, b \pm 10\%$, $\sqrt{\Delta x^2 + \Delta y^2} \leqslant 5$	$\alpha \sim \text{Exp}(1/10)$ $(\Delta x, \Delta y) \sim \mathcal{N}(0, 10^2 I)$ $(c, b) \sim \mathcal{N}(0, 0.3^2 I)$ $\epsilon \sim \mathcal{N}(0, 0.05^2 I)$	$\alpha \sim \text{Exp}(1/10)$ $(\Delta x, \Delta y) \sim \mathcal{N}(0, 10^2 I)$ $(c, b) \sim \mathcal{N}(0, 0.3^2 I)$	12.9 s
	CIFAR-10	$\alpha \leqslant 1, c, b \pm 10\%$, $\sqrt{\Delta x^2 + \Delta y^2} \leqslant 5$	$\alpha \sim \text{Exp}(1)$ $(\Delta x, \Delta y) \sim \mathcal{N}(0, 10^2 I)$ $(c, b) \sim \mathcal{N}(0, 0.3^2 I)$ $\epsilon \sim \mathcal{N}(0, 0.01^2 I)$	$\alpha \sim \text{Exp}(1)$ $(\Delta x, \Delta y) \sim \mathcal{N}(0, 10^2 I)$ $(c, b) \sim \mathcal{N}(0, 0.3^2 I)$	43.1 s
	ImageNet	$\alpha \leqslant 10, c, b \pm 20\%$, $\sqrt{\Delta x^2 + \Delta y^2} \leqslant 10$	$\alpha \sim \text{Exp}(1/5)$ $(\Delta x, \Delta y) \sim \mathcal{N}(0, 20^2 I)$ $(c, b) \sim \mathcal{N}(0, 0.4^2 I)$ $\epsilon \sim \mathcal{N}(0, 0.01^2 I)$	$\alpha \sim \text{Exp}(1/5)$ $(\Delta x, \Delta y) \sim \mathcal{N}(0, 20^2 I)$ $(c, b) \sim \mathcal{N}(0, 0.4^2 I)$	238.1 s
Rotation	MNIST	$r \pm 50^\circ$	Same as Rotation and Brightness	$\epsilon \sim \mathcal{N}(0, 0.12^2 I)$	20.1 s
	CIFAR-10	$r \pm 10^\circ$		$\epsilon \sim \mathcal{N}(0, 0.05^2 I)$	52.8 s
		$r \pm 30^\circ$		$\epsilon \sim \mathcal{N}(0, 0.05^2 I)$	141.0 s
	ImageNet	$r \pm 30^\circ$		$\epsilon \sim \mathcal{N}(0, 0.5^2 I)$	2358.1 s
Scaling	MNIST	$s \pm 30\%$	Same as Scaling and Brightness	$\epsilon \sim \mathcal{N}(0, 0.12^2 I)$	17.7 s
	CIFAR-10	$s \pm 30\%$		$\epsilon \sim \mathcal{N}(0, 0.12^2 I)$	42.2 s
	ImageNet	$s \pm 30\%$		$\epsilon \sim \mathcal{N}(0, 0.5^2 I)$	1201.2 s
Rotation and Brightness	MNIST	$r \pm 50^\circ, b \pm 20\%$	$r \sim \mathcal{U}([-55, 55])$ $\epsilon \sim \mathcal{N}(0, 0.12^2 I)$ $b \sim \mathcal{N}(0, 0.2^2)$	$\epsilon \sim \mathcal{N}(0, 0.12^2 I)$ $b \sim \mathcal{N}(0, 0.2^2)$	31.4 s
	CIFAR-10	$r \pm 10^\circ, b \pm 10\%$	$r \sim \mathcal{U}([-12.5, 12.5])$ $\epsilon \sim \mathcal{N}(0, 0.05^2 I)$ $b \sim \mathcal{N}(0, 0.2^2)$	$\epsilon \sim \mathcal{N}(0, 0.05^2 I)$ $b \sim \mathcal{N}(0, 0.2^2)$	62.3 s
		$r \pm 30^\circ, b \pm 20\%$	$r \sim \mathcal{U}([-35, 35])$ $\epsilon \sim \mathcal{N}(0, 0.05^2 I)$ $b \sim \mathcal{N}(0, 0.2^2)$	$\epsilon \sim \mathcal{N}(0, 0.05^2 I)$ $b \sim \mathcal{N}(0, 0.2^2)$	157.0 s
	ImageNet	$r \pm 30^\circ, b \pm 20\%$	$r \sim \mathcal{U}([-35, 35])$ $\epsilon \sim \mathcal{N}(0, 0.5^2 I)$ $b \sim \mathcal{N}(0, 0.2^2)$	$\epsilon \sim \mathcal{N}(0, 0.5^2 I)$ $b \sim \mathcal{N}(0, 0.2^2)$	2475.6 s
Scaling and Brightness	MNIST	$s \pm 50\%, b \pm 50\%$	$s \sim \mathcal{U}([0.45, 1.55])$ $\epsilon \sim \mathcal{N}(0, 0.12^2 I)$ $b \sim \mathcal{N}(0, 0.5^2)$	$\epsilon \sim \mathcal{N}(0, 0.12^2 I)$ $b \sim \mathcal{N}(0, 0.5^2)$	74.9 s
	CIFAR-10	$s \pm 30\%, b \pm 30\%$	$s \sim \mathcal{U}([0.65, 1.35])$ $\epsilon \sim \mathcal{N}(0, 0.12^2 I)$ $b \sim \mathcal{N}(0, 0.3^2)$	$\epsilon \sim \mathcal{N}(0, 0.12^2 I)$ $b \sim \mathcal{N}(0, 0.3^2)$	44.5 s
	ImageNet	$s \pm 30\%, b \pm 30\%$	$s \sim \mathcal{U}([0.65, 1.35])$ $\epsilon \sim \mathcal{N}(0, 0.5^2 I)$ $b \sim \mathcal{N}(0, 0.3^2)$	$\epsilon \sim \mathcal{N}(0, 0.5^2 I)$ $b \sim \mathcal{N}(0, 0.3^2)$	1401.6 s
Rotation, Brightness, and ℓ_2	MNIST	$r \pm 50^\circ, b \pm 20\%, \|\delta\|_2 \leqslant .05$	Same as Rotation and Brightness	Same as Rotation and Brightness	35.1 s
	CIFAR-10	$r \pm 10^\circ, b \pm 10\%, \|\delta\|_2 \leqslant .05$			132.5 s
		$r \pm 30^\circ, b \pm 20\%, \|\delta\|_2 \leqslant .05$			520.2 s
	ImageNet	$r \pm 30^\circ, b \pm 20\%, \|\delta\|_2 \leqslant .05$			3463.8 s
Scaling, Brightness, and ℓ_2	MNIST	$s \pm 50\%, b \pm 50\%, \|\delta\|_2 \leqslant .05$	Same as Scaling and Brightness	Same as Scaling and Brightness	75.1 s
	CIFAR-10	$s \pm 30\%, b \pm 30\%, \|\delta\|_2 \leqslant .05$			50.0 s
	ImageNet	$s \pm 30\%, b \pm 30\%, \|\delta\|_2 \leqslant .05$			1657.7 s

E.8.7 Comparison of Random Attack and Adaptive Attacks: Detail

In Table 7.3, we compare the empirical robust accuracy of vanilla models and TSS models under random attacks and two adaptive attacks: Random+ and PGD. However, due to space limits, we omit the empirical accuracy of each adaptive attack and we just present the minimum empirical accuracy among them. In Table E.3, Table E.4, and Table E.5, for all transformations on MNIST, CIFAR-10, and ImageNet respectively, we present the detailed empirical accuracy of each attack under different number of initial starts $N = 10/20/50/100$. Note that the main table (Table 7.3) shows the empirical accuracy with $N = 100$ for all attacks.

From these three tables, we cross-validate the findings shown in the main chapter: the adaptive attack decreases the empirical accuracy of TSS models slightly, while it decreases that of vanilla models more. Moreover, when comparing these three attacks, we find that with a small number of initial starts (e.g., $N = 10$), the PGD attack is typically the most powerful. However, with a large number of initial starts (e.g., $N = 100$), Random+ attack sometimes becomes better. We conjecture that the optimization goal of PGD attack—maximization of cross-entropy loss—might be sub-optimal in terms of increasing the misclassification rate. Thus, with a small number of initial starts, PGD is better than Random/Random+ attack due to the iterative ascending. However, with a large number of initial starts, both PGD and Random+ attack can sufficiently explore the adversarial region, and PGD may be misled by the optimization goal to a benign region. It would be an interesting future work to study these intriguing properties of semantic attacks.

E.8.8 Empirical Robustness against Unforeseen Attacks: Evaluation Protocol and Result Breakdown

In the main text (Section 7.7.2), we briefly show that TSS is generalizable to defend against unforeseen physical attacks by evaluating on CIFAR-10-C and ImageNet-C. Here, we first introduce the detailed evaluation protocol, then a breakdown of the result table (Table 7.4) in the main text and show empirical accuracy on each type of corruption.

Evaluated Models

On either CIFAR-10-C and ImageNet-C, we choose three models for evaluation: the vanilla model, the AugMix [146] trained model, and our TSS model for defending the composition of Gaussian blur, translation, brightness, and contrast. The vanilla models and TSS models are the same models as used in main experiments. The AugMix is the state-of-the-art empirical

Table E.3: Comparison between empirical robust accuracy against random and adaptive attacks and certified robust accuracy on **MNIST**. The attack radii are consistent with Table 7.3. The most powerful attack in each setting is highlighted in bold font. The adaptive attacks are shown in gray rows. Note that PGD attack cannot apply to translation transformation because the parameter space is discrete.

Transformation	Attack Radius	Model	Empirical Robust Accuracy					Certified
			Attack	Initial Starts $N =$			Robust	
				10	20	50	100	Accuracy	
Gaussian Blur	Squared Radius $\alpha \leqslant 36$	**TSS**	Random	93.2%	92.2%	92.0%	91.4%	90.6%
			Random+	92.4%	92.2%	**91.2%**	**91.2%**	
			PGD	**91.6%**	**91.6%**	91.6%	91.6%	
		Vanilla	Random	14.0%	12.4%	**12.2%**	**12.2%**	-
			Random+	12.4%	12.4%	**12.2%**	**12.2%**	
			PGD	**12.2%**	**12.2%**	**12.2%**	**12.2%**	
Translation (Reflection Pad.)	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 8$	**TSS**	Random	**99.6%**	**99.6%**	**99.6%**	**99.6%**	99.6%
			Random+	**99.6%**	**99.6%**	**99.6%**	**99.6%**	
			PGD	-	-	-	-	
		Vanilla	Random	**0.0%**	**0.0%**	**0.0%**	**0.0%**	-
			Random+	**0.0%**	**0.0%**	**0.0%**	**0.0%**	
			PGD	-	-	-	-	
Brightness	$b \pm 50\%$	**TSS**	Random	**98.2%**	**98.2%**	**98.2%**	**98.2%**	98.2%
			Random+	**98.2%**	**98.2%**	**98.2%**	**98.2%**	
			PGD	**98.2%**	**98.2%**	**98.2%**	**98.2%**	
		Vanilla	Random	97.2%	**96.6%**	**96.6%**	**96.6%**	-
			Random+	96.8%	**96.6%**	**96.6%**	**96.6%**	
			PGD	**96.6%**	**96.6%**	**96.6%**	**96.6%**	
Contrast and Brightness	$c \pm 50\%$, $b \pm 50\%$	**TSS**	Random	**98.0%**	**98.0%**	**98.0%**	**98.0%**	97.6%
			Random+	**98.0%**	**98.0%**	**98.0%**	**98.0%**	
			PGD	**98.0%**	**98.0%**	**98.0%**	**98.0%**	
		Vanilla	Random	96.8%	95.8%	95.0%	94.6%	-
			Random+	95.8%	94.4%	93.8%	93.6%	
			PGD	**93.6%**	**93.4%**	**93.2%**	**93.2%**	
Gaussian Blur, Translation Contrast, and Brightness	$\alpha \leqslant 5$, $c \pm 10\%$, $b \pm 10\%$, $\sqrt{\Delta x^2 + \Delta y^2} \leqslant 5$	**TSS**	Random	97.6%	97.6%	97.6%	97.2%	90.2%
			Random+	97.6%	**97.2%**	**97.0%**	**97.0%**	
			PGD	**97.4%**	97.4%	97.2%	**97.0%**	
		Vanilla	Random	10.0%	4.4%	1.4%	**0.4%**	-
			Random+	**6.8%**	2.4%	1.2%	**0.4%**	
			PGD	7.0%	**1.4%**	**0.8%**	**0.4%**	
Rotation	$r \pm 50^\circ$	**TSS**	Random	98.6%	**98.4%**	**98.2%**	98.4%	97.4%
			Random+	98.6%	98.6%	98.4%	**98.2%**	
			PGD	**98.2%**	**98.4%**	98.4%	**98.2%**	
		Vanilla	Random	27.2%	17.4%	13.8%	12.2%	-
			Random+	**15.4%**	**13.0%**	**11.0%**	**11.0%**	
			PGD	16.4%	15.6%	15.4%	15.2%	
Scaling	$s \pm 30\%$	**TSS**	Random	**99.2%**	**99.2%**	**99.2%**	**99.2%**	97.2%
			Random+	**99.2%**	**99.2%**	**99.2%**	**99.2%**	
			PGD	**99.2%**	**99.2%**	**99.2%**	**99.2%**	
		Vanilla	Random	92.0%	91.4%	90.2%	90.2%	-
			Random+	**90.0%**	**89.4%**	**89.2%**	**89.2%**	
			PGD	90.4%	90.2%	90.2%	90.2%	
Rotation and Brightness	$r \pm 50\%$, $b \pm 20\%$	**TSS**	Random	98.8%	98.4%	98.2%	98.2%	97.0%
			Random+	98.6%	98.2%	**98.0%**	98.2%	
			PGD	**98.2%**	**98.0%**	**98.0%**	**98.0%**	
		Vanilla	Random	28.8%	17.8%	12.6%	11.0%	-
			Random+	16.6%	**11.6%**	**10.4%**	**10.4%**	
			PGD	**13.4%**	13.6%	13.0%	12.6%	
Scaling and Brightness	$s \pm 50\%$, $b \pm 50\%$	**TSS**	Random	98.6%	98.6%	98.4%	**97.8%**	96.6%
			Random+	98.4%	98.0%	**97.8%**	**97.8%**	
			PGD	**98.2%**	**97.8%**	**97.8%**	**97.8%**	
		Vanilla	Random	57.4%	46.0%	31.0%	24.8%	-
			Random+	40.4%	28.0%	**19.8%**	**15.6%**	
			PGD	**29.0%**	**25.2%**	25.0%	24.0%	
Rotation, Brightness, and ℓ_2	$r \pm 50\%$, $b \pm 20\%$, $\|\delta\|_2 \leqslant .05$	**TSS**	Random	98.2%	97.8%	**97.6%**	97.6%	96.6%
			Random+	98.4%	98.0%	97.8%	97.6%	
			PGD	**97.6%**	**97.6%**	**97.6%**	**97.4%**	
		Vanilla	Random	27.6%	17.2%	11.4%	10.8%	-
			Random+	15.2%	**11.2%**	**9.4%**	**9.0%**	
			PGD	**13.4%**	11.8%	12.0%	11.8%	
Scaling, Brightness, and ℓ_2	$s \pm 50\%$, $b \pm 50\%$, $\|\delta\|_2 \leqslant .05$	**TSS**	Random	98.4%	98.4%	**97.6%**	**97.6%**	96.4%
			Random+	**97.8%**	97.8%	**97.6%**	**97.6%**	
			PGD	**97.8%**	**97.6%**	**97.6%**	**97.6%**	
		Vanilla	Random	50.4%	38.2%	28.2%	22.2%	-
			Random+	34.4%	23.2%	**13.4%**	**12.2%**	
			PGD	**23.4%**	**22.0%**	21.6%	20.8%	

Table E.4: Comparison between empirical robust accuracy against random and adaptive attacks and certified robust accuracy on **CIFAR-10**. The attack radii are consistent with Table 7.3. The most powerful attack in each setting is highlighted in bold font. The adaptive attacks are shown in gray rows. Note that the PGD attack cannot apply to translation transformation because the parameter space is discrete.

Transformation	Attack Radius	Model	Attack	Empirical Robust Accuracy (Initial Starts N = 10)	(20)	(50)	(100)	Certified Robust Accuracy
Gaussian Blur	Squared Radius $\alpha \leqslant 16$	**TSS**	Random	66.4%	66.4%	**65.8%**	**65.8%**	63.6%
			Random+	66.8%	66.0%	**65.8%**	**65.8%**	
			PGD	**65.8%**	**65.8%**	**65.8%**	**65.8%**	
		Vanilla	Random	4.8%	4.2%	**3.4%**	**3.4%**	-
			Random+	4.6%	4.0%	3.6%	**3.4%**	
			PGD	**3.4%**	**3.4%**	**3.4%**	**3.4%**	
Translation (Reflection Pad.)	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 20$	**TSS**	Random	**86.2%**	**86.0%**	86.2%	86.2%	80.8%
			Random+	86.4%	**86.0%**	**86.0%**	**86.0%**	
			PGD	-	-	-	-	
		Vanilla	Random	**8.0%**	**7.0%**	4.4%	**4.2%**	-
			Random+	8.2%	7.2%	**4.2%**	**4.2%**	
			PGD	-	-	-	-	
Brightness	$b \pm 40\%$	**TSS**	Random	87.2%	87.2%	87.4%	**87.2%**	87.0%
			Random+	**87.0%**	**87.0%**	**87.0%**	87.4%	
			PGD	87.4%	87.4%	87.4%	87.4%	
		Vanilla	Random	57.8%	51.2%	45.8%	44.4%	-
			Random+	**49.8%**	**44.2%**	**42.8%**	**42.6%**	
			PGD	52.4%	51.0%	50.8%	50.8%	
Contrast and Brightness	$c \pm 40\%$, $b \pm 40\%$	**TSS**	Random	86.2%	86.2%	86.2%	86.0%	82.4%
			Random+	**85.8%**	86.2%	86.0%	**85.8%**	
			PGD	**85.8%**	**85.8%**	**85.8%**	**85.8%**	
		Vanilla	Random	48.0%	40.0%	27.2%	21.0%	-
			Random+	32.0%	23.2%	14.8%	**9.6%**	
			PGD	**17.0%**	**13.0%**	**12.2%**	11.8%	
Gaussian Blur, Translation, Contrast, and Brightness	$\alpha \leqslant 1$, $c \pm 10\%$, $b \pm 10\%$, $\sqrt{\Delta x^2 + \Delta y^2} \leqslant 5$	**TSS**	Random	71.0%	**69.2%**	**68.0%**	**67.6%**	58.2%
			Random+	70.6%	69.8%	68.4%	67.8%	
			PGD	**69.8%**	69.8%	69.0%	68.0%	
		Vanilla	Random	21.2%	16.6%	12.0%	9.6%	-
			Random+	18.6%	14.2%	9.0%	7.2%	
			PGD	**12.8%**	**9.8%**	**6.8%**	**5.6%**	
Rotation	$r \pm 10°$	**TSS**	Random	78.0%	77.0%	76.8%	76.6%	70.6%
			Random+	77.4%	**76.8%**	**76.4%**	**76.4%**	
			PGD	**76.8%**	**76.8%**	76.8%	76.6%	
		Vanilla	Random	69.2%	68.0%	**65.6%**	65.6%	-
			Random+	68.4%	67.2%	66.0%	65.6%	
			PGD	**66.4%**	**66.0%**	**65.6%**	**65.4%**	
	$r \pm 30°$	**TSS**	Random	71.8%	70.2%	69.8%	**69.2%**	63.6%
			Random+	71.0%	**69.4%**	**69.2%**	69.4%	
			PGD	**70.4%**	70.0%	70.0%	69.8%	
		Vanilla	Random	31.6%	27.4%	22.6%	21.6%	-
			Random+	32.2%	27.2%	23.8%	**21.4%**	
			PGD	**25.2%**	**23.8%**	**23.2%**	23.2%	
Scaling	$s \pm 30\%$	**TSS**	Random	69.6%	67.8%	67.8%	67.2%	58.8%
			Random+	69.2%	68.4%	67.4%	**67.0%**	
			PGD	**67.8%**	**67.6%**	**67.2%**	**67.0%**	
		Vanilla	Random	60.0%	54.6%	52.8%	51.6%	-
			Random+	56.6%	53.8%	52.2%	**51.2%**	
			PGD	**53.2%**	**52.4%**	**52.0%**	52.0%	
Rotation and Brightness	$r \pm 10°$, $b \pm 10\%$	**TSS**	Random	77.2%	76.8%	77.0%	76.6%	70.6%
			Random+	77.2%	**76.6%**	**76.4%**	**76.0%**	
			PGD	**76.6%**	**76.6%**	**76.4%**	76.4%	
		Vanilla	Random	67.2%	64.8%	60.6%	59.4%	-
			Random+	66.0%	63.0%	59.4%	57.8%	
			PGD	**57.8%**	**57.6%**	**57.0%**	**56.8%**	
	$r \pm 30°$, $b \pm 20\%$	**TSS**	Random	72.0%	70.2%	68.8%	68.4%	61.4%
			Random+	70.6%	68.8%	**68.0%**	**68.2%**	
			PGD	**69.2%**	**68.6%**	68.6%	68.6%	
		Vanilla	Random	26.6%	20.2%	15.8%	13.0%	-
			Random+	18.8%	16.0%	11.6%	9.4%	
			PGD	**12.2%**	**10.4%**	**9.2%**	**9.0%**	
Scaling and Brightness	$s \pm 30\%$, $b \pm 30\%$	**TSS**	Random	68.6%	68.6%	67.4%	67.2%	54.2%
			Random+	68.4%	68.0%	67.0%	**66.8%**	
			PGD	**67.4%**	**67.4%**	**66.8%**	**66.8%**	
		Vanilla	Random	39.2%	30.6%	20.0%	17.4%	-
			Random+	30.4%	19.4%	15.4%	**11.6%**	
			PGD	**16.0%**	**14.4%**	**13.0%**	13.0%	
Rotation, Brightness, and ℓ_2	$r \pm 10°$, $b \pm 10\%$, $\|\delta\|_2 \leqslant .05$	**TSS**	Random	74.2%	72.8%	71.8%	71.6%	64.2%
			Random+	72.8%	72.2%	71.8%	**71.2%**	
			PGD	**71.6%**	**71.6%**	**71.6%**	71.6%	
		Vanilla	Random	40.4%	35.8%	34.4%	31.8%	-
			Random+	36.4%	**34.6%**	**30.8%**	**29.6%**	
			PGD	**36.0%**	35.0%	34.6%	34.6%	
	$r \pm 30°$, $b \pm 20\%$, $\|\delta\|_2 \leqslant .05$	**TSS**	Random	67.6%	66.2%	64.8%	65.2%	55.2%
			Random+	65.6%	65.6%	65.2%	64.4%	
			PGD	**65.2%**	**64.6%**	**64.0%**	**64.0%**	
		Vanilla	Random	7.6%	5.4%	2.6%	0.8%	-
			Random+	3.8%	2.4%	1.2%	**0.4%**	
			PGD	**1.2%**	**0.6%**	**0.6%**	0.6%	
Scaling, Brightness, and ℓ_2	$s \pm 30\%$, $b \pm 30\%$, $\|\delta\|_2 \leqslant .05$	**TSS**	Random	67.6%	66.8%	65.2%	65.0%	51.2%
			Random+	66.0%	66.2%	64.6%	64.4%	
			PGD	**64.2%**	**62.2%**	**61.8%**	**61.8%**	
		Vanilla	Random	15.6%	11.4%	5.8%	4.4%	-
			Random+	8.2%	5.0%	**2.0%**	**2.0%**	
			PGD	**3.8%**	**2.8%**	2.8%	2.6%	

defense on the CIFAR-10-C and ImageNet-C dataset according to [86]. For AugMix, on CIFAR-10-C, since the model weights are not released, we use the official implementation of AugMix (`https://github.com/google-research/augmix`) and extend the code to support our used model architecture (ResNet-110) for a fair comparison. We run the code with the suggested hyperparameters and achieve similar performance as reported in their paper. On ImageNet-C, we directly use the officially released model weights. The model has the same architecture (ResNet-50) as ours so the comparison is naturally fair. Note that all these models are trained only on the clean CIFAR-10 or ImageNet training set. We do not include the evaluation of MNIST models since there is no corrupted MNIST dataset available within our best knowledge.

Empirical Accuracy Computation

We compute the **empirical accuracy** (on CIFAR-10-C/ImageNet-C) as the ratio of correctly predicted samples among the test samples, where the test samples are all corrupted at the highest severity level (level 5) to model the strongest unforeseen semantic attacker. For each corruption type, there is a full test set generated by 1-to-1 mapping from the original clean test set samples processed with the corruption. Being consistent with the main experiment's protocol, for each corruption type, we uniformly pick 500 samples from the corresponding test set. Then, we compute the average empirical accuracy among all 19 corruptions and report it in Table 7.4 in main text.

As a reference, we also include their certified accuracy against the composition of Gaussian blur, brightness, contrast, and translation. Since our TSS provides robustness certification only for smoothed models, we apply the same smoothing strategy as our TSS models, hoping for providing robustness certificates for baseline models. As shown in Table 7.4, only TSS models can be certified with nontrivial certified robust accuracy.

Breakdown

In Table E.6, we show the breakdown of empirical accuracy for all models evaluated in Table 7.4. Note that our TSS models are trained using only four of these 19 corruptions (brightness, contrast, Gaussian blur, and additive Gaussian noise). Almost on all the corruptions, TSS has higher accuracy than vanilla models and sometimes higher than the state-of-the-art defense—AugMix. Interestingly, we find TSS models have different generalization abilities on these corruptions. The additive Gaussian noise has the best generalization ability, because TSS model also achieves much higher accuracy against impulse noise and

shot noise than all the baselines. The Gaussian blur also generalizes well, because we can see significantly higher accuracy of TSS models against zoom blur, glass blur, motion blur, and defocus blur especially on CIFAR-10-C. Finally, brightness and contrast, even though they seem to be among the simplest transformations, have the poorest generalization ability. For example, under severe corruptions, the empirical accuracy of brightness is even below than that of vanilla models. Manual inspection of the corrupted images showed that corrupted brightness or contrast images are severely altered so that they are hard to be distinguished even by humans, giving a hint for possible reasons for the poor performance on these images. We thus conjecture that overly severe corruption could be the reason, and we think that it would be an interesting future direction to study these different generalization abilities in depth.

E.8.9 Certified Accuracy Beyond Certified Radii

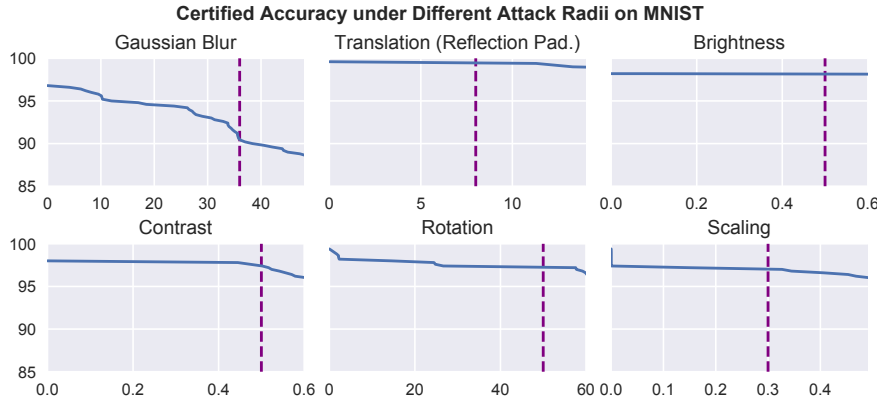


Figure E.2: The blue curves show the certified robust accuracy on **MNIST**. The predefined certified radii are shown as purple vertical dotted lines. **No significant degradation after exceeding the predefined radii**. For Contrast subfigure, we allow additional 50% brightness change.

In Figure E.2 and Figure E.3, on MNIST and CIFAR-10, the purple vertical dotted lines stand for the predefined certified radii that the models aim to defend, and the blue curves show the certified robust accuracy (y axis) with respect to attack radii (x axis). The figures imply that the TSS models that aim to defend against transformations within certain thresholds still maintain **high certified accuracy** when the transformation parameters go even far beyond the thresholds.

In Table E.7, Table E.8, and Table E.9, we further list the empirical robust accuracy of TSS and vanilla models when the attacker goes beyond the predefined certified radius. The

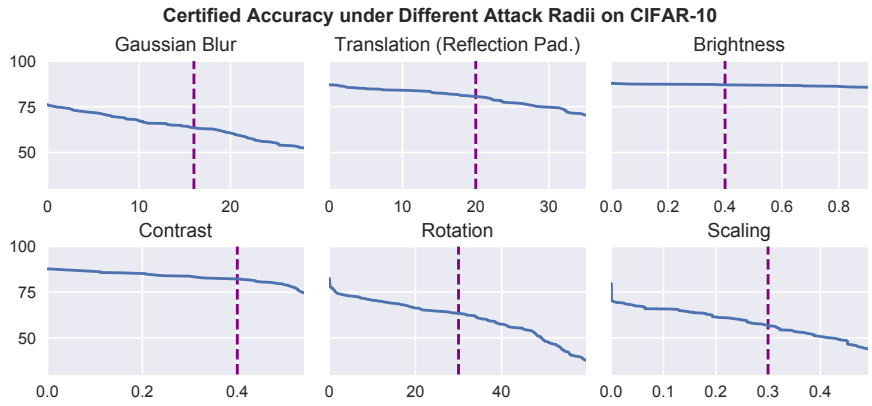


Figure E.3: The blue curves show the certified robust accuracy on **CIFAR-10**. The predefined certified radii are shown as purple vertical dotted lines. **No significant degradation after exceeding the predefined certified radii**. For Contrast subfigure, we allow additional 40% brightness change.

empirical robust accuracy is computed as the minimum among all three attacks: Random, Random+, and PGD. We observe that the empirical robust accuracy follows the same tendency. For example, on CIFAR-10 dataset, the TSS model is trained to defend against the rotation transformation within 30° where it achieves 69.2%/63.6% empirical/certified accuracy. When the rotation angle goes up to 60° the model still preserves 46.8%/37.4% empirical/certified accuracy. On the contrary, the vanilla model's empirical accuracy is reduced from 21.4% (30° rotation) to 3.2% (60° rotation).

E.8.10 Different Smoothing Variance Levels: More Results

In Section 7.7.3 we have shown the study on smoothing variance levels on ImageNet (Table 7.5). Here, we further present our study of smoothing variance levels on MNIST and CIFAR-10. They are shown in Table E.10 and Table E.11 respectively. The smoothing variances shown in the two tables are for both training and inference-time smoothing. Except for smoothing variance, all other hyperparameters for training and certification are kept the same and consistent with the main experiments (Table 7.3). As we can observe, the same conclusion still holds: usually, when the smoothing variance increases, the benign accuracy drops and the certified robust accuracy first rises and then drops. The reason is that larger smoothing variance makes the input more severely transformed so that the benign accuracy becomes smaller. On the other hand, larger smoothing variance makes the robustness easier to be certified as we can observe in various robustness conditions in Appendix E.2, where the required lower bound of p_A becomes smaller. This is the reason for the "first rise" on

certified accuracy. However, when the smoothing variance becomes too large, the benign accuracy becomes too low, and according to our definition, the certified accuracy is upper bounded by the benign accuracy (precondition of robustness is correctness). This is the reason for the "then drop" on certified accuracy.

We again observe that the range of acceptable variance is usually wide. For example, on CIFAR-10, for rotation transformation, the certified robust accuracy is 63.6%/62.0%/59.0% across a wide range of smoothing variance: 0.05, 0.09, 0.12. Thus, even in the presence of such trade-off, without fine-tuning the smoothing variances, we can still obtain high certified robust accuracy and high benign accuracy as reported in Table 7.3 and Table E.1 respectively.

We remark that the gray cells in Table E.10 and Table E.11 indicate the smoothing variances used in our main experiments. We did not tune the smoothing variances so these cells might be sub-optimal (though usually close to optimal) and they are just placed here for indication purpose.

E.8.11 Tightness-Efficiency Trade-Off

We notice that as we increase the number of samples when estimating the interpolation error in (7.27) and (7.30), the interpolation error $M_{\mathcal{S}}$ and the upper bound $\sqrt{M} \geqslant M_{\mathcal{S}}$ become smaller and the certification becomes tighter, leading to higher certified robust accuracy. However, the computation time is also increased, resulting in a trade-off between speed and accuracy. In Table E.12 and Table E.13, we illustrate this trade-off on two differentially resolvable transformations: composition of rotation and brightness on CIFAR-10, and composition of scaling and brightnes on MNIST. From the tables, we find that, for these compositions, as the sample numbers N and n increase, the interpolation error decreases and computing time increases (linearly with N and n). As a consequence, if using a large number of samples, we can decrease the smoothing noise level σ and achieve both higher certified accuracy and higher benign accuracy at the cost of larger computation time.

Table E.5: Comparison between empirical robust accuracy against random and adaptive attacks and certified robust accuracy on **ImageNet**. The attack radii are consistent with Table 7.3. The most powerful attack in each setting is highlighted in bold font. The adaptive attacks are shown in gray rows. Note that PGD attack cannot apply to translation transformation because the parameter space is discrete.

Transformation	Attack Radius	Model	Attack	Empirical Robust Accuracy (Initial Starts N = 10)	(20)	(50)	(100)	Certified Robust Accuracy
Gaussian Blur	Squared Radius $\alpha \leqslant 36$	**TSS**	Random	53.2%	**52.8%**	**52.8%**	52.8%	51.6%
			Random+	53.2%	**52.8%**	**52.8%**	52.8%	
			PGD	**52.8%**	**52.8%**	**52.8%**	**52.6%**	
		Vanilla	Random	9.6%	8.6%	8.4%	8.4%	-
			Random+	8.8%	**8.2%**	**8.2%**	**8.2%**	
			PGD	**8.4%**	**8.2%**	**8.2%**	**8.2%**	
Translation (Reflection Pad.)	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 100$	**TSS**	Random	70.0%	69.6%	**69.2%**	**69.2%**	50.0%
			Random+	**69.4%**	**69.2%**	**69.2%**	**69.2%**	
			PGD	-	-	-	-	
		Vanilla	Random	**55.8%**	**53.4%**	**48.8%**	46.6%	-
			Random+	57.2%	54.6%	50.6%	**46.2%**	
			PGD	-	-	-	-	
Brightness	$b \pm 40\%$	**TSS**	Random	70.8%	**70.4%**	**70.4%**	**70.4%**	70.0%
			Random+	**70.4%**	**70.4%**	**70.4%**	**70.4%**	
			PGD	**70.4%**	**70.4%**	**70.4%**	**70.4%**	
		Vanilla	Random	31.6%	26.6%	21.6%	19.6%	-
			Random+	22.8%	**19.8%**	**18.4%**	**18.4%**	
			PGD	**22.0%**	22.4%	21.8%	21.8%	
Contrast and Brightness	$c \pm 40\%$, $b \pm 40\%$	**TSS**	Random	70.4%	69.2%	**68.4%**	**68.4%**	61.4%
			Random+	69.2%	68.8%	**68.4%**	**68.4%**	
			PGD	**68.4%**	**68.4%**	**68.4%**	**68.4%**	
		Vanilla	Random	20.8%	10.4%	3.6%	1.2%	-
			Random+	8.0%	2.0%	0.4%	**0.0%**	
			PGD	**1.8%**	**0.2%**	**0.0%**	**0.0%**	
Gaussian Blur, Translation Contrast, and Brightness	$\alpha \leqslant 10$, $c \pm 20\%$, $b \pm 20\%$, $\sqrt{\Delta x^2 + \Delta y^2} \leqslant 10$	**TSS**	Random	51.8%	50.2%	49.2%	48.8%	32.8%
			Random+	51.4%	**49.6%**	**48.0%**	48.2%	
			PGD	**49.6%**	**49.6%**	48.2%	**47.4%**	
		Vanilla	Random	20.6%	17.4%	12.0%	9.4%	-
			Random+	15.2%	12.8%	7.8%	6.6%	
			PGD	**11.2%**	**8.0%**	**6.0%**	**4.0%**	
Rotation	$r \pm 30\%$	**TSS**	Random	40.2%	**38.4%**	38.4%	**37.8%**	30.4%
			Random+	**39.0%**	38.6%	**38.0%**	**37.8%**	
			PGD	40.4%	39.8%	39.8%	39.4%	
		Vanilla	Random	47.8%	44.4%	41.4%	40.0%	-
			Random+	45.0%	43.6%	40.6%	38.8%	
			PGD	**39.6%**	**38.4%**	**37.8%**	**37.0%**	
Scaling	$s \pm 30\%$	**TSS**	Random	40.2%	38.0%	37.4%	37.4%	26.4%
			Random+	**38.8%**	**37.2%**	**36.8%**	**36.4%**	
			PGD	39.0%	37.8%	37.6%	37.8%	
		Vanilla	Random	55.2%	53.0%	51.2%	50.0%	-
			Random+	55.6%	52.8%	50.6%	50.0%	
			PGD	**50.6%**	**49.8%**	**49.4%**	**49.8%**	
Rotation and Brightness	$r \pm 30^\circ$ $b \pm 20\%$	**TSS**	Random	**38.8%**	**38.0%**	37.2%	37.4%	26.8%
			Random+	39.0%	38.2%	**37.0%**	**36.8%**	
			PGD	39.6%	39.4%	38.6%	38.8%	
		Vanilla	Random	40.4%	35.4%	29.2%	22.4%	-
			Random+	35.2%	31.2%	25.2%	**21.2%**	
			PGD	**25.0%**	**23.2%**	**22.2%**	21.4%	
Scaling and Brightness	$s \pm 30\%$, $b \pm 30\%$	**TSS**	Random	40.2%	38.0%	**36.4%**	36.4%	23.4%
			Random+	38.0%	**37.0%**	36.6%	**36.0%**	
			PGD	**37.0%**	**37.0%**	36.6%	36.6%	
		Vanilla	Random	34.4%	26.2%	19.4%	16.0%	-
			Random+	21.0%	**15.0%**	**12.4%**	**8.8%**	
			PGD	**17.6%**	15.2%	13.8%	13.4%	
Rotation, Brightness, and ℓ_2	$r \pm 30^\circ$, $b \pm 20\%$, $\|\delta\|_2 \leqslant .05$	**TSS**	Random	39.4%	38.2%	37.8%	37.0%	26.6%
			Random+	**38.2%**	**37.8%**	**36.6%**	**36.4%**	
			PGD	38.8%	38.8%	38.4%	38.0%	
		Vanilla	Random	26.0%	23.2%	19.8%	17.6%	-
			Random+	21.4%	18.4%	16.0%	14.4%	
			PGD	**16.6%**	**14.6%**	**14.2%**	**14.0%**	
Scaling, Brightness, and ℓ_2	$s \pm 30\%$, $b \pm 30\%$, $\|\delta\|_2 \leqslant .05$	**TSS**	Random	40.2%	38.2%	37.2%	36.0%	22.6%
			Random+	38.0%	**36.4%**	**35.8%**	**35.6%**	
			PGD	**36.8%**	**36.4%**	36.4%	36.0%	
		Vanilla	Random	24.4%	17.2%	11.4%	7.4%	-
			Random+	13.8%	**8.4%**	**5.8%**	**4.8%**	
			PGD	**9.8%**	8.8%	7.4%	7.4%	

Table E.6: Comparison of Empirical Accuracy for each corruption evaluated from the highest severity level (5) of CIFAR-10-C and ImageNet-C.

Corruption		CIFAR-10			ImageNet		
Category	Type	Vanilla	AugMix [146]	TSS	Vanilla	AugMix [146]	TSS
Weather	Snow	68.2%	**75.6%**	69.4%	16.0%	**22.6%**	13.8%
	Fog	63.4%	**65.4%**	62.0%	**24.0%**	22.2%	18.0%
	Frost	59.2%	67.8%	**73.8%**	21.6%	**24.8%**	22.6%
	Brightness	**82.4%**	**82.4%**	71.8%	**56.8%**	56.6%	35.8%
Blur	Zoom Blur	52.6%	70.8%	**75.2%**	21.4%	**31.0%**	20.4%
	Glass Blur	46.6%	50.2%	**72.2%**	8.0%	**14.0%**	13.8%
	Motion Blur	54.8%	68.6%	**70.2%**	14.2%	**25.2%**	11.4%
	Defocus Blur	49.0%	72.2%	**75.6%**	14.0%	22.6%	**25.6%**
Noise	Impulse Noise	29.8%	**51.0%**	46.2%	4.0%	9.8%	**12.0%**
	Gaussian Noise	34.8%	56.4%	**62.8%**	4.4%	9.6%	**12.8%**
	Shot Noise	43.0%	**63.4%**	62.6%	4.0%	13.0%	**14.0%**
Digital	Pixelate	42.0%	59.0%	**76.0%**	19.6%	39.2%	**55.6%**
	Elastic Transform	71.4%	65.2%	**74.4%**	14.8%	**23.8%**	23.6%
	Contrast	23.8%	26.0%	**49.8%**	4.2%	**11.6%**	5.0%
	JPEG Compression	70.8%	**73.0%**	71.8%	33.6%	**45.4%**	31.6%
Extra	Saturate	79.6%	**83.4%**	63.6%	41.6%	**43.4%**	25.8%
	Spatter	72.8%	**82.0%**	69.0%	22.4%	**30.6%**	17.6%
	Speckle Noise	45.2%	**64.0%**	58.8%	11.4%	**27.4%**	23.6%
	Gaussian Blur	34.6%	67.4%	**75.8%**	11.2%	15.2%	**33.0%**
	Average	53.89%	65.46%	**67.42%**	18.27%	**25.68%**	21.89%

Table E.7: Empirical and certified robust accuracy on **MNIST** when attack radii go beyond the predefined one. The predefined certified radii are consistent with Table 7.3.
* For Contrast we allow additional ±50% brightness change since a single Contrast attack is not powerful enough.

				Beyond Predefined Radii		
Gaussian Blur		Radii:	36	40	44	48
	TSS	Empirical	91.2%	90.8%	90.4%	89.2%
		Certified	90.6%	90.0%	89.6%	88.8%
	Vanilla	Empirical	12.2%	11.8%	11.2%	11.2%
Translation (Reflection Pad.)		Radii:	8	10	12	14
	TSS	Empirical	99.6%	99.6%	99.6%	99.6%
		Certified	99.6%	99.6%	99.4%	99.0%
	Vanilla	Empirical	0.0%	0.0%	0.0%	0.0%
Brightness		Radii:	50%	52%	55%	60%
	TSS	Empirical	98.2%	98.2%	98.2%	98.2%
		Certified	98.2%	98.2%	98.2%	98.2%
	Vanilla	Empirical	96.6%	96.2%	95.6%	94.4%
Contrast*		Radii:	50%	52%	55%	60%
	TSS	Empirical	98.0%	98.0%	98.0%	98.0%
		Certified	97.6%	97.2%	96.8%	96.2%
	Vanilla	Empirical	93.2%	93.2%	93.2%	93.0%
Rotation		Radii:	50°	52°	55°	60°
	TSS	Empirical	98.2%	98.2%	98.2%	97.8%
		Certified	97.4%	97.4%	97.4%	96.6%
	Vanilla	Empirical	11.0%	9.8%	8.4%	7.2%
Scaling		Radii:	30%	35%	40%	50%
	TSS	Empirical	99.2%	98.8%	98.8%	98.6%
		Certified	97.2%	96.8%	96.8%	96.0%
	Vanilla	Empirical	89.2%	82.6%	72.8%	45.4%

Table E.8: Empirical and certified robust accuracy on **CIFAR-10** when attack radii go beyond the predefined one. The predefined certified radii are consistent with Table 7.3.
* For Contrast we allow additional 40% brightness change since a single Contrast attack is not powerful enough.

				Beyond Predefined Radii		
Gaussian Blur		Radii:	16	20	24	28
	TSS	Empirical	65.8%	63.4%	61.2%	56.4%
		Certified	63.6%	60.8%	56.0%	52.6%
	Vanilla	Empirical	3.4%	3.2%	3.0%	2.8%
Tranlation (Reflection Pad.)		Radii:	20	25	30	35
	TSS	Empirical	86.0%	86.0%	85.8%	85.8%
		Certified	80.8%	77.4%	74.8%	70.6%
	Vanilla	Empirical	4.2%	3.6%	3.6%	2.8%
Brightness		Radii:	40%	45%	50%	55%
	TSS	Empirical	87.2%	87.0%	87.0%	87.0%
		Certified	87.0%	87.0%	87.0%	87.0%
	Vanilla	Empirical	42.6%	32.8%	21.2%	14.0%
Contrast*		Radii	40%	45%	50%	55%
	TSS	Empirical	85.8%	85.8%	85.4%	85.2%
		Certified	82.4%	80.8%	79.2%	71.8%
	Vanilla	Empirical	9.6%	7.8%	5.6%	4.8%
Rotation		Radii:	30°	40°	50°	60°
	TSS	Empirical	69.2%	64.0%	57.8%	46.8%
		Certified	63.6%	57.6%	48.2%	37.4%
	Vanilla	Empirical	21.4%	8.8%	5.0%	3.2%
Scaling		Radii:	30%	35%	40%	50%
	TSS	Empirical	67.0%	65.0%	60.6%	54.8%
		Certified	58.8%	53.6%	51.0%	43.4%
	Vanilla	Empirical	51.2%	43.0%	34.8%	21.2%

Table E.9: Empirical and certified robust accuracy on **ImageNet** when attack radii go beyond the predefined one. The predefined certified radii are consistent with Table 7.3.
* For Contrast/Rotation/Scaling we allow additional $\pm 40\%/\pm 20\%/\pm 30\%$ brightness change since a single Contrast/Rotation/Scaling attack is not powerful enough.

				Beyond Predefined Radii	
Gaussian Blur		Radii:	36	40	44
	TSS	Empirical	52.6%	51.2%	49.8%
		Certified	51.6%	50.0%	48.8%
	Vanilla	Empirical	8.2%	7.2%	6.2%
Translation (Reflection Pad.)		Radii:	100	105	110
	TSS	Empirical	69.2%	69.2%	69.0%
		Certified	50.0%	49.4%	46.8%
	Vanilla	Empirical	46.2%	37.6%	36.6%
Brightness		Radii:	40%	45%	50%
	TSS	Empirical	70.4%	70.2%	70.0%
		Certified	70.0%	69.8%	69.6%
	Vanilla	Empirical	18.4%	10.0%	5.2%
Contrast*		Radii:	40%	45%	50%
	TSS	Empirical	68.4%	68.2%	67.6%
		Certified	61.4%	55.8%	45.0%
	Vanilla	Empirical	0.0%	0.0%	0.0%
Rotation*		Radii:	30°	35°	45°
	TSS	Empirical	36.8%	36.4%	33.4%
		Certified	26.8%	26.2%	21.8%
	Vanilla	Empirical	21.2%	19.4%	16.2%
Scaling*		Radii:	30%	40%	50%
	TSS	Empirical	36.0%	32.4%	26.6%
		Certified	23.4%	18.4%	11.6%
	Vanilla	Empirical	8.8%	8.8%	7.0%

Table E.10: Study of the impact of different smoothing variance levels on certified robust accuracy and benign accuracy on **MNIST** for TSS. The attack radii are consistent with Table 7.3. The "Dist." refers to both training and smoothing distribution. The variance used in Table 7.3 is labeled in gray.

Transformation	Attack Radius	Certified Accuracy and Benign Accuracy under Different Variance Levels			
Gaussian Blur	$\alpha \leqslant 36$	Dist. of α	Exp(1/5)	Exp(1/10)	Exp(1/20)
		Cert. Rob. Acc.	90.4%	**90.6%**	89.2%
		Benign Acc.	**97.0%**	96.8%	93.4%
Translation (Reflection Pad.)	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 8$	Dist. of $(\Delta x, \Delta y)$	$\mathcal{N}(0, 5^2 I)$	$\mathcal{N}(0, 10^2 I)$	$\mathcal{N}(0, 15^2 I)$
		Cert. Rob. Acc.	99.0%	**99.6%**	99.4%
		Benign Acc.	**99.6%**	**99.6%**	**99.6%**
Brightness	$b \pm 50\%$	Dist. of (c, b)	$\mathcal{N}(0, 0.5^2 I)$	$\mathcal{N}(0, 0.6^2 I)$	$\mathcal{N}(0, 0.7^2 I)$
		Cert. Rob. Acc.	**98.4%**	98.2%	**98.4%**
		Benign Acc.	**98.4%**	**98.4%**	**98.4%**
Contrast	$c \pm 50\%$	Dist. of (c, b)	$\mathcal{N}(0, 0.5^2 I)$	$\mathcal{N}(0, 0.6^2 I)$	$\mathcal{N}(0, 0.7^2 I)$
		Cert. Rob. Acc.	0.0%	98.0%	**98.4%**
		Benign Acc.	**98.4%**	**98.4%**	**98.4%**
Rotation	$r \pm 50°$	Dist. of ϵ	$\mathcal{N}(0, 0.05^2 I)$	$\mathcal{N}(0, 0.12^2 I)$	$\mathcal{N}(0, 0.20^2 I)$
		Cert. Rob. Acc.	**97.6%**	97.4%	**97.6%**
		Benign Acc.	99.2%	**99.4%**	99.2%
Scaling	$s \pm 30\%$	Dist. of ϵ	$\mathcal{N}(0, 0.05^2 I)$	$\mathcal{N}(0, 0.12^2 I)$	$\mathcal{N}(0, 0.20^2 I)$
		Cert. Rob. Acc.	96.6%	**97.2%**	96.0%
		Benign Acc.	**99.4%**	**99.4%**	99.0%

Table E.11: Study of the impact of different smoothing variance levels on certified robust accuracy and benign accuracy on **CIFAR-10** for TSS. The attack radii are consistent with Table 7.3. The "Dist." refers to both training and smoothing distribution. The variance used in Table 7.3 is labeled in gray.

Transformation	Attack Radius	Certified Accuracy and Benign Accuracy under Different Variance Levels			
Gaussian Blur	$\alpha \leqslant 16$	Dist. of α	Exp(1/5)	Exp(1/10)	Exp(1/20)
		Cert. Rob. Acc.	**63.6%**	60.6%	53.0%
		Benign Acc.	**76.2%**	68.0%	57.4%
Translation (Reflection Pad.)	$\sqrt{\Delta x^2 + \Delta y^2} \leqslant 20$	Dist. of $(\Delta x, \Delta y)$	$\mathcal{N}(0, 10^2 I)$	$\mathcal{N}(0, 15^2 I)$	$\mathcal{N}(0, 20^2 I)$
		Cert. Rob. Acc.	76.2%	**80.8%**	74.4%
		Benign Acc.	**89.0%**	87.0%	84.6%
Brightness	$b \pm 40\%$	Dist. of (c, b)	$\mathcal{N}(0, 0.2^2 I)$	$\mathcal{N}(0, 0.3^2 I)$	$\mathcal{N}(0, 0.4^2 I)$
		Cert. Rob. Acc.	**87.4%**	87.0%	86.2%
		Benign Acc.	**87.8%**	**87.8%**	86.4%
Contrast	$c \pm 40\%$	Dist. of (c, b)	$\mathcal{N}(0, 0.2^2 I)$	$\mathcal{N}(0, 0.3^2 I)$	$\mathcal{N}(0, 0.4^2 I)$
		Cert. Rob. Acc.	0.0%	**82.4%**	**82.4%**
		Benign Acc.	**87.8%**	**87.8%**	86.4%
Rotation	$r \pm 30°$	Dist. of ϵ	$\mathcal{N}(0, 0.05^2 I)$	$\mathcal{N}(0, 0.09^2 I)$	$\mathcal{N}(0, 0.12^2 I)$
		Cert. Rob. Acc.	**63.6%**	62.0%	59.0%
		Benign Acc.	**82.0%**	78.6%	72.2%
Scaling	$s \pm 30\%$	Dist. of ϵ	$\mathcal{N}(0, 0.05^2 I)$	$\mathcal{N}(0, 0.09^2 I)$	$\mathcal{N}(0, 0.12^2 I)$
		Cert. Rob. Acc.	59.0%	**59.4%**	58.8%
		Benign Acc.	**85.4%**	81.6%	79.2%

Table E.12: Average interpolation upper bound $\sqrt{M}$ (7.27), average computation time, and "Certified accuracy (average certification time)" for varying number of samples and smoothing noise levels. Results on **CIFAR-10** against the composition of **rotation** $\pm 10^\circ$ and **brightness** change $\pm 10\%$.

Number of Samples		Interpolation		Smoothing Noise Level σ		
First-Level	Second-Level	Avg. $\sqrt{M}$	Avg. Comp. Time	0.05	0.09	0.12
$N = 556$	$n = 2,000$	0.050	22.50 s	70.2% (62.32 s)	65.2% (86.60 s)	61.2% (53.73 s)
$N = 556$	$n = 200$	0.131	1.97 s	42.0% (490.21 s)	59.2% (93.19 s)	60.4% (86.60 s)
$N = 56$	$n = 2,000$	0.322	1.90 s	1.2% (6.18 s)	12.6% (16.64 s)	29.2% (25.77 s)
$N = 56$	$n = 200$	0.499	0.27 s	0.0% (5.22 s)	1.2% (5.68 s)	3.4% (8.49 s)
			Benign Accuracy:	83.0%	79.2%	79.6%

Table E.13: Average interpolation upper bound $\sqrt{M}$ (7.27), average bound computation time, and "Certified robust accuracy (average certification time)" when using different number of samples and various smoothing noise levels. Data is collected on **MNIST** dataset against the composition of **scaling** $\pm 50\%$ and **brightness** change $\pm 50\%$.

Number of Samples		Interpolation		Smoothing Noise Level σ		
First-Level	Second-Level	Avg. $\sqrt{M}$	Avg. Comp. Time	0.05	0.09	0.12
$N = 2,500$	$n = 500$	0.064	10.52 s	97.2% (92.36 s)	97.4% (76.25 s)	96.6% (67.44 s)
$N = 2,500$	$n = 50$	0.163	0.90 s	18.8% (157.48 s)	97.0% (217.97 s)	95.0% (97.91 s)
$N = 250$	$n = 500$	0.441	0.74 s	0.0% (0.80 s)	6.0% (4.91 s)	16.2% (12.48 s)
$N = 250$	$n = 50$	0.641	0.13 s	0.0% (0.79 s)	0.0% (0.71 s)	0.6% (1.60 s)
			Benign Accuracy:	99.4%	99.6%	99.4%

APPENDIX F: APPENDIX FOR CHAPTER 8

F.1 PROOFS

F.1.1 Proof of Lemma 8.1

We recall Lemma 8.1:

Lemma 8.1 (Lipschitz Continuity of Smoothed Value Function)**.** Given the action-value function $Q^\pi : \mathcal{S} \times \mathcal{A} \to [V_{\min}, V_{\max}]$, the smoothed function $\widetilde{Q}^\pi$ with smoothing parameter σ is L-Lipschitz continuous with $L = \frac{V_{\max} - V_{\min}}{\sigma}\sqrt{\frac{2}{\pi}}$ w.r.t. the state input.

Proof. To prove Lemma 8.1, we leverage the technique in the proof for Lemma 1 of Salman et al. [317] in their Appendix A.

For each action $a \in \mathcal{A}$, our smoothed value function is

$$\widetilde{Q}^\pi(s,a) := \mathop{\mathbb{E}}_{\Delta \sim \mathcal{N}(0,\sigma^2 \boldsymbol{I}_N)} Q^\pi(s+\Delta, a) = \frac{1}{(2\pi)^{N/2}\sigma^N} \int_{\mathbb{R}^n} Q^\pi(t,a) \exp\left(-\frac{1}{2\sigma^2}\|s-t\|^2\right) \mathrm{d}t. \tag{F.1}$$

Taking the gradient w.r.t. s, we obtain

$$\nabla_s \widetilde{Q}^\pi(s,a) = \frac{1}{(2\pi)^{N/2}\sigma^N} \int_{\mathbb{R}^n} Q^\pi(t,a) \frac{1}{\sigma^2}(s-t) \exp\left(-\frac{1}{2\sigma^2}\|s-t\|^2\right) \mathrm{d}t. \tag{F.2}$$

For any unit direction u, we have

$$\begin{aligned}
&u \cdot \nabla_s \widetilde{Q}^\pi(s,a) \\
&\leqslant \frac{1}{(2\pi)^{N/2}\sigma^N} \int_{\mathbb{R}^n} \frac{V_{\max} - V_{\min}}{\sigma^2} |u(s-t)| \exp\left(-\frac{1}{2\sigma^2}\|s-t\|^2\right) \mathrm{d}t && \text{(F.3)} \\
&= \frac{V_{\max} - V_{\min}}{\sigma^2} \int_{\mathbb{R}^n} \frac{1}{(2\pi)^{1/2}\sigma} |s_i - t| \exp\left(-\frac{1}{2\sigma^2}|s_i - t|^2\right) \mathrm{d}t \cdot \prod_{j \neq i} \int_{\mathbb{R}^n} \frac{1}{(2\pi)^{1/2}\sigma} \exp\left(-\frac{1}{2\sigma^2}|s_j - t|^2\right) \mathrm{d}t && \text{(F.4)} \\
&= \frac{V_{\max} - V_{\min}}{\sigma}\sqrt{\frac{2}{\pi}} && \text{(F.5)}
\end{aligned}$$

Thus, $\widetilde{Q}^\pi$ is L-Lipschitz continuous with $L = \frac{V_{\max} - V_{\min}}{\sigma}\sqrt{\frac{2}{\pi}}$ w.r.t. the state input. QED.

F.1.2 Proof of Theorem 8.1

We recall Theorem 8.1:

Theorem F.1. Let $Q^\pi : \mathcal{S} \times \mathcal{A} \to [V_{\min}, V_{\max}]$ be a trained value network, $\widetilde{Q}^\pi$ be the smoothed function with (8.3). At time step t with state s_t, we can compute the lower bound r_t of maximum perturbation magnitude $\bar{\epsilon}(s_t)$ (i.e., $r_t \leqslant \bar{\epsilon}(s_t)$, $\bar{\epsilon}$ defined in Definition 8.1) for locally smoothed policy $\tilde{\pi}$:

$$r_t = \frac{\sigma}{2}\left(\Phi^{-1}\left(\frac{\widetilde{Q}^\pi(s_t, a_1) - V_{\min}}{V_{\max} - V_{\min}}\right) - \Phi^{-1}\left(\frac{\widetilde{Q}^\pi(s_t, a_2) - V_{\min}}{V_{\max} - V_{\min}}\right)\right), \tag{F.6}$$

where Φ^{-1} is the inverse CDF function, a_1 is the action with the highest $\widetilde{Q}^\pi$ value at state s_t, and a_2 is the runner-up action. We name the lower bound r_t as certified radius for the state s_t.

We first present a lemma that can help in the proof of Theorem 8.1.

Lemma F.1. Let Φ be the CDF of a standard normal distribution, the mapping $\eta_a(s) := \sigma \cdot \Phi^{-1}\left(\frac{\widetilde{Q}^\pi(s,a) - V_{\min}}{V_{\max} - V_{\min}}\right)$ is 1-Lipschitz continuous.

The lemma can be proved following the same technique as the proof for Lemma 8.1 in Appendix F.1.1. The detailed proof can be referred to in the proof for Lemma 2 of Salman et al. [317] in their Appendix A. We next show how to leverage Lemma F.1 to prove Theorem 8.1.

Proof for Theorem 8.1. Let the perturbation be δ_t, based on the Lipschitz continuity of the mapping η, we have

$$\eta_{a_1}(s_t) - \eta_{a_1}(s_t + \delta_t) \leqslant \|\delta_t\|_2, \tag{F.7}$$

$$\eta_{a_2}(s_t + \delta_t) - \eta_{a_2}(s_t) \leqslant \|\delta_t\|_2. \tag{F.8}$$

Suppose that under perturbation δ_t, the action selection would be misled in the sense that the smoothed value for the original action a_1 is lower than that of another action a_2, i.e., $\widetilde{Q}^\pi(s_t + \delta_t, a_1) \leqslant \widetilde{Q}^\pi(s_t + \delta_t, a_2)$. Then, based on the monotonicity of η, we have

$$\eta_{a_1}(s_t + \delta) \leqslant \eta_{a_2}(s_t + \delta). \tag{F.9}$$

Summing up (F.7), (F.8), and (F.9), we obtain

$$\|\delta_t\|_2 \geqslant \frac{1}{2}\left(\eta_{a_1}(s_t) - \eta_{a_2}(s_t)\right) \tag{F.10}$$

$$= \frac{\sigma}{2}\left(\Phi^{-1}\left(\frac{\widetilde{Q}^{\pi}(s_t, a_1) - V_{\min}}{V_{\max} - V_{\min}}\right) - \Phi^{-1}\left(\frac{\widetilde{Q}^{\pi}(s_t, a_2) - V_{\min}}{V_{\max} - V_{\min}}\right)\right) \tag{F.11}$$

which is a lower bound of the *maximum perturbation magnitude* $\bar{\epsilon}(s_t)$ that can be tolerated at state s_t. Hence, when r_t takes the value of the computed lower bound, it satisfies the condition that $r_t \leqslant \bar{\epsilon}(s_t)$. QED.

F.1.3 Proof of Lemma 8.2

We recall Lemma 8.2:

Lemma 8.2 (Lipschitz Continuity of Smoothed Perturbed Return Function)**.** Let F be the perturbed return function function defined in (8.6), the smoothed perturbed return function $\widetilde{F}_{\pi}$ is $\frac{(J_{\max}-J_{\min})}{\sigma}\sqrt{\frac{2}{\pi}}$-Lipschitz continuous, where

$$\widetilde{F}_{\pi}\left(\oplus_{t=0}^{H-1}\delta_t\right) := \mathop{\mathbb{E}}_{\Delta\sim\mathcal{N}(0,\sigma^2 I_{H\times N})} F_{\pi}\left(\oplus_{t=0}^{H-1}(\delta_t + \Delta_t)\right). \tag{8.7}$$

Proof. The proof can be done in a similar fashion as the proof for Lemma 8.1 in Appendix F.1.1. Compared with the smoothed value network where the expectation is taken over the sampled states, here, similarly, the smoothed perturbed return function is derived by taking the expectation over sampled σ-randomized trajectories. The difference is that the output range of the Q-network is $[V_{\min}, V_{\max}]$, while the output range of the perturbed return function is $[J_{\min}, J_{\max}]$. Thus, the smoothed perturbed return function $\widetilde{F}$ is $\frac{(J_{\max}-J_{\min})}{\sigma}\sqrt{2/\pi}$-Lipschitz continuous. QED.

F.1.4 Proof of Theorem 8.2

We recall the definition of $\widetilde{F}_{\pi}$ as well as Theorem 8.2:

$$\widetilde{F}_{\pi}\left(\oplus_{t=0}^{H-1}\delta_t\right) := \mathop{\mathbb{E}}_{\zeta\sim\mathcal{N}(0,\sigma^2 I_{H\times N})} F_{\pi}\left(\oplus_{t=0}^{H-1}(\delta_t + \zeta_t)\right). \tag{F.12}$$

Theorem 8.2 (Expectation Bound)**.** Let

$$\underline{J_E} = \widetilde{F}_{\pi}\left(\oplus_{t=0}^{H-1}\mathbf{0}\right) - L\epsilon\sqrt{H}, \quad \text{where} \quad L = \frac{(J_{\max} - J_{\min})}{\sigma}\sqrt{\frac{2}{\pi}}. \tag{8.8}$$

Then $\underline{J_E} \leqslant \mathbb{E}\left[J_{\epsilon}(\pi')\right]$.

Proof. We note the following equality

$$\widetilde{F}_\pi(\oplus_{t=0}^{H-1}\delta_t) \overset{(a)}{=} \mathbb{E}\left[F_\pi\left(\oplus_{t=0}^{H-1}(\delta_t+\zeta_t)\right)\right] \overset{(b)}{=} \mathbb{E}\left[F_{\pi'}\left(\oplus_{t=0}^{H-1}\delta_t\right)\right] \overset{(c)}{=} \mathbb{E}\left[J_\epsilon^H(\pi')\right], \tag{F.13}$$

where (a) comes from the definition of the smoothed perturbed return function $\widetilde{F}_\pi$, (b) is due to the definition of the σ-randomized policy π', and (c) arises from the definition of the perturbed cumulative reward J_ϵ^H. Thus, the expected perturbed cumulative reward $\mathbb{E}\left[J_\epsilon^H(\pi')\right]$ is equivalent to the smoothed perturbed return function $\widetilde{F}_\pi\left(\oplus_{t=0}^{H-1}\delta_t\right)$. Furthermore, since the distance between the all-zero $\oplus_{t=0}^{H-1}\mathbf{0}$ and the adversarial perturbations $\oplus_{t=0}^{H-1}\delta_t$ is bounded by $\epsilon\sqrt{H}$, leveraging the Lipschitz smoothness of $\widetilde{F}$ in Lemma 8.2, we obtain the lower bound of the expected perturbed cumulative reward $\mathbb{E}\left[J_\epsilon^H(\pi')\right]$ as $\widetilde{F}_\pi\left(\oplus_{t=0}^{H-1}\mathbf{0}\right) - L\epsilon\sqrt{H}$. QED.

F.1.5 Proof of Theorem 8.3

We recall the definition of $\widetilde{F}_\pi^p$ as well as Theorem 8.3:

$$\widetilde{F}_\pi^p\left(\oplus_{t=0}^{H-1}\delta_t\right) := \sup_y\left\{y\in\mathbb{R} \mid \Pr\left[F_\pi\left(\oplus_{t=0}^{H-1}(\delta_t+\zeta_t)\right)\leqslant y\right]\leqslant p\right\}. \tag{F.14}$$

Theorem 8.3 (Percentile Bound)**.** Let $\underline{J_p} = \widetilde{F}_\pi^{p'}\left(\oplus_{t=0}^{H-1}\mathbf{0}\right)$, where $p' := \Phi\left(\Phi^{-1}(p) - \frac{\epsilon\sqrt{H}}{\sigma}\right)$. Then $\underline{J_p} \leqslant$ the p-th percentile of $J_\epsilon(\pi')$.

Proof. To prove Theorem 8.3, we leverage the technique in the proof for Lemma 2 of Chiang et al. [70] in their Appendix B.

For brevity, we abbreviate $\delta := \oplus_{t=0}^{H-1}\delta_t$ and redefine the plus operator such that $\delta+\zeta := \oplus_{t=0}^{H-1}(\delta_t+\zeta_t)$. Then, similar to Lemma F.1, we have the conclusion that

$$\delta \mapsto \sigma\cdot\Phi^{-1}\left(\Pr\left[F_\pi(\delta+\zeta)\leqslant\widetilde{F}_\pi^{p'}(\mathbf{0})\right]\right) \tag{F.15}$$

is 1-Lipschitz continuous, where $\zeta\sim\mathcal{N}(0,\sigma^2 I_{H\times N})$.

Thus, under the perturbations $\delta_t\in B_\epsilon$ for $t=0\dots H-1$, we have

$$\Phi^{-1}\left(\Pr\left[F_\pi(\delta+\zeta)\leqslant\widetilde{F}_\pi^{p'}(\mathbf{0})\right]\right) \leqslant \Phi^{-1}\left(\Pr\left[F_\pi(\zeta)\leqslant\widetilde{F}_\pi^{p'}(\mathbf{0})\right]\right) + \frac{\|\delta\|_2}{\sigma} \tag{F.16}$$

$$\leqslant \Phi^{-1}\left(\Pr\left[F_\pi(\zeta) \leqslant \widetilde{F}_\pi^{p'}(\mathbf{0})\right]\right) + \frac{\epsilon\sqrt{H}}{\sigma} \quad (\text{Since } \|\delta\|_2 \leqslant \epsilon\sqrt{H}) \tag{F.17}$$

$$= \Phi^{-1}(p') + \frac{\epsilon\sqrt{H}}{\sigma} \quad (\text{By definition of } \widetilde{F}_\pi^{p}) \tag{F.18}$$

$$= \Phi^{-1}(p) \quad (\text{By definition of } p'). \tag{F.19}$$

Since Φ^{-1} monotonically increase, this implies that $\Pr\left[F_\pi(\delta+\zeta) \leqslant \widetilde{F}_\pi^{p'}(\mathbf{0})\right] \leqslant p$. According to the definition of $\widetilde{F}_\pi^p$ in (F.14), we see that $\widetilde{F}_\pi^{p'}(\mathbf{0}) \leqslant \widetilde{F}_\pi^{p}(\delta)$, i.e., $\underline{J_p} \leqslant$ the p-th percentile of $J_\epsilon(\pi')$. Hence, the theorem is proved. QED.

F.1.6 Proof of Theorem 8.4

We recall Theorem 8.4:

Theorem 8.4. Let $(r_t^1, \ldots, r_t^{|\mathcal{A}|-1})$ be a sequence of certified radii for state s_t at time step t, where r_t^k denotes the radius such that if $\epsilon < r_t^k$, the possible action at time step t will belong to the actions corresponding to top k action values of $\widetilde{Q}$ at state s_t. The definition of r_t in Theorem 8.1 is equivalent to r_t^1 here. The radii can be computed similarly as follows:

$$r_t^k = \frac{\sigma}{2}\left(\Phi^{-1}\left(\frac{\widetilde{Q}^\pi(s_t, a_1) - V_{\min}}{V_{\max} - V_{\min}}\right) - \Phi^{-1}\left(\frac{\widetilde{Q}^\pi(s_t, a_{k+1}) - V_{\min}}{V_{\max} - V_{\min}}\right)\right), \quad 1 \leqslant k < |\mathcal{A}|, \tag{8.10}$$

where a_1 is the action of the highest $\widetilde{Q}$ value at state s_t and a_{k+1} is the $(k+1)$-th best action. We additionally define $r_t^0(s_t) = 0$, which is also compatible with the definition above.

Proof. Replacing a_2 with a_{k+1} in the proof for Theorem 8.1 in Appendix F.1.2 directly leads to Theorem 8.4. QED.

F.2 DETAILS OF CERTIFICATION STRATEGIES

In this section, we cover the concrete details regarding the implementation of our three certification strategies, as a complement to the high-level ideas introduced back in Section 8.3 and Section 8.4.

F.2.1 Detailed Algorithm of CROP-LoAct

Algorithm F.1: CROP-LoAct: Local smoothing for certifying per-state action

Input: state s, trained value network Q^π with range $[V_{\min}, V_{\max}]$; parameters for smoothing: sampling times m, smoothing variance σ^2, one-sided confidence parameter α

Output: smoothed value network $\widetilde{Q}^\pi$, selected action a, certification indicator *cert*, certified radius r constant L

`// Step 1: smoothing`

Generate noise samples $\delta_i \sim \mathcal{N}(\prime, \sigma^{\in}\mathcal{I})$ for $1 \leqslant i \leqslant m$

for *each action* $a \in \mathcal{A}$ **do**

  `// clipping and averaging`

  $\widetilde{Q}^\pi(s,a) \leftarrow \frac{1}{m}\sum_{i=1}^{m} \texttt{clip}(Q^\pi(s+\delta_i, a), \min = V_{\min}, \max = V_{\max})$

end

$a_1, a_2 \leftarrow$ best action and runner-up action given by $\widetilde{Q}^\pi$

`// Step 2: certification`

$\Delta = (V_{\max} - V_{\min})\sqrt{\frac{1}{2m}\ln\frac{1}{\alpha}}$

`// confidence interval`

if $\widetilde{Q}^\pi(s,a_1) \geqslant \widetilde{Q}^\pi(s,a_2) + 2\Delta$ **then**

  `// certification success`

  $cert \leftarrow$ `True`

  $r \leftarrow \frac{\sigma}{2}(\Phi^{-1}(\frac{\widetilde{Q}^\pi(s,a_1)-\Delta-V_{\min}}{V_{\max}-V_{\min}}) - \Phi^{-1}(\frac{\widetilde{Q}^\pi(s,a_2)+\Delta-V_{\min}}{V_{\max}-V_{\min}}))$

else

  `// certification failure`

  $cert \leftarrow$ `False`

  $r \leftarrow$ `undefined`

end

return $\widetilde{Q}^\pi$, a_1, *cert*, r

We present the concrete algorithm of CROP-LoAct in Algorithm F.1 for the procedures introduced in Section 8.3.2. For each given state s_t, we first perform Monte Carlo sampling [77, 204] to achieve local smoothing. Based on the smoothed value function $\widetilde{Q}^\pi$, we then compute the robustness certification for per-state action, i.e., the certified radius r_t at the given state s_t, following Theorem 8.1.

Detailed Inference Procedure. During inference, we invoke the model with m samples of Gaussian noise at each time step, and use the averaged Q value on these m noisy samples to obtain the greedy action selection and compute the certification. This procedure is similar to Predict in Cohen et al. [77], but since RL involves multiple step decisions, we do not take the "abstain" decision as in Predict in Cohen et al. [77]; instead, CROP-LoAct will take the greedy action at all steps no matter whether the action can be certified or not.

Estimation of the Algorithm Parameters $V_{\min}, V_{\max}$. Our estimate is obtained via sampling the trajectories and calculating the Q values associated with the state-action pairs along these trajectories. Since we perform clipping using the obtained bounds, the certification is sound. The cost for estimating $V_{\min}$ and $V_{\max}$ is essentially associated with the number of sampled trajectories, the number of steps in each trajectory, the number of sam-

Algorithm F.2: CROP-GRe: Global smoothing for certifying cumulative reward

Input: initial state distribution d_0, trained value network Q^π, game cumulative reward range $[J_{\min}, J_{\max}]$, number of steps in an episode H, perturbation magnitude at each state ϵ, percentile p; parameters for smoothing: sampling times m, smoothing variance σ^2, one-sided confidence parameter α

Output: Expectation bound $\underline{J_E}$, p-th percentile bound $\underline{J_p}$

// Step 1: smoothing
for $i = 1$ *to* m **do**
  $s_0 \sim d_0, J_i^C \leftarrow 0$ // initialization
  Generate macro-state noise $\delta_t \sim \mathcal{N}(\prime, \sigma^{\in}\mathcal{I})$ for $0 \leqslant t < H$
  for $t = 0$ *to* $H - 1$ **do**
    $a_t \leftarrow \arg\max_a Q(s_t + \delta_t, a)$
    Execute action a_t and observe reward re_t and next state s_{t+1} // take a step
    $J_i^C \leftarrow J_i^C + re_t$ // accumulate the reward
  end
end
// Step 2.1: certifying expectation bound
$\widetilde{F} \leftarrow \frac{1}{m}\sum_{i=1}^m J_i^C$ // smoothed actual reward
$\Delta_{\text{conf}} \leftarrow (R_{\max} - R_{\min})\sqrt{\frac{\ln(1/\alpha)}{2m}}$ // confidence interval
$\Delta_{\text{lip}} \leftarrow \frac{R_{\max} - R_{\min}}{\sigma}\sqrt{\frac{2}{\pi}} \cdot \epsilon\sqrt{H}$ // bound given by Lipschitz continuity
$\underline{J_E} \leftarrow \widetilde{F} - \Delta_{\text{conf}} - \Delta_{\text{lip}}$

// Step 2.2: certifying percentile bound
$k \leftarrow \texttt{ComputeOrderStats}(\epsilon, \sigma, p, m, H)$
$\underline{J_p} \leftarrow k$-th smallest value in $\{J_i^C\}_{i=1}^m$

return $\underline{J_E}, \underline{J_p}$

pled Gaussian noise m per step, and the cost of doing one forward pass of the Q network; thus, the estimation can be done with low cost. Furthermore, we can balance the trade-off between the accuracy and efficiency of the estimation, and for any configuration of $V_{\min}$ and $V_{\max}$, the certification will invariably be sound (reason above).

F.2.2 Detailed Algorithm of CROP-GRe

We present the concrete algorithm of CROP-GRe in Algorithm F.2 for the procedures introduced in Section 8.4.1. Similarly to CROP-LoAct, the algorithm also consists of two parts: performing smoothing and computing certification, where we compute both the expectation bound $\underline{J_E}$ and the percentile bound $\underline{J_p}$.

Step 1: Global Smoothing. We adopt Monte Carlo sampling [77, 204] to estimate the smoothed perturbed return function $\widetilde{F}$ by sampling multiple σ-randomized trajectories via drawing m noise sequences. For each noise sequence $\zeta \sim \mathcal{N}(0, \sigma^2 I_{H\times N})$, we apply noise ζ_t to the input state s_t sequentially, and obtain the sum of the reward $J_i^C = \sum_{t=0}^{H-1} re_t$ as the return for this σ-randomized trajectory. We then aggregate the smoothed perturbed return values $\{J_i^C\}_{i=1}^m$ via mean smoothing and percentile smoothing.

Step 2: Certification for Perturbed Cumulative Reward. First, we compute the *expectation bound* $\underline{J_E}$ using Theorem 8.2. Since the smoothed perturbed return function $\widetilde{F}$ is obtained based on m sampled noise sequences, we use Hoeffding's inequality [150] to compute the lower bound of the random variable $\widetilde{F}\left(\oplus_{t=0}^{H-1}\mathbf{0}\right)$ with a confidence level α. We then calculate the lower bound of $\widetilde{F}\left(\oplus_{t=0}^{H-1}\delta_t\right)$ under all possible ℓ_2-bounded perturbations $\delta_t \in B_\epsilon$ leveraging the smoothness of $\widetilde{F}$.

We then compute the *percentile bound* $\underline{J_p}$ using Theorem 8.3. We let J_i^C be sorted increasingly, and perform normal approximations [352] to compute the largest empirical order statistic J_k^C such that $\Pr\left[\underline{J_p} \geqslant J_k^C\right] \geqslant 1-\alpha$. The empirical order statistic J_k^C is then used as the proxy of $\underline{J_p}$ under α confidence level. We next provide detailed explanations for `ComputeOrderStats`, which aims to compute the order k using binomial formula plus normal approximation.

Estimation of $\widetilde{F}_\pi$ (in Lemma 8.2). For computing the expectation bound (i.e., the lower bound of $\mathbb{E}_\Delta\left[J_\epsilon(\pi')\right]$), an intermediate step is to estimate $\widetilde{F}_\pi$ (as shown in line 8 of Algorithm F.2).

We emphasize that the *accuracy* of the estimation does not influence the *soundness* of the lower bound calculation. The reason is given below. The number of sampled randomized trajectories m controls the trade-off between the estimation *efficiency* and *accuracy*, as well as the *tightness* of the derived lower bound. Concretely, a small m would provide high efficiency, low estimation accuracy, and loose lower bound; but the lower bound is always *sound*, since our algorithm explicitly accounts for the *inaccuracy* associated with m via leveraging the Hoeffding's inequality in line 9 in Algorithm F.2.

Details of `ComputeOrderStats`. We consider the sorted sequence $J_1^C \leqslant J_2^C \leqslant \cdots \leqslant J_m^C$. We additionally set $J_0^C = -\infty$ and $J_{m+1}^C = \infty$. Our goal is to find the largest k such that $\Pr\left[\underline{J_p} \geqslant J_k^C\right] \geqslant 1-\alpha$. We evaluate the probability explicitly as follows:

$$\Pr\left[\underline{J_p} \geqslant J_k^C\right] = \sum_{i=k}^{m} \Pr\left[J_i^C \leqslant \underline{J_p} < J_{i+1}^C\right] = \sum_{i=k}^{m} \binom{m}{i} (p')^i (1-p')^{m-i}. \tag{F.20}$$

Thus, the condition $\Pr\left[\underline{J_p} \geqslant J_k^C\right] \geqslant 1-\alpha$ is equivalent to

$$\sum_{i=0}^{k-1} \binom{m}{i} (p')^i (1-p')^{m-i} \leqslant \alpha. \tag{F.21}$$

Given large enough m, the LHS of (F.21) can be approximated via a normal distribution with mean equal to mp' and variance equal to $mp'(1-p')$. Concretely, we perform binary search to find the largest k that satisfies the constraint.

We finally explain the upper bound of ϵ that can be certified for each given smoothing variance. In practical implementation, for a given sampling number m and confidence level parameter α, if p' is too small, then the condition (F.21) may not be satisfied even for $k = 1$. This implies that the existence of an upper bound of ϵ that can be certified for each smoothing parameter σ, recalling that $p' := \Phi\left(\Phi^{-1}(p) - \epsilon\sqrt{H}/\sigma\right)$.

Detailed Inference and Certification Procedures. We next provide detailed descriptions of the inference and certification procedures, respectively.

Inference procedure. We deploy the σ-randomized policy π' as defined in Definition 8.4 in Section 8.4.1. That is, for each observed state, we sample one time of Gaussian noise Δ_t to add to s_t, and take action according to this single randomized observation $a_t = \pi(s_t + \Delta_t)$. Thus, in deployment time, the Δ-randomized policy π' executes on *one* rollout—at each step it takes one observation and chooses one action (following the procedure described above).

Certification procedure. We sample m randomized trajectories in CROP-GRe instead of sampling m noisy states per time step as in CROP-LoAct. For each sampled randomized trajectory, at each time step in the trajectory, we invoke the model with 1 sample of Gaussian noise (as shown in (8.6)). Given the m randomized trajectories and the cumulative reward for each of them, we compute the certification using Theorem 8.2 and Theorem 8.3.

Algorithm Parameters $J_{\min}, J_{\max}$. We use the default values of $J_{\min}, J_{\max}$ in the game specifications rather than estimate them, which are independent with the trained models. Thus there is no computational cost at all for obtaining the two parameters. As mentioned under Theorem 8.2 in Section 8.4.1, using these default values may induce a loose bound in practice, so we further propose the percentile smoothing that aims to eliminate the dependency of our certification on $J_{\min}$ and $J_{\max}$, thus achieving a tighter bound.

F.2.3 Detailed Algorithms of CROP-LoRe

In the following, we will explain the *trajectory exploration and expansion*, the *growth of perturbation magnitude*, and *optimization tricks* in details.

Trajectory Exploration and Expansion. CROP-LoRe organizes all possible trajectories in the form of a search tree and progressively grows it. Each node of the tree represents

a state, and the depth of the node is equal to the time step of the corresponding state in the trajectory. The root node (at depth 0) represents the initial state s_0. For each node, leveraging Theorem 8.4, we compute a non-decreasing sequence $\{r^k(s)\}_{k=0}^{|\mathcal{A}|-1}$ corresponding to required perturbation radii for π to choose each alternative action (the subscript t is omitted for brevity). Suppose the current ϵ satisfies $r^i(s) \leqslant \epsilon < r^{i+1}(s)$. We grow $(i+1)$ branches from current state s corresponding to the original action and i alternative actions since $\epsilon \geqslant r^j(s)$ for $1 \leqslant j \leqslant i$. For nodes on the newly expanded branch, we repeat the same procedure to expand the tree with depth-first search [363] until the terminal state of the game is reached or the node depth reaches H. As we expand, we keep the record of cumulative reward for each trajectory and update the lower bound $\underline{J}$ when reaching the end of the trajectory if necessary.

Perturbation Magnitude Growth. When all trajectories for perturbation magnitude ϵ are explored, we need to increase ϵ to seek for certification under larger perturbations. Luckily, since the action space is discrete, we do not need to examine every $\epsilon \in \mathbb{R}^+$ (which is infeasible) but only need to examine the next ϵ where the chosen action in some step may change. We leverage *priority queue* [377] to effectively find out such next "critical" ϵ. Concretely, along the trajectory exploration, at each tree node, we search for the possible actions and store actions corresponding to $\{r^k(s)\}_{k=i+1}^{|\mathcal{A}|-1}$ into the priority queue, since these actions are exactly those need to be explored when ϵ grows. After all trajectories for ϵ are fully explored, we pop out the head element from the priority queue as the next node to expand and the next perturbation magnitude ϵ to grow. We repeat this process until the priority queue becomes empty or the perturbation magnitude ϵ reaches the predefined threshold.

Additional Optimization. We adopt some additional optimization tricks to reduce the complexity of the algorithm. First, for environments with no negative reward, we perform pruning to limit the tree size—if the cumulative reward leading to the current node already reaches the recorded lower bound, we can perform pruning, since the tree that follows will not serve to update the lower bound. This largely reduces the potential search space. We additionally adopt the memorization [261] technique which is commonly applied in search algorithms.

We point out a few more potential improvements. *First*, with more specific knowledge of the game mechanisms, the search algorithm can be further optimized. Take the Pong game as an example, given the horizontal speed of the ball, we can compress the time steps where the ball is flying between the two paddles, thus reducing the computation. *Second*,

empirical attacks may be efficiently incorporated into the algorithm framework to provide upper bounds that help with pruning.

Time Complexity. The time complexity of CROP-LoRe is $O(H|S_{\text{explored}}|\times(\log|S_{\text{explored}}|+|A|T))$, where $|S_{\text{explored}}|$ is the number of explored states throughout the search procedure, which is no larger than cardinality of state set, H is the horizon length, $|A|$ is the cardinality of action set, and T is the time complexity of performing local smoothing. The main bottleneck of the algorithm is the large number of possible states, which is in the worst case exponential to state dimension. However, to provide a sound worst-case certification agnostic to game properties, exploring all possible states may be inevitable.

Detailed Inference and Certification Procedures and Important Modules in Algorithm F.3. We next provide detailed descriptions of the inference and certification procedures, respectively, along with other important modules.

Inference procedure. In CROP-LoRe, we deploy the locally smoothed policy $\tilde{\pi}$ as defined in (8.3) in Section 8.3.1. Concretely, for each observed state s_t, we sample a batch of Gaussian noise to add to s_t, and take action according to the computed mean Q value on the batch of randomized observations. (Actually, the *inference* procedure per step is exactly the same as in CROP-LoAct described in Appendix F.2.1.)

Certification procedure. We obtain the *certification* via Theorem 8.4 and the adaptive search algorithm, which outputs a collection of pairs $\{(\epsilon_i, \underline{J}_{\epsilon_i})\}_{i=1}^{|C|}$ sorted in ascending order of ϵ_i, where $|C|$ is the length of the collection. The interpretation for this collection of pairs is provided below. For all ϵ, let i be the largest integer such that $\epsilon_i \leqslant \epsilon < \epsilon_{i+1}$, then as long as the perturbation magnitude $\epsilon \leqslant \epsilon'$, the cumulative reward $J_\epsilon(\tilde{\pi}) \geqslant \underline{J}_{\epsilon_i}$. This is supported by the fact that all certified radii associated with all nodes in the entire expanded tree compose a discrete set of finite cardinality; and the perturbation value between two adjacent certified radii in the returned collection will not lead to a different tree from the tree corresponding to the largest smaller perturbation magnitude. This actually is also the inspiration for the certification design.

Important modules. The function `GetAction` computes the possible actions at a given state s under the limit ϵ, while the procedure `Expand` accomplishes the task of expanding upon a given node/state. The main part of the algorithm involves a loop that repeatedly selects the next element from the priority queue, i.e., a node associated with an ϵ value, to expand upon.

Additional Clarifications. We clarify that the algorithm only requires access to the environment such that it can obtain the reward and next state via *interacting* with the environment $\mathcal{E}$ (i.e., taking action a at state s and obtain next action s and reward r), but does not require access to an oracle transition function Γ. We further emphasize that access to the environment that supports back-tracking is already sufficient for conducting our adaptive search algorithm.

Algorithm F.3: CROP-LoRe: Adaptive search for certifying cumulative reward

Input: Environment $\mathcal{E} = (\mathcal{S}, \mathcal{A}, \mathcal{R}, -, \lceil \prime)$, trained value network Q^π with range $[V_{\min}, V_{\max}]$; parameters for randomized smoothing: sampling times m, smoothing variance σ^2, one-sided confidence parameter α

Output: a map M that maps an attack magnitude ϵ to the corresponding certified lower bound of reward J

// Initialize global variables

$p\_que \leftarrow \varnothing$ // initialize an empty priority queue containing tuples of (state s, action a, radius r, reward J), sorted by increasing r

$M \leftarrow \varnothing$

$J_{\text{global}} \leftarrow \infty$ // initialize global minimum reward

$\Delta = (V_{\max} - V_{\min}) \sqrt{\frac{1}{2m} \ln \frac{1}{\alpha}}$ // confidence bound

Function GetActions$(s, \epsilon_{\text{lim}}, J_{\text{cur}})$:

  Generate noise samples $\delta_i \sim \mathcal{N}(\prime, \sigma^{\in} \mathcal{I})$ for $1 \leqslant i \leqslant m$

  for *each action* $a \in \mathcal{A}$ **do**

    $\widetilde{Q}^\pi(s, a) \leftarrow \frac{1}{m} \sum_{i=1}^{m} \texttt{clip}(Q^\pi(s + \delta_i, a), \min = V_{\min}, \max = V_{\max})$

  end

  $a^\star \leftarrow \arg\max_{a \in \mathcal{A}} \widetilde{Q}^\pi(s, a)$

  $a\_list \leftarrow \varnothing$

  for *each action* $a \in \mathcal{A}$ **do**

    if $\Gamma(s, a) = \perp$ **then**

      continue

    end

    if $\widetilde{Q}^\pi(s, a^\star) \geqslant \widetilde{Q}^\pi(s, a) + 2\Delta$ **then**

      $r \leftarrow \frac{\sigma}{2} (\Phi^{-1}(\frac{\widetilde{Q}^\pi(s, a^\star) - \Delta - V_{\min}}{V_{\max} - V_{\min}}) - \Phi^{-1}(\frac{\widetilde{Q}^\pi(s, a) + \Delta - V_{\min}}{V_{\max} - V_{\min}}))$

    else

      $r \leftarrow 0$

    end

    if $r \leqslant \epsilon_{\text{lim}}$ **then**

      // take possible actions

      $a\_list \leftarrow a\_list \cup \{a\}$

    else

      // store impossible actions in queue for later expansion

      p_que.push$((s, a, r, J_{\text{cur}}))$

    end

  end

  return a_list

Procedure Expand$(s, \epsilon_{\text{lim}}, J_{\text{cur}})$:

  if $J_{\text{cur}} \geqslant J_{\text{global}}$ **then**

    return 0 // pruning

  end

  $a\_list \leftarrow$ GetActions$(s, \epsilon_{\text{lim}}, J_{\text{cur}})$

  if $a\_list = \varnothing$ **then**

    $J_{\text{global}} \leftarrow \texttt{min}(J_{\text{global}}, J_{\text{cur}})$

    return 0

  end

  for $a \in a\_list$ **do**

    $s' \leftarrow \Gamma(s, a)$

    $ret \leftarrow$ Expand$(s', \epsilon_{\text{lim}}, J_{\text{cur}} + R(s, a))$

  end

$s_0 \sim d_0$ // initialize initial state

Expand $(s_0, \epsilon_{\text{lim}} = 0, J_{\text{cur}} = 0)$ // expand initial trajectory

while *True* **do**

  if $p\_que = \varnothing$ **then**

    break

  end

  // pop out the first element

  $(s, a, r, J) \leftarrow p\_que$.pop()

  // examine the next first element

  $(\_, \_, r', \_) \leftarrow p\_que$.top()

  // derive the critical ϵ's

  $\epsilon \leftarrow r, \epsilon' \leftarrow r'$

  // obtain a pair of mapping

  $M[\epsilon] \leftarrow J_{\text{global}}$

  // expand the tree from the new node

  Expand $(\Gamma(s, a), \epsilon', J + R(s, a))$

end

APPENDIX G: APPENDIX FOR CHAPTER 9

G.1 ALGORITHM PSEUDOCODE AND DISCUSSION

G.1.1 COPA Training Protocol

Algorithm G.1: COPA training protocol.

Input: training dataset D, number of partitions k, deterministic hash function h

Output: COPA subpolicies $\{\pi_i\}_{i=0}^{k-1}$

for $i \in [k]$ **do**
  $D_i \leftarrow \{\tau \in D \mid h(\tau) \equiv i \pmod{k}\}$ // Separate the training data D into k partitions
end
for *each partition* D_i **do**
  $\pi_i \leftarrow \mathcal{M}_0(D_i)$ // Subpolicy trained on partition D_i with offline RL algorithm $\mathcal{M}_0$
end
return $\{\pi_i\}_{i=0}^{k-1}$

G.1.2 COPA Per-state Action Certification

Per-State Partition Aggregation (PARL).

Algorithm G.2: COPA per-state certification algorithm for Per-State Partition Aggregation (PARL).

Input: environment $\mathcal{E} = (\mathcal{S}, \mathcal{A}, R, P, H, d_0)$, subpolicies $\{\pi_i\}_{i=0}^{k-1}$

Output: COPA robust size at each time step $\{\bar{K}_t\}_{t=0}^{H-1}$

$s_0 \sim d_0$ // sample initial state
for t *from* 0 *to* $H-1$ **do**
  for *each* $a \in \mathcal{A}$ **do**
    Compute $n_a(s_t)$ from subpolicies' $\{\pi_i\}_{i=0}^{k-1}$ decisions // n_a is defined in Section 9.2
  end
  Determine the chosen action $a_t \leftarrow \pi_{\mathrm{P}}(s_t)$ according to PARL (Definition 9.4)
  Compute $\bar{K}_t$ according to Equation (9.6) in Theorem 9.1
  $s_{t+1} \sim P(s_t, a_t)$
end
return $\{\bar{K}_t\}_{t=0}^{H-1}$

Temporal Partition Aggregation (TPARL).

Algorithm G.3: COPA per-state certification algorithm for Temporal Partition Aggregation (TPARL).

Input: environment $\mathcal{E} = (\mathcal{S}, \mathcal{A}, R, P, H, d_0)$, subpolicies $\{\pi_i\}_{i=0}^{k-1}$, window size W

Output: COPA robust size at each time step $\{\bar{K}_t\}_{t=0}^{H-1}$

$s_0 \sim d_0$ // sample initial state

for t *from* 0 *to* $H-1$ **do**

  for *each* $a \in \mathcal{A}$ **do**

    Compute $n_a(s_t)$ from subpolicies' $\{\pi_i\}_{i=0}^{k-1}$ decisions // n_a is defined in Section 9.2

  end

  Determine the chosen action $a_t \leftarrow \pi_{\mathrm{T}}(s_t)$ according to TPARL (Definition 9.5)

  $p_{\min} \leftarrow \infty$

  for *each* $a' \in \mathcal{A}, a' \neq a_t$ **do**

    for i *from* 0 *to* $k-1$ **do**

      Compute $h_{i,a_t,a'}$ according to Equation (9.8)

    end

    $\{h^{(i)}_{a_t,a'}\}_{i=1}^{k} \leftarrow \mathsf{sorted}(\{h_{i,a_t,a'}\}_{i=0}^{k-1}, \mathtt{reverse} = \mathtt{True})$

    Compute $\delta_{a_t,a'}$

    $\mathtt{sum} \leftarrow 0, p \leftarrow 0$

    for j *from* 1 *to* k **do**

      if $\mathtt{sum} + h^{(j)}_{a_t,a'} > \delta_{a_t,a'}$ **then**

        $p \leftarrow j-1$

        break

      end

      $p \leftarrow j, \mathtt{sum} \leftarrow \mathtt{sum} + h^{(j)}_{a_t,a'}$

    end

    $p_{\min} \leftarrow \min\{p_{\min}, p\}$

  end

  $\bar{K}_t \leftarrow p_{\min}$

  $s_{t+1} \sim P(s_t, a_t)$

end

return $\{\bar{K}_t\}_{t=0}^{H-1}$

Dynamic Temporal Partition Aggregation (DPARL).

Algorithm G.4: COPA per-state certification algorithm for Dynamic Temporal Partition Aggregation (DPARL).

Input: environment $\mathcal{E} = (\mathcal{S}, \mathcal{A}, R, P, H, d_0)$, subpolicies $\{\pi_i\}_{i=0}^{k-1}$, maximum window size $W_{\max}$
Output: COPA robust size at each time step $\{\bar{K}_t\}_{t=0}^{H-1}$

$s_0 \sim d_0$ // sample initial state
for t *from* 0 *to* $H-1$ **do**
  for *each* $a \in \mathcal{A}$ **do**
    Compute $n_a(s_t)$ from subpolicies' $\{\pi_i\}_{i=0}^{k-1}$ decisions given s_t // n_a is defined in Section 9.2
  end
  Determine the chosen action $a_t \leftarrow \pi_{\mathrm{D}}(s_t)$ and chosen window size W' according to DPARL (Definition 9.6)
  $p_{\min} \leftarrow \infty$
  $\bar{K}_t \leftarrow$ Algorithm G.3 // use Algorithm G.3 with W replaced by W' to compute $\bar{K}_t$
  $\bar{K}_t^D \leftarrow \bar{K}_t$
  for *each* $a' \in \mathcal{A}, a' \neq a_t$ **do**
    for *each* $a'' \in \mathcal{A}, a'' \neq a_t$ **do**
      for W^* *from* 1 *to* $\min\{W_{\max}, t+1\}$ **do**
        $a^{\#} \leftarrow \arg\max_{a_0 \neq a', a_0 \in \mathcal{A}} n_{a_0}(s_{t-W^*+1:t})$
        for w *from* 1 *to* $\max\{W', W^*\}$ **do**
          Compute $\max_{a_0 \in \mathcal{A}} \sigma^w(a_0)$ according to Equation (9.11)
        end
        for i *from* 0 *to* $k-1$ **do**
          for w *from* 1 *to* $\max\{W', W^*\}$ **do**
            Compute $\sigma^w(\pi_i(s_{t-w}))$ according to Equation (9.11)
          end
          Compute g_i according to Equation (9.11)
        end
        $\{g^{(i)}\}_{i=1}^{k} \leftarrow \mathsf{sorted}(\{g_i\}_{i=0}^{k-1}, \mathtt{reverse} = \mathtt{True})$
        Compute $n_{a'}^{W^*}, n_{a^{\#}}^{W^*}, n_{a_t}^{W'}, n_{a''}^{W'}$
        $\mathtt{tmp} \leftarrow W'(n_{a'}^{W^*} - n_{a^{\#}}^{W^*}) - W^*(n_{a_t}^{W'} - n_{a''}^{W'}) - \mathbb{I}[a' > a_t]$
        $\mathtt{sum} \leftarrow 0, p \leftarrow 0$
        for j *from* 1 *to* k **do**
          if $\mathtt{sum} + \mathtt{tmp} \geqslant 0$ **then**
            $p \leftarrow j - 1$
            break
          end
          $p \leftarrow j, \mathtt{sum} \leftarrow \mathtt{sum} + g^{(j)}$
        end
        $L_{a',a''}^{W^*,W'} \leftarrow p, \bar{K}_t^D \leftarrow \min\{\bar{K}_t^D, L_{a',a''}^{W^*,W'}\}$
      end
    end
  end
  $s_{t+1} \sim P(s_t, a_t)$
end
return $\{\bar{K}_t^D\}_{t=0}^{H-1}$

G.1.3 COPA Cumulative Reward Certification

COPA-LoRe alternately executes the procedure of *trajectory exploration and expansion* and *poisoning threshold growth*. In trajectory exploration and expansion, COPA-LoRe organizes all possible trajectories in the form of a search tree and progressively grows it. For each node (representing a state), we leverage Theorems 9.4, 9.5 and 9.7 to compute the Possible Action Set. We then expand the tree branches corresponding to the actions in the derived set. In poisoning threshold growth, when all trajectories for the current poisoning threshold are explored, we increase K to seek certification under larger poisoning sizes, via

maintaining a priority queue of all poisoning sizes during the expansion of the tree. The iterative procedures end when the priority queue becomes empty.

The highlighted line in the following algorithm needs to inject different algorithms based on the aggregation protocol: for PARL (π_{P}), use Theorem 9.4; for TPARL (π_{T}), use Theorem 9.5; for DPARL (π_{D}), use Theorem 9.7.

Algorithm G.5: COPA-LoRe: adaptive tree search for cumulative reward certification.

Input: environment $\mathcal{E} = (\mathcal{S}, \mathcal{A}, R, P, H, d_0)$, subpolicies $\{\pi_i\}_{i=0}^{k-1}$, aggregated policy π_{P} or π_{T} (with W) or π_{T} (with $W_{\max}$)

Output: a map M that maps poisoning size K to corresponding certified lower bound of cumulative reward $\underline{J}_K$

```
// Initialize global variables
p_que ← ∅ // initialize an empty priority queue containing tuples of (state history s_{0:t}, action a, poisoning size K, reward J) sorted by increasing K
J_global ← ∞ // initialize global minimum reward
Function GetActions(s_{0:t}, K_lim, J_cur):
    A ← possibleActions(s_{0:t}, {π_i}_{i=1}^{k-1}, π_*, K_lim) // compute the possible action set given state history, subpolicies, aggregation policy π_*, and poisoning size according to theorems in Section 9.4.2
    a_list ← ∅
    for each action a ∈ A do
        if P(s, a) = ⊥ then
            continue // game terminate here, no possible larger or lower cumulative reward to search
        end
        a_list ← a_list ∪ {a} // record possible actions to expand
    end
    if A ≠ A then
        K' ← min_{possibleActions(s_{0:t}, {π_i}_{i=0}^{k-1}, π_*, K) ≠ A} K
        A_new ← possibleActions(s_{0:t}, {π_i}_{i=1}^{k-1}, π_*, K') \ A // compute the immediate actions that will possibly be chosen if enlarging the poisoning size
        for each action a ∈ A_new do
            p_que.push((s_{0:t}, a, K', J_cur))
        end
    end
    return a_list
Procedure Expand(s_{0:t}, K_lim, J_cur):
    if J_cur ⩾ J_global ∧ (step reward is non-negative) then
        return // pruning
    end
    a_list ← GetActions(s_{0:t}, {π_i}_{i=0}^{k-1}, π_*, K_lim) // compute according to theorems in Section 9.4.2
    if a_list = ∅ then
        J_global ← min{J_global, J_cur}
        return
    end
    for a ∈ a_list do
        s_{t+1} ← P(s_t, a)
        if s_{t+1} = ⊥ then
            J_global ← min{J_global, J_cur}
        end
        else
            Expand (s_{0:t+1}, K_lim, J_cur + R(s_t, a))
        end
    end
M ← ∅
s_0 ← d_0
// initial state is s_0
Expand (s_0, K_lim = 0, J_cur = 0) // expand initial trajectory
while True do
    if p_que = ∅ then
        break // no state to expand
    end
    (s_{0:t}, a, K, J) ← p_que.pop() // pop out the first element
    (_, _, K', _) ← p_que.top() // examine the next element
    M(K) ← J_global // obtain one new point of certification result
    s_{t+1} ← P(s_t, a)
    Expand (s_{0:t+1}, K', J + R(s_t, a)) // expand the tree from the new node
end
return M // indeed the algorithm can terminate at any time within the while loop
```

Time Complexity. The complexity of COPA-LoRe is $O(H|\mathcal{S}_{\text{explored}}|(\log|\mathcal{S}_{\text{explored}}| + |\mathcal{A}|T))$, where $|\mathcal{S}_{\text{explored}}|$ is the number of explored states throughout the search procedure, which is no larger than cardinality of state set $\mathcal{S}$, H is the horizon length, $|\mathcal{A}|$ is the cardinality of action set, and T is the time complexity of per-state action certification. The main bottleneck of the algorithm is the large number of possible states, which is in the worse case exponential to state dimension. However, to provide a deterministic worst-case certification agnostic to environment properties, exploring all possible states may be inevitable.

Relation with Chapter 8. The COPA-LoRe algorithm is inspired from CROP-LoRe in Chapter 8, which also leverages tree search to explore all possible states and thus derive the robustness guarantee. However, the major distinction is that the algorithm in Chapter 8 tries to derive a probabilistic guarantee of RL robustness against state perturbations, while COPA-LoRe derives a deterministic guarantee of RL robustness against poisoning attacks. We think this general tree search methodology can be further extended to provide certification beyond evasion attack and poisoning attack and we leave it as future work.

Extension to stochastic MDPs. In the current version of COPA-LoRe, the exhaustive search is enabled by the deterministic MDP assumption. However, we foresee the potential of conveniently extending COPA-LoRe to stochastic MDPs and will discuss one concrete method below. In contrast to interacting with the environment for one time to obtain the *deterministic* next state transition in the deterministic MDP case, in the case of stochastic MDPs, we can leverage *sampling* (i.e., by repeatedly taking the same action at the same state) to obtain the set of high probability next state transitions with high confidence. In this way, our COPA-LoRe will be able to yield a *probabilistic bound*, in comparison with the *deterministic bound* achieved in this chapter enabled by the deterministic MDP assumption.

G.2 PROOFS

G.2.1 Tightness of Per-state Action Certification in PARL

Proof of Proposition 9.1. We prove by construction. Given the state s_t, we first locate the subpolicies whose chosen action is a, and denote the set of them as

$$B = \{i \in [u] \mid \pi_i(s_t) = a = \pi_{\text{P}}(s_t)\}. \tag{G.1}$$

We also denote $a' = \arg\max_{a' \neq a} n_{a'}(s_t) + \mathbb{I}[a' < a]$. According to $\bar{K}_t$'s definition (Equation (9.6)),

$$|B| = n_a(s_t) > n_a(s_t)/2 \geqslant \bar{K}_t. \tag{G.2}$$

We now pick an arbitrary subset $B' \subseteq B$ such that $|B'| = \bar{K}_t + 1$. For each $i \in B'$, we locate its corresponding parititioned dataset D_i for training subpolicy π_i. We insert one point p_i to D_i, such that our chosen learning algorithm $\mathcal{M}_0$ can train a subpolicy $\tilde{\pi}_i = \mathcal{M}_0(D_i \cup \{p_i\})$ that satisfies $\tilde{\pi}_i(s_t) = a'$. For example, the point could be $p_i = \{(s_t, a', s', \infty)\}$ and $\mathcal{M}_0$ learns the action with maximum reward for the memorized nearest state, where s' can be adjusted to make sure the point is hashed to parition i.[22] Then, we construct the poisoned dataset

$$\widetilde{D} = \left(\bigcup_{i \in B'} D_i \cup \{p_i\} \right) \cup \left(\bigcup_{i \in [u] \setminus B'} D_i \right). \tag{G.3}$$

Therefore, we have $\widetilde{D} \ominus D = \cup_{i \in B'} p_i = |B'| = \bar{K}_t + 1$.

On this poisoned dataset, we train subpolicies $\{\tilde{\pi}_i\}_{i=0}^{u-1}$ and get the aggregated policy $\widetilde{\pi_{\mathrm{P}}}$. To study $\widetilde{\pi_{\mathrm{P}}}(s_t)$, we compute the aggregated action count $\widetilde{n}_a$ on these poisoned subpolicies. We found that $\widetilde{n}_a(s_t) = n_a(s_t) - |B'|$ and $\widetilde{n}_{a'}(s_t) = n_{a'}(s_t) + |B'|$. Therefore,

$$\widetilde{n}_a(s_t) - \widetilde{n}_{a'}(s_t) = n_a(s_t) - n_{a'}(s_t) - 2(\bar{K}_t + 1) \tag{G.4}$$

$$= n_a(s_t) - n_{a'}(s_t) - 2\left(\left\lfloor \frac{n_a(s_t) - n_{a'}(s_t) - \mathbb{I}[a' < a]}{2} \right\rfloor + 1 \right) \tag{G.5}$$

$$< n_a(s_t) - n_{a'}(s_t) - (n_a(s_t) - n_{a'}(s_t) - \mathbb{I}[a' < a]) = \mathbb{I}[a' < a]. \tag{G.6}$$

Therefore, if $a' < a$, $\widetilde{n}_a(s_t) \leqslant \widetilde{n}_{a'}(s_t)$, and a' has higher priority to be chosen than a; if $a' > a$, $\widetilde{n}_a(s_t) \leqslant \widetilde{n}_{a'}(s_t) - 1$, and a' still has higher priority to be chosen than a. Hence, $\widetilde{\pi_{\mathrm{P}}}(s_t) \neq a = \pi_{\mathrm{P}}(s_t)$. To this point, we conclude the proof with a feasible construction. QED.

G.2.2 Per-state Action Certification in TPARL

Proof of Theorem 9.2. For ease of notation, we let $w = \min\{W, t+1\}$ so w is the actual window size used at step t. We let $t_0 = t - w + 1$, i.e., t_0 is the actual start time step for TPARL aggregation at step t. Now we can write the chosen action at step t without

[22] Strictly speaking, we need to choose a determinisitc hash function h for dataset partitioning such that adjustment on s' and reward (currently ∞, but can be an arbitrary large enough number) can make the point being partitioned to i, i.e., $h(p_i) \equiv i \pmod{u}$. Since our adjustment space is infinite due to infinite number of large enough reward, such assumption can be easily achieved. Same applies to other attack constructions in the following proofs.

poisoning as $a = \pi_{\mathrm{T}}(s_{t_0:t})$.

We prove the theorem by contradiction: We assume that there is a poisoning attack whose poisoning size $K \leqslant \bar{K}_t$ where $\bar{K}_t$ is defined by Equation (9.7) in the theorem, and after poisoning, the chosen action is $a^T \neq a$. We denote $\{\tilde{\pi}_i\}_{i=0}^{u-1}$ to the poisoned subpolicies and $\widetilde{\pi_{\mathrm{T}}}$ to the poisoned TPARL aggregated policy. From the definition of $\bar{K}_t$, we have

$$\sum_{i=1}^{K} h_{a,a^T}^{(i)} \leqslant \sum_{i=1}^{\bar{K}_t} h_{a,a^T}^{(i)} \leqslant \delta_{a,a^T}, \tag{G.7}$$

since $K \leqslant \bar{K}_t$, and each $h_{a,a^T}^{(i)}$ is an element of h_{i',a,a^T} for some i' where

$$h_{i',a,a^T} = \sum_{j=0}^{w-1} \mathbb{I}_{i',a}(s_{t-j}) + w - \sum_{j=0}^{w-1} \mathbb{I}_{i',a^T}(s_{t-j}) \geqslant 0 \tag{G.8}$$

by Equation (9.8) in the theorem. Since the poisoning attack within threshold K can at most affect K subpolicies, we let B be the set of affected policies and assume $|B| = K$ without loss of generality. Formally, $B = \{i \in [u] \mid \exists t' \in [t_0, t], \tilde{\pi}_i(s_{t'}) \neq \pi_i(s_{t'})\}$. Therefore, according to the monotonicity of $h_{a,a^T}^{(i)}$, from Equation (G.7),

$$\sum_{i \in B} h_{i,a,a^T} \leqslant \sum_{i=1}^{K} h_{a,a^T}^{(i)} \leqslant \delta_{a,a^T}. \tag{G.9}$$

According to the assumption of successfully poisoning attack, after attack, the sum of aggregated vote for a^T plus $\mathbb{I}[a^T < a]$ should be larger then that of a. Formally, after the poisoning,

$$\widetilde{n}_{a^T}(s_{t_0:t}) + \mathbb{I}[a^T < a] > \widetilde{n}_a(s_{t_0:t}) \tag{G.10}$$

or equivalently

$$\widetilde{n}_a(s_{t_0:t}) - (\widetilde{n}_{a^T}(s_{t_0:t}) + \mathbb{I}[a^T < a]) < 0. \tag{G.11}$$

From the statement of Theorem 9.2,

$$\delta_{a,a^T} = n_a(s_{t_0:t}) - (n_{a^T}(s_{t_0:t}) + \mathbb{I}[a^T < a]). \tag{G.12}$$

Take the difference of the above two equations, we know that the attack satisfies

$$n_a(s_{t_0:t}) - \widetilde{n}_a(s_{t_0:t}) - (n_{a^T}(s_{t_0:t}) - \widetilde{n}_{a^T}(s_{t_0:t})) > \delta_{a,a^T}. \tag{G.13}$$

Since the attack only changes the subpolicies in B, we have

$$
\begin{aligned}
n_a(s_{t_0:t}) - \widetilde{n}_a(s_{t_0:t}) &= \sum_{i \in B} \left(\sum_{j=0}^{w-1} \mathbb{I}[\pi_i(s_{t-j}) = a] - \sum_{j=0}^{w-1} \mathbb{I}[\tilde{\pi}_i(s_{t-j}) = a] \right) \\
&\leqslant \sum_{i \in B} \sum_{j=0}^{w-1} \mathbb{I}_{i,a}(s_{t-j}), \qquad \text{(G.14)} \\
n_{a^T}(s_{t_0:t}) - \widetilde{n}_{a^T}(s_{t_0:t}) &= \sum_{i \in B} \left(\sum_{j=0}^{w-1} \mathbb{I}[\pi_i(s_{t-j}) = a^T] - \sum_{j=0}^{w-1} \mathbb{I}[\tilde{\pi}_i(s_{t-j}) = a^T] \right) \\
&\geqslant \sum_{i \in B} \left(\sum_{j=0}^{w-1} \mathbb{I}_{i,a^T}(s_{t_j}) - w \right). \qquad \text{(G.15)}
\end{aligned}
$$

Inject them into Equation (G.13) yields

$$
\sum_{i \in B} \underbrace{\left(\sum_{j=0}^{w-1} \mathbb{I}_{i,a}(s_{t-j}) + w - \sum_{j=0}^{w-1} \mathbb{I}_{i,a^T}(s_{t-j}) \right)}_{h_{i,a,a^T}} > \delta_{a,a^T} \qquad \text{(G.16)}
$$

which contradicts Equation (G.9) and thus concludes the proof. QED.

G.2.3 Tightness of Per-state Action Certification in TPARL

Proof of Proposition 9.2. For ease of notation, we let $w = \min\{W, t+1\}$ so w is the actual window size used at step t. We let $t_0 = t - w + 1$, i.e., t_0 is the actual start time step for TPARL aggregation at step t. Now we can write the chosen action at step t without poisoning as $a = \pi_{\mathrm{T}}(s_{t_0:t})$.

We prove by construction, i.e., we construct an poisoning attack with poisoning size $\bar{K}_t + 1$ that deviates the prediction of the poisoned policy. Specifically, we aim to craft a poisoned dataset $\widetilde{D}$ with poisoning size $|D \ominus \widetilde{D}| = \bar{K}_t + 1$, such that for certain learning algorithm $\mathcal{M}_0$, after partitioning and learning on the poisoned dataset, the poisoned subpolicies $\tilde{\pi}_i = \mathcal{M}_0(\widetilde{D}_i)$ can be aggregated to produce different action prediction: $\widetilde{\pi_{\mathrm{T}}}(s_{t_0:t}) \neq a$.

Before construction, we first show that $\bar{K}_t$ given by Equation (9.7) satisfies $\bar{K}_t < u$. If

$\bar{K}_t = u$, it means that for arbitrary $a' \neq a$,

$$\sum_{i=1}^{u} h_{a,a'}^{(i)} = \sum_{i=0}^{u-1} h_{i,a,a'} = \sum_{i=0}^{u-1} \left(\sum_{j=0}^{w-1} \mathbb{I}_{i,a}(s_{t-j}) + w - \sum_{j=0}^{w-1} \mathbb{I}_{i,a'}(s_{t-j}) \right) = n_a(s_{t_0:t}) + uw - n_{a'}(s_{t_0:t})$$
$$\leqslant \delta_{a,a'} = n_a(s_{t_0:t}) - n_{a'}(s_{t_0:t}) - \mathbb{I}[a' < a], \tag{G.17}$$

which implies $uw \leqslant 0$ contracting $uw > 0$. Now, we know $\bar{K}_t < u$, so $\bar{K}_t + 1$, our poisoning size, is smaller or equal to u.

We start our construction by choosing an action a^T:

$$a^T = \underset{a' \neq a, a' \in \mathcal{A}}{\arg\min} \quad \underset{\sum_{i=1}^{p} h_{a,a'}^{(i)} \leqslant \delta_{a,a'}}{\arg\max} \quad p. \tag{G.18}$$

According to the definition of $\bar{K}_t$ in Equation (9.7), for a^T we have

$$\sum_{i=1}^{\bar{K}_t+1} h_{a,a^T}^{(i)} > \delta_{a,a^T}. \tag{G.19}$$

We locate the subpolicies to poison as

$$B = \{i \in [u] \mid h_{a,a^T}^{i} \text{ is } h_{a,a^T}^{(j)} \text{ in the nonincreasing permutation in Equation (9.8)}, j \leqslant \bar{K}_t + 1\} \tag{G.20}$$

Therefore, $|B| = \bar{K}_t + 1$ and

$$\sum_{i \in B} h_{i,a,a^T} > \delta_{a,a^T} \tag{G.21}$$

by Equation (G.19). For each of these subpolicies $i \in B$, we locate its corresponding partitioned dataset D_i for training subpolicy π_i, and insert one trajectory p_i to D_i, such that our chosen learning algorithm $\mathcal{M}_0$ can train a subpolicy $\tilde{\pi}_i = \mathcal{M}(D_i \cup \{p_i\})$ satisfying $\tilde{\pi}_i(s_{t'}) = a^T$ for any $t' \in [t_0, t]$. For example, the trajectory could be $p_i = \{(s_{t'}, a^T, s', \infty)\}_{t'=t_0}^{t}$ and $\mathcal{M}_0$ learns the action with maximum reward for the memorized nearest state, where s' can be adjusted to make sure the trajectory is hashed to partition i. Then, we construct the poisoned dataset

$$\widetilde{D} = \left(\bigcup_{i \in B'} D_i \cup \{p_i\} \right) \cup \left(\bigcup_{i \in [u] \setminus B'} D_i \right). \tag{G.22}$$

Therefore, we have $\widetilde{D} \ominus D = \cup_{i \in B'} p_i = |B'| = \bar{K}_t + 1$. Now, we compare the aggregated

votes for a and a^T before and after poisoning:

$$\widetilde{n}_a(s_{t_0:t}) - n_a(s_{t_0:t}) = \sum_{i\in B}\left(\sum_{j=0}^{w-1}\mathbb{I}[\widetilde{\pi}_i(s_{t-j}) = a] - \sum_{j=0}^{w-1}\mathbb{I}[\pi_i(s_{t-j}) = a]\right) \tag{G.23}$$

$$= -\sum_{i\in B}\sum_{j=0}^{w-1}\mathbb{I}[\pi_i(s_{t-j}) = a], \tag{G.24}$$

$$\widetilde{n}_{a^T}(s_{t_0:t}) - n_{a^T}(s_{t_0:t}) = \sum_{i\in B}\left(\sum_{j=0}^{w-1}\mathbb{I}[\widetilde{\pi}_i(s_{t-j}) = a^T] - \sum_{j=0}^{w-1}\mathbb{I}[\pi_i(s_{t-j}) = a^T]\right) \tag{G.25}$$

$$= \sum_{i\in B}\left(w - \sum_{j=0}^{w-1}\mathbb{I}[\pi_i(s_{t-j}) = a^T]\right). \tag{G.26}$$

Now we compare the margin between aggregated votes for a and a^T after poisoning:

$$\widetilde{n}_{a^T}(s_{t_0:t}) - \widetilde{n}_a(s_{t_0:t}) + \mathbb{I}[a^T < a] \tag{G.27}$$

$$= n_{a^T}(s_{t_0:t}) - n_a(s_{t_0:t}) + \mathbb{I}[a^T < a] + \widetilde{n}_{a^T}(s_{t_0:t}) - n_{a^T}(s_{t_0:t}) - \big(\widetilde{n}_a(s_{t_0:t}) - n_a(s_{t_0:t})\big) \tag{G.28}$$

$$= -\delta_{a,a^T} + \sum_{i\in B}\left(\sum_{j=0}^{w-1}\mathbb{I}[\pi_i(s_{t-j}) = a] + w - \sum_{j=0}^{w-1}\mathbb{I}[\pi_i(s_{t-j}) = a^T]\right) \tag{G.29}$$

$$= -\delta_{a,a^T} + \sum_{i\in B} h_{i,a,a^T} > 0. \tag{G.30}$$

As a result, after poisoning, a^T has higher priority to be chosen than a, i.e., $\widetilde{\pi_{\mathrm{T}}}(s_{t_0:t}) \neq a = \pi_{\mathrm{T}}(s_{t_0:t})$. To this point, we conclude the proof with a feasible attack construction. QED.

G.2.4 Loose Per-state Action Certification in TPARL and Comparision with Tight One

The following corollary of Theorem 9.1 states a loose per-state action certification.

Corollary G.1. Under the same condition as Theorem 9.2,

$$\bar{K}_t = \left\lfloor \frac{n_a(s_{\max\{t-W+1,0\}:t}) - \max_{a'\neq a}\left(n_{a'}(s_{\max\{t-W+1,0\}:t}) + \mathbb{I}[a' < a]\right)}{2\min\{W, t+1\}} \right\rfloor \tag{G.31}$$

is a *tolerable poisoning threshold* at time step t in Definition 9.1, where W is the window size.

Proof of Corollary G.1. We let $w = \min\{t+1, W\}$ to be the actual aggregation window

size at time step t. After poisoning with size K, at most K subpolicies are affected and each affected policy can only make $\pm w$ changes to the aggregated action count. This implies that, after poisoning, for any action $a' \in \mathcal{A}$, the aggregated vote count $\widetilde{n}_{a'}(s_{t-w+1:t}) \in [n_{a'}(s_{t-w+1:t}) - uw, n_{a'}(s_{t-w+1:t}) + uw]$. Thus, when $K \leqslant \bar{K}_t$ where $\bar{K}_t$ is defined in Theorem 9.2, for any $a' \neq a$, we have

$$\tilde{n}_a(s_{t-w+1:t}) - \tilde{n}_{a'}(s_{t-w+1:t}) - \mathbb{I}[a' < a] \tag{G.32}$$

$$= n_a(s_{t-w+1:t}) - n_{a'}(s_{t-w+1:t}) - \mathbb{I}[a' < a] - 2Kw \tag{G.33}$$

$$\geqslant n_a(s_{t-w+1:t}) - n_{a'}(s_{t-w+1:t}) - \mathbb{I}[a' < a] - 2w \cdot \bar{K}_t \tag{G.34}$$

$$\geqslant n_a(s_{t-w+1:t}) - n_{a'}(s_{t-w+1:t}) - \mathbb{I}[a' < a] - \left(n_a(s_{t-w+1:t}) - \max_{a'' \neq a} (n_{a''}(s_{t-w+1:t}) + \mathbb{I}[a'' < a]) \right) \tag{G.35}$$

$$= \max_{a'' \neq a} (n_{a''}(s_{t-w+1:t}) + \mathbb{I}[a'' < a]) - \big(n_{a'}(s_{t-w+1:t}) + \mathbb{I}[a' < a] \big) \geqslant 0. \tag{G.36}$$

From the definition of TPARL protocol, the poisoned policy still chooses action a, which implies $\bar{K}_t$ is a tolerable poisoning threshold. QED.

In the main text, we mention that the certification from Corollary G.1 is looser than that from Theorem 9.2. This assertion is based on the following two facts:

1. According to Proposition 9.2, the certification given by Theorem 9.2 is theoretically tight, which means that any other certification can only be as tight as Theorem 9.2 or looser than it.

2. There exists examples where the computed $\bar{K}_t$ from Theorem 9.2 is larger than that from Corollary G.1.
   For instance, suppose $W = 5$, action set $\mathcal{A} = \{a_1, a_2\}$, and there are three subpolicies. At time step $t = 4$, π_i for s_0 to s_4 are $[a_1, a_1, a_1, a_1, a_2]$ for all subpolicies (i.e., $i \in [3]$). Thus, the benign policy $\pi_{\mathrm{T}}(s_{0:4}) = a_1$. By computation, the $\bar{K}_t$ from Theorem 9.2 is 1; while the $\bar{K}_t$ from Corollary G.1 is 0.

Indeed, Corollary G.1 can be viewed as using $2w$ to upper bound $h_{i,a,a'} = w + \sum_{j=0}^{w-1} \mathbb{I}_{i,a}(s_{t-j}) - \sum_{j=0}^{w-1} \mathbb{I}_{i,a'}(s_{t-j})$. Intuitively, Corollary G.1 assumes every subpolicy can provide $2w$ vote margin shrinkage, and Theorem 9.2 uses $h_{i,a,a'}$ to capture the precise worse-case margin shrinkage and thus provides a tighter certification.

G.2.5 Per-state Action Certification in DPARL

Proof of Theorem 9.3. Without loss of generality, we assume $W_{\max} \leqslant t+1$ and otherwise we let $W_{\max} \leftarrow \min\{W_{\max}, t+1\}$. We let $t_0 = \max\{t - W_{\max} + 1, 0\}$ be the start time step of the maximum possible window. To prove the theorem, our general methodology is to enumerate all possible cases of a successful attack, and derive the tolerable poisoning threshold for each case respectively. Taking a minimum over these tolerable poisoning thresholds gives the required result.

Specifically, we denote $\mathcal{P}$ to the predicate of robustness under poisoning attack: $\mathcal{P} = [\widetilde{\pi_{\mathrm{D}}}(s_{t_0:t}) = a]$, and denote K to the poisoning attack size. Therefore, we can decompose $\mathcal{P}$ as such:

$$\mathcal{P} = \mathcal{P}(W') \wedge \bigwedge_{\substack{1 \leqslant W^* \leqslant W_{\max}, W^* \neq W' \\ a' \neq a}} \neg \mathcal{Q}(W^*, a'). \tag{G.37}$$

Recall that W' is the chosen window size by the protocol DPARL with unattacked subpolicies π_{D} (Equation (9.4)). In Equation (G.37), the predicate $\mathcal{P}(W')$ means that after poisoning attack, whether the prediction under window size W' is still a; the predicate $\mathcal{Q}(W^*, a')$ means that after poisoning attack, whether the chosen action is a' at window size W^* and average vote margin is larger (or equal if $a' < a$) at window size W^* compared to W'. Formally, let $\widetilde{n}_a$ be the aggregated action count after poisoning and $\widetilde{\Delta}_t^W$ be the average vote margin after poisoning at window W (see Equation (9.5)),

$$\mathcal{P}(W') = \left(\arg\max_{a \in \mathcal{A}} \widetilde{n}_a(s_{t-W'+1:t}) = a \right), \tag{G.38}$$

$$\begin{aligned} \mathcal{Q}(W^*, a') = &\left(\arg\max_{a \in \mathcal{A}} \widetilde{n}_a(s_{t-W^*+1:t}) = a' \right) \wedge \\ &\left(\left(\left(\widetilde{\Delta}_t^{W^*} \geqslant \widetilde{\Delta}_t^{W'} \right) \wedge (a' < a) \right) \vee \left(\left(\widetilde{\Delta}_t^{W^*} > \widetilde{\Delta}_t^{W'} \right) \wedge (a' > a) \right) \right). \end{aligned} \tag{G.39}$$

According to Theorem 9.2,

$$K \leqslant \bar{K}_t \Longrightarrow \mathcal{P}(W') \tag{G.40}$$

where $\bar{K}_t$ is defined by Equation (9.7) with W replaced by W'. The following Lemma G.1 shows a sufficient condition for $Q(W^*, a')$. We then aggregate these conditions together with minimum to obtain a sufficient condition for $\mathcal{P}$:

$$K \leqslant \min \left\{ \bar{K}_t, \min_{1 \leqslant W^* \leqslant \min\{W_{\max}, t+1\}, W^* \neq W', a' \neq a, a'' \neq a} L_{a', a''}^{W^*, W'} \right\} \tag{G.41}$$

and thus conclude the proof. QED.

Lemma G.1. Let $\mathcal{Q}(W^*, a')$, K, W' be the same as defined in proof of Theorem 9.3, then

$$K \leqslant \min_{a'' \neq a} L_{a',a''}^{W^*,W'} \Longrightarrow \neg \mathcal{Q}(W^*, a'), \tag{G.42}$$

where $L_{a',a''}^{W^*,W'}$ is defined in Definition 9.7.

Proof. We prove the equivalent form:

$$\mathcal{Q}(W^*, a') \Longrightarrow K > \min_{a'' \neq a} L_{a',a''}^{W^*,W'}. \tag{G.43}$$

Suppose a poisoning attack can successfully achieve $Q(W^*, a')$, we now induce the requirement on its poisoning size K. First, we notice that

$$\begin{aligned} &\left(\left(\widetilde{\Delta}_t^{W^*} \geqslant \widetilde{\Delta}_t^{W'}\right) \wedge (a' < a)\right) \vee \left(\left(\widetilde{\Delta}_t^{W^*} > \widetilde{\Delta}_t^{W'}\right) \wedge (a' > a)\right) \\ \Longleftrightarrow\ & W^*W'\widetilde{\Delta}_t^{W^*} \geqslant W^*W'\widetilde{\Delta}_t^{W'} + \mathbb{I}[a' > a] \end{aligned} \tag{G.44}$$

since $W^*W'\widetilde{\Delta}_t^{W*/W^*}$ is an integer by definition. According to the definition and $\mathcal{Q}(W^*, a')$'s assumption that $\arg\max_{a \in \mathcal{A}} \widetilde{n}_a(s_{t-W^*+1:t}) = a'$,

$$W^*W'\widetilde{\Delta}_t^{W^*} \leqslant W'\left(\widetilde{n}_{a'}(s_{t-W^*+1:t}) - \widetilde{n}_{a^\#}(s_{t-W^*+1:t})\right), \tag{G.45}$$

where "$\leqslant$" comes from the fact that the margin in $\widetilde{\Delta}_t^{W^*}$ should be with respect to the runner-up class after poisoning, and computing with respect to any other class provides an upper bound. Here we choose $a^\# = \arg\max_{a_0 \neq a', a_0 \in \mathcal{A}} n_{a_0}(s_{t-W^*+1:t})$ (see Definition 9.7), the runner-up class before poisoning, to empirically shrink the gap between the bound and actual margin. On the other hand,

$$W^*W'\widetilde{\Delta}_t^{W'} \geqslant W^*\left(\widetilde{n}_a(s_{t-W'+1:t}) - \max_{a'' \neq a} \widetilde{n}_{a''}(s_{t-W'+1:t})\right), \tag{G.46}$$

where "$\geqslant$" comes from the fact that the margin in $\widetilde{\Delta}_t^{W'}$ should use the top class after poisoning, and computing with any other class provides a lower bound. Thus, from Equation (G.44) and the above two relaxations, we get

$$Q(W^*, a') \tag{G.47}$$

$$\Longrightarrow W'\left(\widetilde{n}_{a'}(s_{t-W^*+1:t}) - \widetilde{n}_{a^\#}(s_{t-W^*+1:t})\right) \geqslant W^*\left(\widetilde{n}_a(s_{t-W'+1:t}) - \max_{a'' \neq a} \widetilde{n}_{a''}(s_{t-W'+1:t})\right) + \mathbb{I}[a' > a] \tag{G.48}$$

$$\Longrightarrow \exists a'' \neq a,$$
$$W'\left(\widetilde{n}_{a'}(s_{t-W^*+1:t}) - \widetilde{n}_{a^\#}(s_{t-W^*+1:t})\right) \geqslant W^*\left(\widetilde{n}_{a}(s_{t-W'+1:t}) - \widetilde{n}_{a''}(s_{t-W'+1:t})\right) + \mathbb{I}[a' > a]. \tag{G.49}$$

For each $a'' \neq a$, now we use the last equation as the condition, and show that $K > L^{W^*,W'}_{a',a''}$ is a necessary condition. This proposition is equivalent to

$$\begin{aligned} &K \leqslant L^{W^*,W'}_{a',a''} \\ \Longrightarrow &W'\left(\widetilde{n}_{a'}(s_{t-W^*+1:t}) - \widetilde{n}_{a^\#}(s_{t-W^*+1:t})\right) < W^*\left(\widetilde{n}_{a}(s_{t-W'+1:t}) - \widetilde{n}_{a''}(s_{t-W'+1:t})\right) + \mathbb{I}[a' > a]. \end{aligned} \tag{G.50}$$

Suppose a poisoning attack within poisoning size K changes the subpolicies in set $B \subseteq [u]$. Note that $|B| \leqslant K$. We inspect the objective in Equation (G.50):

$$W'\left(\widetilde{n}_{a'}(s_{t-W^*+1:t}) - \widetilde{n}_{a^\#}(s_{t-W^*+1:t})\right) - W^*\left(\widetilde{n}_{a}(s_{t-W'+1:t}) - \widetilde{n}_{a''}(s_{t-W'+1:t})\right) - \mathbb{I}[a' > a] \tag{G.51}$$

$$\begin{aligned} =&W'\left(n_{a'}(s_{t-W^*+1:t}) - n_{a^\#}(s_{t-W^*+1:t})\right) - W^*\left(n_{a}(s_{t-W'+1:t}) - n_{a''}(s_{t-W'+1:t})\right) - \mathbb{I}[a' > a] \\ &+ \sum_{i\in B}\left(W'\sum_{w=0}^{W^*-1}\mathbb{I}[\tilde{\pi}_i(s_{t-w+1}) = a'] - W'\sum_{w=0}^{W^*-1}\mathbb{I}[\tilde{\pi}_i(s_{t-w+1}) = a^\#]\right. \\ &\qquad \left. -W^*\sum_{w=0}^{W'-1}\mathbb{I}[\tilde{\pi}_i(s_{t-w+1}) = a] + W^*\sum_{w=0}^{W'-1}\mathbb{I}[\tilde{\pi}_i(s_{t-w+1}) = a'']\right) \\ &- \sum_{i\in B}\left(W'\sum_{w=0}^{W^*-1}\mathbb{I}[\pi_i(s_{t-w+1}) = a'] - W'\sum_{w=0}^{W^*-1}\mathbb{I}[\pi_i(s_{t-w+1}) = a^\#]\right. \\ &\qquad \left. -W^*\sum_{w=0}^{W'-1}\mathbb{I}[\pi_i(s_{t-w+1}) = a] + W^*\sum_{w=0}^{W'-1}\mathbb{I}[\pi_i(s_{t-w+1}) = a'']\right) \end{aligned} \tag{G.52}$$

$$\begin{aligned} =&W'\left(n_{a'}(s_{t-W^*+1:t}) - n_{a^\#}(s_{t-W^*+1:t})\right) - W^*\left(n_{a}(s_{t-W'+1:t}) - n_{a''}(s_{t-W'+1:t})\right) - \mathbb{I}[a' > a] \\ &+ \sum_{i\in B}\sum_{w=0}^{\max\{W^*,W'\}}\sigma^w(\tilde{\pi}_i(s_{t-w})) - \sigma^w(\pi_i(s_{t-w})) \end{aligned} \tag{G.53}$$

$$\begin{aligned} \leqslant&W'\left(n_{a'}(s_{t-W^*+1:t}) - n_{a^\#}(s_{t-W^*+1:t})\right) - W^*\left(n_{a}(s_{t-W'+1:t}) - n_{a''}(s_{t-W'+1:t})\right) - \mathbb{I}[a' > a] \\ &+ \sum_{i\in B}\underbrace{\sum_{w=0}^{\max\{W^*,W'\}}\max_{a_0\in\mathcal{A}}\sigma^w(a_0) - \sigma^w(\pi_i(s_{t-w}))}_{g_i} \end{aligned} \tag{G.54}$$

$$\overset{(a)}{\leqslant} W'\left(n_{a'}(s_{t-W^*+1:t}) - n_{a^\#}(s_{t-W^*+1:t})\right)$$

$$- W^* \left(n_a(s_{t-W'+1:t}) - n_{a''}(s_{t-W'+1:t})\right) - \mathbb{I}[a' > a] + \sum_{i=1}^{K} g^{(i)} \tag{G.55}$$

$$\overset{(b)}{\leqslant} W' \left(n_{a'}(s_{t-W^*+1:t}) - n_{a^\#}(s_{t-W^*+1:t})\right)$$

$$- W^* \left(n_a(s_{t-W'+1:t}) - n_{a''}(s_{t-W'+1:t})\right) - \mathbb{I}[a' > a] + \sum_{i=1}^{L^{W^*,W'}_{a',a''}} g^{(i)} \tag{G.56}$$

$$\overset{(c)}{<} 0. \tag{G.57}$$

Thus, Equation (G.50) is proved. Therefore, $K > L^{W^*,W'}_{a',a''}$ is a necessary condition for $Q(W^*, a')$, i.e., Equation (G.43).

In the above deriviation, the definitions of g^i, g^i, and σ^w are from Equation (9.11). (a) comes from the facts that $\{g^{(i)}\}_{i=1}^{u}$ is a nondecreasing permutation of $\{g_i\}_{i=0}^{u-1}$, $g_i \geqslant 0$, and $|B| \leqslant K$. (b) comes from the assumption $K \leqslant L^{W^*,W'}_{a,a''}$ and also $g^{(i)} \geqslant 0$. (c) comes from the definition in Equation (9.10). QED.

G.2.6 Possible Action Set in PARL and Comparison

Certification

Proof of Theorem 9.4. According to the definition of possible action set, we only need to prove the contrary: for any $a \in \mathcal{A} \backslash A^T(K)$, within poisoning size K, the poisoned policy cannot choose a: $\widetilde{\pi_{\mathrm{P}}}(s_t) \neq a$.

According to Equation (9.12), any $a \in \mathcal{A} \backslash A^T(K)$ satisfies

$$\sum_{a' \in \mathcal{A}} \max\{n_{a'}(s_t) - n_a(s_t) - K + \mathbb{I}[a' < a], 0\} > K. \tag{G.58}$$

Given poisoning size K, since each poisoning size can affect only one subpolicy, we know

$$\widetilde{n}_a(s_t) \leqslant n_a(s_t) + K \tag{G.59}$$

where $\widetilde{n}_a$ denotes to the poisoned aggregated action count. We suppose the attack could be successful, then for $a' < a$, $\widetilde{n}_{a'}(s_t) \leqslant \widetilde{n}_a(s_t) - 1$, and thus $n_{a'}(s_t) - \widetilde{n}_{a'}(s_t) \geqslant n_{a'}(s_t) - n_a(s_t) - K + 1$. Similarly, for $a' > a$, $\widetilde{a'}(s_t) \leqslant \widetilde{n}_a(s_t)$, and thus $n_{a'}(s_t) - \widetilde{n}_{a'}(s_t) \geqslant n_{a'}(s_t) - n_a(s_t) - K$. Also, for any $a' \neq a$ after poisoning $n_{a'}(s_t) - \widetilde{n}_{a'}(s_t) \geqslant 0$; otherwise deviating the difference subpolicies' decisions' from a' to a is strictly no-worse. Given these facts, the amount of votes that need to be reduced is the LHS of Equation (G.58) which is larger than K.

However, we only have K poisoning size, i.e., K votes that can be reduced. As a result, our assumption that the attack could be successful is falsified and the poisoned policy cannot choose a. QED.

Corollary G.2 (Loose PARL Action Set)**.** Under the condition of Definition 9.8, suppose the aggregation protocol is PARL as defined in Definition 9.4, then the *possible action set* at step t

$$A^L(K) = \left\{ a \in \mathcal{A} \,\middle|\, \max_{a' \in \mathcal{A}} n_{a'}(s_t) - n_a(s_t) \leqslant 2K - \mathbb{I}[a > \arg\max_{a' \in \mathcal{A}} n_{a'}(s_t)] \right\}. \tag{G.60}$$

Proof of Corollary G.2. Again, we prove the contrary, for any $a \in \mathcal{A} \backslash A^L(K)$, within poisoning size K, the poisoned policy cannot choose a: $\widetilde{\pi_{\mathrm{P}}}(s_t) \neq a$.

According to Equation (G.60), let $a_m = \arg\max_{a' \in \mathcal{A}} n_{a'}(s_t)$, then any $a \in \mathcal{A} \backslash A^L(K)$ satisfies

$$n_{a_m}(s_t) - n_a(s_t) > 2K - \mathbb{I}[a > a_m]. \tag{G.61}$$

After poisoning, we thus have

$$n_{a_m}(s_t) - n_a(s_t) > -\mathbb{I}[a > a_m] \Longrightarrow n_{a_m}(s_t) - n_a(s_t) \geqslant 1 - \mathbb{I}[a > a_m] \tag{G.62}$$

From the definition, $a \notin A^L(K)$ so $a \neq a_m$. If $a_m < a$, $n_{a_m}(s_t) \geqslant n_a(s_t)$; if $a_m > a$, $n_{a_m}(s_t) > n_a(s_t)$. In both cases, a_m has higher priority to be chosen than a, and thus the poisoned policy cannot choose a. QED.

Comparison

Theorem G.1. Under the condition of Definition 9.8, suppose the aggregation protocol is PARL as defined in Definition 9.4, $A^T(K), A^L(K)$ are defined according to Equations (9.12) and (G.60) accordingly, then

1. $A^T(K) \subseteq A^L(K)$; and there are subpolicies $\{\pi_i\}_{i=0}^{u-1}$ and state s_t such that $A^T(K) \subsetneq A^L(K)$.

2. Given subpolicies $\{\pi_i\}_{i=0}^{u-1}$ and state s_t, for any $a \in A^T(K)$, there exists a poisoned training set $\widetilde{D}$ whose poisoning size $|D \ominus \widetilde{D}| \leqslant K$ and some RL training mechanism, such that $\widetilde{\pi_{\mathrm{P}}}(s_t) = a$ where $\widetilde{\pi_{\mathrm{P}}}$ is the poisoned PARL policy trained on $\widetilde{D}$.

Proof of Theorem G.1. We prove the two arguments separately.

1. We first prove $A^T(K) \subseteq A^L(K)$. For any $a \in A^T(K)$, let $a_m = \arg\max_{a' \in \mathcal{A}} n_{a'}(s_t)$, then

$$\sum_{a' \in \mathcal{A}} \max\{n_{a'}(s_t) - n_a(s_t) - K + \mathbb{I}[a' < a], 0\} \leqslant K \tag{G.63}$$

$$\Longrightarrow n_{a_m}(s_t) - n_a(s_t) - K + \mathbb{I}[a_m < a] \leqslant K \tag{G.64}$$

$$\Longrightarrow n_{a_m}(s_t) - n_a(s_t) \leqslant 2K - \mathbb{I}[a > a_m] \tag{G.65}$$

where the last proposition is exactly the set selector of $A^L(K)$ so $a \in A^L(K)$.

We then prove that $A^T(K) \subsetneq A^L(K)$ can happen by construction. Suppose that there are three actions in the action space: $\mathcal{A} = \{a_1, a_2, a_3\}$. We construct subpolicies for current state s_t such that the aggregated action counts are

$$n_{a_1}(s_t) = 10, n_{a_2}(s_t) = 9, n_{a_3}(s_t) = 1. \tag{G.66}$$

Given poisoning size $K = 5$, we find that

$$n_{a_1}(s_t) - n_{a_3}(s_t) = 9 \leqslant 10 - \mathbb{I}[a_3 \geqslant a_1] = 9 \qquad \Longrightarrow a_3 \in a^L(K), \tag{G.67}$$

$$\big(n_{a_1}(s_t) - n_{a_3}(s_t) - K + \mathbb{I}[a_1 < a_3]\big) +$$
$$\big(n_{a_2}(s_t) - n_{a_3}(s_t) - K + \mathbb{I}[a_2 < a_3]\big) = 9 > K = 5 \qquad \Longrightarrow a_3 \notin a^T(K). \tag{G.68}$$

Therefore, $A^T(K) \subsetneq A^L(K)$ for these subpolicies and state s_t.

2. We prove by construction. For any $a \in A^T(K)$, we construct the set of subpolicies to poison $B^a \subseteq [u]$ such that $|B^a| \leqslant K$, then describe the corresponding poisoned dataset $\widetilde{D}^a$, and finally prove that the poisoned policy $\widetilde{\pi_{\mathrm{P}}}^a(s_t) = a$.

For $a \in A^T(K)$, by definition (Equation (9.12)), we know

$$\sum_{a' \in \mathcal{A}} \max\{\underbrace{n_{a'}(s_t) - n_a(s_t) - K + \mathbb{I}[a' < a]}_{:= t_{a,a'}}, 0\} \leqslant K. \tag{G.69}$$

We now define a set of actions $C^a \subseteq \mathcal{A}$ such that

$$C^a = \{a' \in \mathcal{A} \mid t_{a,a'} > 0\}. \tag{G.70}$$

According to this definition, $a \notin C^a$ since $t_{a,a} \leqslant 0$.

Fact G.1. For $a \in A^T(K)$ and any $a' \in \mathcal{A}$, $n_a(s_t) + K - \mathbb{I}[a' < a] \geqslant 0$.

Proof of Fact G.1. Suppose $n_a(s_t) + K - \mathbb{I}[a' < a] \leqslant 0$, then $n_a(s_t) = K = 0$ and $a' < a$. Then

$$\sum_{a' \in \mathcal{A}} \max\{n_{a'}(s_t) - n_a(s_t) - K + \mathbb{I}[a' < a], 0\} \geqslant \sum_{a' \in \mathcal{A}} n_{a'}(s_t) - n_a(s_t) - K = \sum_{a' \in \mathcal{A}} n_{a'}(s_t) = u > 0 \tag{G.71}$$

which contradicts the requirement that the LHS of the above inequality should be $\leqslant K = 0$. QED.

Give Fact G.1, for any $a' \in C^a$, $n_{a'}(s_t) \geqslant t_{a,a'}$. Notice that $n_{a'}(s_t)$ is the number of subpolicies that vote for action a' at state s_t. Therefore, we can pick an arbitrary subset of those subpolicies whose cardinality is $t_{a,a'}$. We denote $B^a_{a'}$ to such subset:

$$B^a_{a'} \subseteq \{i \in [u] \mid \pi_i(s_t) = a'\}, |B^a_{a'}| = t_{a,a'}. \tag{G.72}$$

Now define B^a_α: $B^a_\alpha = \bigcup_{a' \in C^a} B^a_{a'}$. We construct B^a_β to be an arbitrary subset of those subpolicies whose prediction is not a and who are not in B^a_α, and limit the B^a_β's cardinality:

$$B^a_\beta \subseteq \{i \in [u] \mid \pi_i(s_t) \neq a\} \backslash B^a_\alpha, |B^a_\beta| = \min\{K - |B^a_\alpha|, u - n_a(s_t) - |B^a_\alpha|\}. \tag{G.73}$$

Such B^a_β can be selected, because:

- From definition, $B^a_\alpha \subseteq \{i \in [u] \mid \pi_i(s_t) \neq a\}$, where the cardinality of $\{i \in [u] \mid \pi_i(s_t) \neq a\}$ is $u - n_a(s_t)$. So $0 \leqslant u - n_a(s_t) - |B^a_\alpha|$.
- Since
$$|B^a_\alpha| \leqslant \sum_{a' \in C^a} |B^a_{a'}| = \sum_{a' \in C^a} t_{a,a'} = \sum_{a' \in \mathcal{A}, t_{a,a'} > 0} t_{a,a'} \overset{(*)}{\leqslant} K, \tag{G.74}$$
$K - |B^a_\alpha| \geqslant 0$, and thus $|B^a_\beta| \geqslant 0$. Here $(*)$ is due to Equation (G.69).
- The superset $\{i \in [u] \mid \pi_i(s_t) \neq a\} \backslash B^a_\alpha$ has cardinality $u - n_a(s_t) - |B^a_\alpha|$ and $|B^a_\beta| \leqslant u - n_a(s_t) - |B^a_\alpha|$.

To this point, we can define the set of subpolicies to poison:

$$B^a := B^a_\alpha \cup B^a_\beta, \tag{G.75}$$

and $|B^a| = |B^a_\alpha| + |B^a_\beta| \leqslant K$.

In a similar fashion as the attack construction in proof of Proposition 9.1, for each $i \in B^a$, we locate its corresponding partitioned dataset D_i for training subpolicy π^a_i. We inset

one trajectory p_i^a to D_i such that our chosen learning algorithm $\mathcal{M}_0$ can train a subpolicy $\tilde{\pi}_i^a = \mathcal{M}_0(D_i \cup \{p_i^a\})$ such that $\tilde{\pi}_i^a(s_t) = a$. For example, the trajectory could be $p_i^a = \{(s_t, a, s', \infty)\}$ where s' is an arbitrary state that guarantees p_i^a is hashed to partition i; and $\mathcal{M}_0$ learns the action with maximum reward for the memorized nearest state. Then, the poisoned dataset

$$\widetilde{D}^a = \left(\bigcup_{i \in B^a} D_i \cup \{p_i^a\} \right) \cup \left(\bigcup_{i \in [u] \backslash B'} D_i \right). \tag{G.76}$$

Thus, $\widetilde{D}^a$ is $|D \ominus \widetilde{D}^a| = |B^a| \leqslant K$, i.e., the constructed attack's poisoning size is within K.

We now analyze the action prediction of the poisoned policy $\widetilde{\pi_{\mathrm{P}}}$. For action a, after poisoning, $\widetilde{n}_a(s_t) = n_a(s_t) + |B^a| = n_a(s_t) + \min\{K, u - n_a(s_t)\} = \min\{n_a(s_t) + K, u\}$. If $\widetilde{n}_a(s_t) = u$, then all subpolicies vote for a, apparently $\widetilde{\pi_{\mathrm{P}}}^a(s_t) = a$; otherwise, $\widetilde{n}_a(s_t) = n_a(s_t) + K$. In this case, for any action $a' \in C^a$, since we at least choose subpolicies in $B_{a'}^a$ and change their action prediction to a, the aggregated action count after poisoning is $\widetilde{n}_{a'}(s_t) \leqslant n_{a'}(s_t) - |B_{a'}^a| = n_{a'}(s_t) - t_{a,a'} = n_{a'}(s_t) - n_{a'}(s_t) + n_a(s_t) + K - \mathbb{I}[a' < a] = n_a(s_t) + K - \mathbb{I}[a' < a] = \tilde{n}_a(s_t) - \mathbb{I}[a' < a]$. Thus, a' has lower priority to be chosen than a. For any action $a' \notin C^a$ and $a' \neq a$, by the definition of C^a, $t_{a,a'} = n_{a'}(s_t) - n_a(s_t) - K + \mathbb{I}[a' < a] \leqslant 0$. After poisoning, the vote of a' does not increase, i.e., $\widetilde{n}_{a'}(s_t) \leqslant n_{a'}(s_t) \leqslant n_a(s_t) + K - \mathbb{I}[a' < a] = \widetilde{n}_a(s_t) - \mathbb{I}[a' < a]$. Thus, a' also has lower priority to be chosen than a. In conclusion, we have $\widetilde{\pi_{\mathrm{P}}}^a(s_t) = a$.

To this point, for any $a \in A^T(K)$, we successfully construct the corresponding poisoned dataset $\widetilde{D}^a$ within poisoning size K such that $\widetilde{\pi_{\mathrm{P}}}^a(s_t) = a$, thus concludes the proof.

QED.

G.2.7 Possible Action Set in TPARL

Proof of Theorem 9.5. For ease of notation, we let $w = \min\{W, t+1\}$, so w is the actual window size used at step t. We let $t_0 = t - w + 1$, i.e., t_0 is the actual start time step for TPARL aggregation at our current step t. Now we can write chosen action at step t without poisoning as $\pi_{\mathrm{T}}(s_{t_0:t})$.

We only need to prove the contrary: for any $a \in \mathcal{A} \backslash A(K)$, within poisoning size K, the poisoned policy cannot choose a: $\widetilde{\pi_{\mathrm{T}}}(s_{t_0:t}) \neq a$.

According to Equation (9.13), for such a, there exists $a' \neq a$ such that

$$\sum_{i=1}^{K} h_{a',a}^{(i)} \leqslant \delta_{a',a} = n_{a'}(s_{t_0:t}) - n_a(s_{t_0:t}) - \mathbb{I}[a < a']. \tag{G.77}$$

Suppose there exists such poisoning attack that lets $\widetilde{\pi_{\mathrm{T}}}(s_{t_0:t}) = a$. This implies that for a', after poisoning, we have

$$\widetilde{n}_{a'}(s_{t_0:t}) - \widetilde{n}_{a'}(s_{t_0:t}) - \mathbb{I}[a' < a] < 0, \tag{G.78}$$

where $\widetilde{n}_a$ is the aggregated action count after poisoning. Since the poisoning size is within K, it can affect at most K subpolicies. We let $B \subseteq [u], |B| \leqslant K$ to represent the affected subpolicy set. Therefore,

$$\begin{aligned}
&\widetilde{n}_{a'}(s_{t_0:t}) - \widetilde{n}_a(s_{t_0:t}) - \mathbb{I}[a' < a] \\
=&\underbrace{n_{a'}(s_{t_0:t}) - n_a(s_{t_0:t}) - \mathbb{I}[a' < a]}_{\delta_{a',a}} + (\widetilde{n}_{a'}(s_{t_0:t}) - n_{a'}(s_{t_0:t})) - (\widetilde{n}_a(s_{t_0:t}) - n_a(s_{t_0:t})) && \text{(G.79)} \\
=&\delta_{a',a} + \sum_{i\in B}\left(\sum_{j=0}^{w-1} \mathbb{I}[\tilde{\pi}_i(s_{t-j}) = a', \pi_i(s_{t-j}) \neq a'] - \sum_{j=0}^{w-1} \mathbb{I}[\tilde{\pi}_i(s_{t-j}) \neq a', \pi_i(s_{t-j}) = a']\right) && \text{(G.80)} \\
&- \sum_{i\in B}\left(\sum_{j=0}^{w-1} \mathbb{I}[\tilde{\pi}_i(s_{t-j}) = a, \pi_i(s_{t-j}) \neq a] - \sum_{j=0}^{w-1} \mathbb{I}[\tilde{\pi}_i(s_{t-j}) \neq a, \pi_i(s_{t-j}) = a]\right) && \text{(G.81)} \\
\geqslant&\delta_{a',a} - \sum_{i\in B}\left(\sum_{j=0}^{w-1} \mathbb{I}[\tilde{\pi}_i(s_{t-j}) \neq a', \pi_i(s_{t-j}) = a'] + \sum_{j=0}^{w-1} \mathbb{I}[\tilde{\pi}_i(s_{t-j}) = a, \pi_i(s_{t-j}) \neq a]\right) && \text{(G.82)} \\
\geqslant&\delta_{a',a} - \sum_{i\in B}\left(\sum_{j=0}^{w-1} \mathbb{I}[\pi_i(s_{t-j}) = a'] + \sum_{j=0}^{w-1} \mathbb{I}[\pi_i(s_{t-j}) \neq a]\right) && \text{(G.83)} \\
=&\delta_{a',a} - \sum_{i\in B}\left(\sum_{j=0}^{w-1} \mathbb{I}_{i,a'}(s_{t-j}) + w - \sum_{j=0}^{w-1} \mathbb{I}_{i,a}(s_{t-j})\right) && \text{(G.84)} \\
=&\delta_{a',a} - \sum_{i\in B} h_{i,a',a} \overset{(a)}{\geqslant} \delta_{a',a} - \sum_{i=1}^{|B|} h_{a',a}^{(i)} \overset{(b)}{\geqslant} \delta_{a',a} - \sum_{i=1}^{K} h_{a',a}^{(i)} \overset{(c)}{\geqslant} 0. && \text{(G.85)}
\end{aligned}$$

This contradicts with Equation (G.78), and thus the assumption is falsified, i.e., there is no such poisoning attack that let $\widetilde{\pi_{\mathrm{T}}}(s_{t_0:t}) = a$. In the above equations, (a) is due to the fact that $h_{a',a}^{(i)}$ is a nonincreasing permutation of $\{h_{i,a',a}\}_{i=0}^{u-1}$. (b) is due to the facts that $|B| \leqslant K$

and $h^{(i)}_{a',a} \geqslant 0$. (c) comes from Equation (G.77). QED.

G.2.8 Hardness for Computing Tight Possible Action Set in TPARL

Proof of Theorem 9.6. By Definition 9.8, the possible action set with minimum cardinality (called minimal possible action set hereinafter) is unique. Otherwise, suppose A and B are both minimal possible action set, but $A \neq B$, then $A \cap B$ is a smaller and valid possible action set. Therefore, the oracle that returns the minimal possible action set, denoted by MINSET, can tell whether any action $a \in \mathsf{MINSET}$ and thus whether any action a can be chosen by some poisoned policy $\widetilde{\pi}$ whose poisoning size is within K. In other words, the problem of determining whether an action a can be chosen by some poisoned policy $\tilde{\pi}$ whose poisoning size is within K, denoted by $\mathsf{ATTKACT}$, is polynomially equivalent to MINSET: $\mathsf{MINSET} \equiv_P \mathsf{ATTKACT}$. Now, we show a polynomial reduction from the set cover problem to $\mathsf{ATTKACT}$, which implies that our MINSET problem is an NP-complete problem.

The decision version of the set cover problem [179], denoted by $\mathsf{SETCOVER}$, is a well-known NP-complete problem and is defined as follows. The inputs are

1. a universal set of n elements: $\mathcal{U} = \{u_1, u_2, \cdots, u_n\}$;
2. a set of subsets of $\mathcal{U}$: $\mathcal{V} = \{V_1, \cdots, V_m\}, V_i \subseteq \mathcal{U}, 1 \leqslant i \leqslant m, \bigcup_{i=1}^{m} V_i = \mathcal{U}$;
3. a positive number $K \in \mathbb{R}_+$.

The output is a boolean variable b, indicating that whether there exists a subset $\mathcal{W} \subseteq \mathcal{V}, |\mathcal{W}| \leqslant K$, such that $\forall u_i \in \mathcal{U}, \exists V \in \mathcal{W}, u_i \in V$. Given an oracle to $\mathsf{ATTKACT}$, we need to show $\mathsf{SETCOVER}$ can be solved in polynomial time, i.e., $\mathsf{SETCOVER} \leqslant_P \mathsf{ATTKACT}$.

- If $K \geqslant n$:
  We scan all sets $V_i \in \mathcal{V}$. To being with, we have a record set $S \leftarrow \varnothing$, and an answer set $\mathcal{W} \leftarrow \varnothing$. Whenever we encounter a set that contains a new element $u_i \notin S$, we put $\mathcal{W} \leftarrow \mathcal{W} \cup \{V_j\}$, and record this element $S \leftarrow S \cup \{u_i\}$. After one scan pass, $\mathcal{W}$ covers all elements of $\mathcal{U}$ (since $\bigcup_{j=1}^{m} V_j = \mathcal{U}$), and $|\mathcal{W}| \leqslant n \leqslant K$. Therefore, $\mathcal{W}$ is a valid set cover. Since we can always find such $\mathcal{W}$, we can directly answer `true`.
- If $K \geqslant m$:
  We can directly return $\mathcal{V}$ as a valid set cover, since $\bigcup_{j=1}^{m} V_j = \mathcal{U}$ and $|\mathcal{V}| \leqslant m \leqslant K$. Thus, we can answer `true`.

- If $K < \min\{n, m\}$:
  This is the general case which we need to handle. Now we construct the $\mathsf{ATTKACT}(K)$ problem so that we can trigger its oracle to solve $\mathsf{SETCOVER}$.

  1. The poisoning size is K.
  2. The action space $\mathcal{A} = \mathcal{U} \cup \{b\} \cup \Gamma$ where $\Gamma := \{\#_1, \ldots, \#_{m^2n}\}$. ($|\mathcal{A}| = n + 1 + m^2n$.) The sorting of actions is $u_1 < u_2 < \cdots < u_n < b < \#_1 < \cdots < \#_{m^2n}$.
  3. The subpolicies are $\{\pi_j^a\}_{j=1}^m \cup \{\pi_i^b, \pi_{i,j}^c \mid 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant K-1\}$, where π_j^a corresponds to $V_j \in \mathcal{V}$, and $\pi_i^b, \pi_{i,j}^c$ correspond to $u_i \in \mathcal{U}$. (Number of subpolicies $u = m + Kn \leqslant m + n^2$.)
  4. The current time step is $t = nm$, and the window size $W = nm$.
  5. The input action is b, i.e., asking whether b can be chosen by some poisoned TPARL policy, i.e., $\widetilde{\pi_{\mathrm{T}}}(s_{1:t}) = b$, if the poisoning size is within K.

  Now, we construct the states at each step t ($1 \leqslant t \leqslant nm$) so that the subpolicies' action predictions at these steps are as follows.

  1. Count the appearing time of each u_i in $\mathcal{V}$, and denote it by c_i: $c_i = \sum_{j=1}^m \mathbb{I}[u_i \in V_j]$. For each u_i, select a V_{j_0} that contains u_i, and in the corresponding $\pi_{j_0}^a$, assign $m - c_i + 1$ steps to predict u_i; for all other V_j that contains u_i, in the corresponding π_j^a, assign one step to predict u_i.
  2. After this process, each π_j^a at least has one time step whose action prediction is u_i for each $u_i \in V_j$. Among all $\{\pi_j^a\}_{j=1}^m$ and all time steps $1 \leqslant t \leqslant nm$, mn step-action cells are filled, and the remaining $(m^2n - mn)$ cells are filled by $\#_l \in \Gamma$ sequentially.
  3. For each π_i^b, arbitrarily select $W - m$ time steps to assign action prediction as u_i; and fill in other m time steps by remaining $\#_l \in \Gamma$ sequentially.
  4. For each $\pi_{i,j}^c$, for all time steps, let the action prediction be u_i.

  As we can observe, the number of actions, the number of subpolicies, and the window size are all bounded by a polynomial of n and m. Therefore, such construction can be done in polynomial time.

  We then show $\mathsf{SETCOVER} = \texttt{true} \iff \exists K', b \in \mathsf{ATTKACT}(K'), 1 \leqslant K' \leqslant K$.

  - $\Longrightarrow$:
    Suppose the covering set is $\mathcal{W} \subseteq \mathcal{V}$, we denote K' to $|\mathcal{W}|$, and construct the $\mathsf{ATTKACT}$ problem with poisoning size K' as described above.

We can construct a poisoning strategy to let $\widetilde{\pi_{\mathrm{T}}}(s_{1:nm}) = b$, The poisoning strategy is to find out π_j^a for each $V_j \in \mathcal{W}$, and to let them predict action b throughout all time steps: $\widetilde{\pi}_j^a(s'_t) = b, 1 \leqslant t' \leqslant nm$.

After poisoning, the aggregated action count $\widetilde{n}_b(s_{1:nm}) = |\mathcal{W}| \times nm = K'nm$. Since $\mathcal{W}$ covers every $u_i \in \mathcal{U}$, for each $u_i \in \mathcal{U}$ there exists a set $V_j \ni u_i$, whose corresponding $\widetilde{\pi}_j^a$ is poisoned to predict b. Thus, $\widetilde{n}_{u_i}(s_{1:nm}) < n_{u_i}(s_{1:nm}) = m + (W-m) + W \times (K'-1) = WK' = K'nm$. For any $\#_l \in \Gamma$, $\widetilde{n}_{\#_l}(s_{1:nm}) \leqslant n_{\#_l}(s_{1:nm}) = 1$. In summary,

$$\widetilde{n}_{u_i}(s_{1:nm}) < K'nm, \widetilde{n}_b(s_{1:nm}) = K'nm, \widetilde{n}_{\#_l}(s_{1:nm}) = 1. \tag{G.86}$$

Thus, after TPARL aggregation, the poisoned policy $\widetilde{\pi_{\mathrm{T}}}(s_{1:nm}) = b$, and therefore $b \in \mathsf{ATTKACT}(K')$.

– $\Longleftarrow$:

Suppose it is K' that let $b \in \mathsf{ATTKACT}(K')$, which implies that there exists such a poisoning attack within size K' that misleads the poisoned policy to b: $\widetilde{\pi_{\mathrm{T}}}(s_{1:nm}) = b$. Since the poisoning size is K', after poisoning the aggregated action count

$$\widetilde{n}_b(s_{1:nm}) \leqslant K'W = K'nm. \tag{G.87}$$

For each $u_i \in \mathcal{U}$, since $n_{u_i}(s_{1:nm}) = m + (W-m) + W \times (K'-1) = K'nm$, we always have

$$\widetilde{n}_{u_i}(s_{1:nm}) \overset{(*)}{<} \widetilde{n}_b(s_{1:nm}) \leqslant n_{u_i}(s_{1:nm}), \tag{G.88}$$

where $(*)$ is due to the condition of successful attack. We denote the set of poisoned subpolicies by Π ($|\Pi| \leqslant K'$). Therefore, Equation (G.88) implies that for each $u_i \in \mathcal{U}$, there exists at least one subpolicy

$$\pi'_{u_i} \in \Pi, \pi'_{u_i} \in \{\pi_j^a \mid u_i \in V_j\} \cup \{\pi_i^b\} \cup \{\pi_{i,j}^c \mid 1 \leqslant j \leqslant K-1\} \tag{G.89}$$

that is poisoned by the attack, otherwise the aggregated vote $\widetilde{n}_{u_i}$ cannot change.

We partition Γ by Γ^a and Γ^{bc}, where

$$\Gamma^a = \Gamma \cap \{\pi_j^a\}_{j=1}^m, \quad \Gamma^b = \Gamma \cap \{\pi_i^b, \pi_{i,j}^c \mid 1 \leqslant i \leqslant n, 1 \leqslant j \leqslant K-1\}. \tag{G.90}$$

We construct additional poisoning set Γ_+^a following this process: In the beginning, $\Gamma_+^a \leftarrow \varnothing$. For each $u_i \in \mathcal{U}$, if $\Gamma^a \cap \{\pi_j^a \mid u_i \in V_j\}$ is not empty, skip. Otherwise, according to Equation (G.89), $\Gamma^b \cap \{\pi_i^b, \pi_{i,j}^c \mid 1 \leqslant j \leqslant K-1\}$ is not empty. In this

case, we find an arbitrary covering set of u_i, namely $V_{j_0} \ni u_i$, and put $\pi^a_{j_0}$ into Γ^a_+. When the process terminates, we find that for each $u_i \in \mathcal{U}$,

$$(\Gamma^a \cup \Gamma^a_+) \cap \{\pi^a_j \mid u_i \in V_j\} \neq \varnothing. \tag{G.91}$$

Following the mapping $\{\pi^a_j\}_{j=1}^m \longleftrightarrow \{V_j \mid 1 \leqslant j \leqslant m\} = \mathcal{V}$, the subset $(\Gamma^a \cup \Gamma^a_+) \subseteq \{\pi^a_j\}_{j=1}^m$ can be mapped to $(\mathcal{W}^a \cup \mathcal{W}^a_+) \subseteq \mathcal{V}$. From Equation (G.91), $(\mathcal{W}^a \cup \mathcal{W}^a_+)$ is a valid set cover for $\mathcal{U}$. We now study the cardinality of this set cover. From the process, we know that every $V_{j_0} \in \mathcal{W}^a_+$ corresponds to a different set in Γ^b. Thus, $|\mathcal{W}^a_+| \leqslant |\Gamma^b|$, which implies that

$$|\mathcal{W}^a \cup \mathcal{W}^a_+| \leqslant |\mathcal{W}^a| + |\mathcal{W}^a_+| \leqslant |\Gamma^a| + |\Gamma^b| = |\Gamma| \leqslant K'. \tag{G.92}$$

To this point, we successfully construct a set cover within cardinally $K' \leqslant K$ that covers $\mathcal{U}$, so $\mathsf{SETCOVER} = \texttt{true}$.

Therefore, we can check whether $\mathsf{SETCOVER} = \texttt{true}$ by iterating K' from 1 to K ($< \min\{n, m\}$ iterations), constructing the $\mathsf{ATTKACT}(K')$ problem, and querying the oracle. The whole process can be done in polynomial time assumping $O(1)$ computation time of the $\mathsf{ATTKACT}(K')$ oracle.

To this point, we have shown $\mathsf{SETCOVER} \leqslant_P \mathsf{ATTKACT}$. On the other hand, an undeterminisitic Turing machine can try different poisoning strategies by branching on whether to poison current subpolicy and what actions to be assigned to each poisoned subpolicy. The decision of whether the poisoning is successful can be done in polynomial time and $b \in \mathsf{ATTKACT}(K)$ corresponds to the existence of successfully attacked branches. Thus, $\mathsf{ATTKACT} \in \mathsf{NP}$. Given that $\mathsf{SETCOVER}$ is an NP-complete problem, so does $\mathsf{ATTKACT}$ and MINSET (since $\mathsf{MINSET} \equiv_P \mathsf{ATTKACT}$). QED.

G.2.9 Possible Action Set in DPARL

Proof of Theorem 9.7. For ease of notation, we assume that $W_{\max} \leqslant t+1$, and otherwise we let $W_{\max} \leftarrow t+1$. We let $t_0 = \max\{t - W_{\max} + 1, 0\}$ be the start time step of the maximum possible window.

We only need to prove the contrary: for any $a \in \mathcal{A} \backslash A(K)$, within poisoning size K, the poisoned policy cannot choose a: $\widetilde{\pi_{\mathrm{D}}}(s_{t_0:t}) \neq a$. We prove by contradiction: we assume that there exists such a poisoning attack within poisoning size K that lets $\widetilde{\pi_{\mathrm{D}}}(s_{t_0:t}) = a$. From

the expression of $A(K)$ (Equation (9.14)), $a \neq a_t = \pi_{\mathrm{D}}(s_{t_0:t})$. Suppose the selected time window before the attack is W' (selected according to Equation (9.4) based on Δ_t^W and n_a), and the selected time window after the attack is $\widetilde{W}'$ (selected according to Equation (9.4) based on $\widetilde{\Delta}_t^W$ and $\widetilde{n}_a$).

- If $\widetilde{W}' = W'$:
  Suppose we use the TPARL aggregation policy with window size $W = W'$ instead of current DPARL aggregation policy, then we will have $\pi_{\mathrm{T}}(s_{t-W'+1:t}) = a_t$ and $\widetilde{\pi_{\mathrm{T}}}(s_{t-W'+1:t}) = a$. Thus, according to the definition of possible action set (Definition 9.8), $a \in A(K)$ where $A(K)$ is defined by Equation (9.13) in Theorem 9.5. This implies that $a \in A(K)$ where $A(K)$ is defined by Equation (9.14), which contradicts the assumption that $a \in \mathcal{A} \backslash A(K)$.

- If $\widetilde{W}' \neq W'$:
  According to the definition in Equation (9.14),

$$\min_{1 \leqslant W^* \leqslant W_{\max}, W^* \neq W', a'' \neq a_t} L_{a,a''}^{W^*,W'} > K. \tag{G.93}$$

  We define $a_t^{\#} = \arg\max_{a_0 \neq a_t} \widetilde{n}_{a_0}(s_{t-W'+1:t})$. Then, the above equation implies that

$$L_{a,a_t^{\#}}^{\widetilde{W}',W'} > K \tag{G.94}$$

  and thus

$$\sum_{i=1}^{L_{a,a_t^{\#}}^{\widetilde{W}',W'}} g^{(i)} + W'(n_a^{\widetilde{W}'} - n_{a^{\#}}^{\widetilde{W}'}) - \widetilde{W}'(n_{a_t}^{W'} - n_{a_t^{\#}}^{W'}) - \mathbb{I}[a > a_t] < 0. \tag{G.95}$$

  Since $g^{(i)} \geqslant 0$ by definition (9.11),

$$\sum_{i=1}^{K} g^{(i)} + W'(n_a^{\widetilde{W}'} - n_{a^{\#}}^{\widetilde{W}'}) - \widetilde{W}'(n_{a_t}^{W'} - n_{a_t^{\#}}^{W'}) - \mathbb{I}[a > a_t] < 0, \tag{G.96}$$

  where $a^{\#} = \arg\max_{a_0 \neq a', a_0 \in \mathcal{A}} n_{a_0}(s_{t-\widetilde{W}'+1:t})$ and n_a^w is a shorthand of $n_a(s_{t-w+1:t})$.
  Following the derivation from Equation (G.50) to (a), we have

$$\sum_{i=1}^{K} g^{(i)} + W'(n_a^{\widetilde{W}'} - n_{a^{\#}}^{\widetilde{W}'}) - \widetilde{W}'(n_{a_t}^{W'} - n_{a_t^{\#}}^{W'}) - \mathbb{I}[a > a_t] \geqslant W'(\widetilde{n}_a^{\widetilde{W}'} - \widetilde{n}_{a^{\#}}^{\widetilde{W}'}) - \widetilde{W}'(\widetilde{n}_{a_t}^{W'} - \widetilde{n}_{a_t^{\#}}^{W'}) - \mathbb{I}[a > a_t]. \tag{G.97}$$

  Combined with Equation (G.96),

$$W'(\widetilde{n}_a^{\widetilde{W}'} - \widetilde{n}_{a^{\#}}^{\widetilde{W}'}) - \widetilde{W}'(\widetilde{n}_{a_t}^{W'} - \widetilde{n}_{a_t^{\#}}^{W'}) - \mathbb{I}[a > a_t] < 0. \tag{G.98}$$

On the other hand, the successful attack assumption, i.e., $\widetilde{\pi_{\mathrm{D}}}(s_{t_0:t}) = a$ and $\pi_{\mathrm{D}}(s_{t_0:t}) = a_t$, implies that

$$\widetilde{\Delta}_t^{\widetilde{W}'} = \frac{\widetilde{n}_a^{\widetilde{W}'} - \widetilde{n}_{a^{(2)}}^{\widetilde{W}'}}{\widetilde{W}'} > \frac{\widetilde{n}_{a_{W'}}^{W'} - \widetilde{n}_{a_{W'}^{(2)}}^{W'}}{W'} = \widetilde{\Delta}_t^{W'}, \tag{G.99}$$

where ">" is "$\geqslant$" if $a < a_t$. In the above equation,

$$a^{(2)} = \underset{a_0 \neq a, a_0 \in \mathcal{A}}{\arg\max}\, \widetilde{n}_{a_0}^{\widetilde{W}'}, a_{W'} = \underset{a_0 \in \mathcal{A}}{\arg\max}\, \widetilde{n}_{a_0}^{W'}, a_{W'}^{(2)} = \underset{a_0 \neq a_{W'}, a_0 \in \mathcal{A}}{\arg\max}\, \widetilde{n}_{a_0}^{W'}. \tag{G.100}$$

Intuitively, after poisoning, $a^{(2)}$ is the runner-up action at window $\widetilde{W}'$, $a_{W'}$ is the action at window W', and $a_{W'}^{(2)}$ is the runner-up action at window W'. We rewrite Equation (G.99) to

$$W'(\widetilde{n}_a^{\widetilde{W}'} - \widetilde{n}_{a^{(2)}}^{\widetilde{W}'}) \geqslant \widetilde{W}'(\widetilde{n}_{a_{W'}}^{W'} - \widetilde{n}_{a_{W'}^{(2)}}^{W'}) + \mathbb{I}[a > a_t]. \tag{G.101}$$

We have the following two observations:

1. $\widetilde{n}_a^{\widetilde{W}'} - \widetilde{n}_{a^{\#}}^{\widetilde{W}'} \geqslant \widetilde{n}_a^{\widetilde{W}'} - \widetilde{n}_{a^{(2)}}^{\widetilde{W}'}$, since a is the top action and $a^{(2)}$ is the runner-up action and their margin should be the smallest.
2. $\widetilde{n}_{a_{W'}}^{W'} - \widetilde{n}_{a_{W'}^{(2)}}^{W'} \geqslant \widetilde{n}_{a_t}^{W'} - \widetilde{n}_{a_t^{\#}}^{W'}$, because 1) if $a_t = a_{W'}$, LHS equals to RHS; 2) if $a_t \neq a_{W'}$, LHS $\geqslant 0$ and RHS $\leqslant 0$.

Plugging these two observations to two sides of Equation (G.101), we get

$$W'(\widetilde{n}_a^{\widetilde{W}'} - \widetilde{n}_{a^{\#}}^{\widetilde{W}'}) \geqslant \widetilde{W}'(\widetilde{n}_{a_t}^{W'} - \widetilde{n}_{a_t^{\#}}^{W'}) + \mathbb{I}[a > a_t]. \tag{G.102}$$

This contradicts with Equation (G.98).

Since in both cases, we find contradictions. Now we can conclude that for any $a \in \mathcal{A} \backslash A(K)$, within poisoning size K, the poisoned policy cannot choose a: $\widetilde{\pi_{\mathrm{D}}}(s_{t_0:t}) \neq a$. QED.

G.3 EXPERIMENTAL DETAILS

G.3.1 Details of the Offline RL algorithms and Implementations

We experimented with three offline RL algorithms: DQN [265], QR-DQN [88], and C51 [31]. The first one is the standard baseline, while the latter two are distributional RL algorithms which show SOTA results in offline RL tasks. We first briefly introduce the algorithm ideas, followed by the implementation details.

Algorithm Ideas. The core of Q-learning [398] is the Bellman optimality equation [32]

$$Q^\star(s,a) = \mathbb{E}R(s,a) + \gamma \mathbb{E}_{s' \sim P} \max_{a' \in \mathcal{A}} Q^\star\left(s', a'\right), \tag{G.103}$$

where a parameterized Q^θ is adopted to approximate the optimal $Q^\star$ and iteratively improved. In DQN [265] specifically, the parameterization is achieved by using a convolutional neural network [202]. In contrast to estimating the mean action value $Q^\pi(s,a)$ in DQN, distributional RL algorithms estimate a density over the values of the state-action pairs. The distributional Bellman optimality can be expressed as follows:

$$Z^\star(s,a) \stackrel{D}{=} r + \gamma Z^\star\left(s', \text{argmax}_{a' \in \mathcal{A}} Q^\star\left(s', a'\right)\right) \text{ where } r \sim R(s,a), s' \sim P(\cdot \mid s, a). \tag{G.104}$$

Concretely, QR-DQN [88] approximates the density $D^\star$ with a uniform mixture of K Dirac delta functions, while C51 [31] approximates the density using a categorical distribution over a set of anchor points.

Implementation Details. For training the subpolicies using offline RL training algorithms, we use the code base of Agarwal et al. [3]. The configuration files containing the detailed hyperparameters can be found at their public repository `https://github.com/google-research/batch_rl/tree/master/batch_rl/fixed_replay/configs`. For the three methods DQN, QR-DQN, and C51, the names of the configuration files are `dqn.gin`, `quantile.gin`, and `c51.gin`, respectively. On each partition, we train the subpolicy for 50 epochs.

G.3.2 Concrete Experimental Procedures

We conduct experiments on two Atari 2600 games, Freeway and Breakout from OpenAI Gym [43], and one autonomous driving environment Highway [210], following Section 9.3.

Concretely, in the **training** stage, we first partition the training dataset into u partitions ($u = 30$, 50, or 100) by using the hash function $h(\tau)$ pre-defined in each environment. Let s^i be the i-th value in the state representation and d be the dimensionality of the state. In *Atari games* where the state is the game frame, $h(\tau)$ is defined as the sum of all pixel values of all frames in the trajectory τ, i.e., $h(\tau) = \sum_{s \in \tau} \sum_{i \in [d]} s^i$. In *Highway* where the state is a floating point number scalar containing the positions and velocities of all vehicles, we define $h(\tau) = \sum_{s \in \tau} \sum_{i \in [d]} f(s^i)$ where $f : \mathbb{R} \to \mathbb{Z}$ is a deterministic function that maps the given float value to integer space. Concretely, we take $f(x)$ as the sum of the higher

16 bits and the lower 16 bits of its 32-bit representation under IEEE 754 standard [135]. We then train the subpolicies on the partitions with offline RL training algorithm $\mathcal{M}_0 \in$ {DQN [265], QR-DQN [88], C51 [31]}. The detailed description of the algorithms can be found in Appendix G.3.1.

In the **aggregation** stage, we apply the three proposed aggregation protocols (PARL (Theorem 9.1), TPARL (Theorem 9.2), and DPARL (Theorem 9.3)) on the trained u subpolicies and derive the aggregated policies π_{P}, π_{T}, and π_{D} accordingly.

Finally, in the **certification** stage, for each aggregated policy, we provide the per-state action and cumulative reward certification following our theorems. When certifying the *per-state action*, we constrain the maximum trajectory length $H = 1000$ for Atari games and $H = 30$ for Highway (which is the full length in Highway) and report results averaged over 20 runs; for the *cumulative reward* certification, we adopt the trajectory length $H = 400$ for evaluating Freeway, $H = 75$ for Breakout, and $H = 30$ for Highway.

Configuration of Trajectory Length H. For *Atari games*, we do not evaluate the full episode length (up to tens of thousands steps), since our goal is to compare the relative certified robustness of different RL algorithms, and the evaluation on relatively short trajectory is sufficient under affordable computation cost. Moreover, different episodes in Atari games are oftentimes of different lengths; thus it is necessary that we restrict the episode length to enable a fair comparison. For *Highway*, we evaluate on the full episode ($H = 30$) where we can efficiently achieve effective comparisons.

APPENDIX H: APPENDIX FOR CHAPTER 11

H.1 PROOFS OF MAIN RESULTS

This appendix entails the complete proofs for Proposition 11.1, Theorem 11.1, Theorem 11.2, Lemma 11.1, and Theorem 11.3 in the main text. For complex proofs such as that for Theorem 11.3, we also provide high-level illustrations before delving into the formal proof.

H.1.1 Proof of Proposition 11.1

Proof of Proposition 11.1. Since each term $\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y|Y=y, X_s=i]$ is within $[0,\epsilon]$, we consider two cases: $y \neq 1$ and $y=1$. If $y \neq 1$, $\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y=y, X_s=i] \leqslant \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y|Y=y, X_s=i] \leqslant \epsilon$ and so will be their differences for $X_s=i$ and $X_s=j$. If $y=1$, $\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y=y, X_s=i] = 1 - \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) \neq Y|Y=y, X_s=i] \in [1-\epsilon, 1]$, and also the differences for $X_s=i$ and $X_s=j$ are always within ϵ. This proves ϵ-EO.

Now consider DP. We notice that for any a,

$$\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|X_s=a] = \sum_{y=1}^{C} \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y=y, X_s=a] \cdot \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[Y=y|X_s=a]. \tag{H.1}$$

Thus,

$$\begin{aligned}
&\Big| \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|X_s=i] - \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|X_s=j] \Big| && \text{(H.2)}\\
&\overset{(*)}{\leqslant} \sum_{y=1}^{C} \Big| \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y=y, X_s=i] - \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[h_{\boldsymbol{\theta}}(\boldsymbol{X}) = 1|Y=y, X_s=j] \Big| \\
&\qquad\qquad \cdot \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[Y=y|X_s=i] && \text{(H.3)}\\
&\leqslant \sum_{y=1}^{C} \epsilon \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[Y=y|X_s=i] = \epsilon && \text{(H.4)}
\end{aligned}$$

which proves ϵ-DP, where $(*)$ leverages the fair base rate property of $\mathcal{Q}$ which gives $\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[Y=y|X_s=i] = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[Y=y|X_s=j]$.

QED.

H.1.2 Proof of Theorem 11.1

Proof of Theorem 11.1. We first prove the key eq. (11.6).

$$
\begin{aligned}
H(\mathcal{P},\mathcal{Q}) \leqslant \rho &\iff H^2(\mathcal{P},\mathcal{Q}) \leqslant \rho^2 && \text{(H.5)}\\
&\iff \frac{1}{2}\int_{\mathcal{Z}} \left(\sqrt{p(\boldsymbol{z})}-\sqrt{q(\boldsymbol{z})}\right)^2 \mathrm{d}\mu(\boldsymbol{z}) \leqslant \rho^2 && \text{(H.6)}\\
&\iff \frac{1}{2}\left(\int_{\mathcal{Z}} p(\boldsymbol{z})\mathrm{d}\mu(\boldsymbol{z}) + \int_{\mathcal{Z}} q(\boldsymbol{z})\mathrm{d}\mu(\boldsymbol{z})\right) - \int_{\mathcal{Z}} \sqrt{p(\boldsymbol{z})q(\boldsymbol{z})}\mathrm{d}\mu(\boldsymbol{z}) \leqslant \rho^2 && \text{(H.7)}\\
&\iff \int_{\mathcal{Z}} \sqrt{p(\boldsymbol{z})q(\boldsymbol{z})}\mathrm{d}\mu(\boldsymbol{z}) \geqslant 1-\rho^2 && \text{(H.8)}\\
&\iff \sum_{i=1}^{N}\int_{\mathcal{Z}_i} \sqrt{p_i q_i}\cdot\sqrt{p_i(\boldsymbol{z})q_i(\boldsymbol{z})}\mathrm{d}\mu(\boldsymbol{z}) \geqslant 1-\rho^2 && \text{(H.9)}\\
&\iff \sum_{i=1}^{N}\sqrt{p_i q_i}\left(1-H^2(\mathcal{P}_i,\mathcal{Q}_i)\right) \geqslant 1-\rho^2 && \text{(H.10)}
\end{aligned}
$$

where $p_i(\cdot)$ and $q_i(\cdot)$ are density functions of subpopulation distributions $\mathcal{P}_i$ and $\mathcal{Q}_i$ respectively.

Then, we show that any feasible solution of Equation (11.5) satisfies the constraints in Equation (11.7). We let $\mathcal{Q}^\star$ and $\boldsymbol{\theta}^\star$ denote a feasible solution of Equation (11.5), i.e.,

$$
H(\mathcal{P},\mathcal{Q}^\star) \leqslant \rho, \quad e_j(\mathcal{P}, h_{\boldsymbol{\theta}^\star}) \leqslant v_j \,\forall j \in [L], \quad g_j(\mathcal{Q}^\star) \leqslant u_j \,\forall j \in [M]. \tag{H.11}
$$

We let $\{q_i^\star\}_{i=1}^N$ denote the proportions of $\mathcal{Q}^\star$ within each support partition $\mathcal{Z}_i$, and $\{\mathcal{Q}_i^\star\}_{i=1}^N$ the $\mathcal{Q}^\star$ in each subpopulation. By Equation (H.10), we have $1-\rho^2-\sum_{i=1}^N \sqrt{p_i q_i^\star}(1-\rho_i^2) \leqslant 0$ where $\rho_i = H^2(\mathcal{P}_i,\mathcal{Q}_i^\star)$. Note that by definition, $\sum_{i=1}^N q_i^\star = 1$ and $\forall i \in [N], q_i^\star \geqslant 0, \rho_i \geqslant 0$. Furthermore, by the implication relations stated in Theorem 11.1, for any $j \in [L]$, $e'_j(\{\mathcal{P}_i\}_{i=1}^N, \{p_i\}_{i=1}^N, h_{\boldsymbol{\theta}^\star}) \leqslant v'_j$; and for any $j \in [M]$, $g'_j(\{\mathcal{Q}_i^\star\}_{i=1}^N, \{q_i^\star\}_{i=1}^N) \leqslant u'_j$. To this point, we have shown $\mathcal{Q}^\star$ and $\boldsymbol{\theta}^\star$ satisfy all constraints in Equation (11.7), i.e., $\mathcal{Q}^\star$ and $\boldsymbol{\theta}^\star$ is a feasible solution of Equation (11.7). Since Equation (11.7) expresses the optimal (maximum) solution, Equation (11.7) (in Theorem 11.1) $\geqslant$ Equation (11.5). QED.

H.1.3 Proof of Theorem 11.2

Proof of Theorem 11.2. The proof of Theorem 11.2 is composed of three parts: (1) the optimization problem provides a fairness certificate for Problem 2; (2) the certificate is tight; and (3) the optimization problem is convex.

(1) Suppose the maximum of Problem 2 is attained with the test distribution $\mathcal{Q}^\star$ in the sensitive shifting setting, then we decompose both $\mathcal{P}$ and $\mathcal{Q}^\star$ according to both the sensitive attribute and the label:

$$\mathcal{P} = \sum_{s=1}^{S}\sum_{y=1}^{C} p_{s,y}\mathcal{P}_{s,y}, \quad \mathcal{Q}^\star = \sum_{s=1}^{S}\sum_{y=1}^{C} q^\star_{s,y}\mathcal{Q}^\star_{s,y}. \tag{H.12}$$

Since $\mathcal{Q}^\star$ is a fair base rate distribution, for any $i, j \in [S]$, $b^{\mathcal{Q}^\star}_{i,y} = b^{\mathcal{Q}^\star}_{j,y}$ where $b^{\mathcal{Q}^\star}_{s,y} = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[Y = y|X_s = s]$. As a result, $\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[Y = y|X_s = s] = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[Y = y]$. Now we define

$$k^\star_s := \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[X_s = s], \quad r^\star_y := \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[Y = y], \tag{H.13}$$

and then

$$q^\star_{s,y} = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[X_s = s, Y = y] = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[X_s = s] \cdot \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[Y = y|X_s = s] = k^\star_s r^\star_y. \tag{H.14}$$

By the distance constraint in Problem 2 (namely $H(\mathcal{P}, \mathcal{Q}^\star) \leqslant \rho$) and Equation (H.10), we have

$$\sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y}q^\star_{s,y}}\left(1 - H^2(\mathcal{P}_{s,y}, \mathcal{Q}^\star_{s,y})\right) \geqslant 1 - \rho^2. \tag{H.15}$$

Since there is only sensitive shifting, $H^2(\mathcal{P}_{s,y}, \mathcal{Q}^\star_{s,y}) = 0$, given Equation (H.14), we have

$$\sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y}k^\star_s r^\star_y} \geqslant 1 - \rho^2. \tag{H.16}$$

Now, we can observe that the $k^\star_s$ and $r^\star_y$ induced by $\mathcal{Q}^\star$ satisfy all constraints of Problem 2. For the objective,

Objective in Theorem 11.2

$$= \sum_{s=1}^{S}\sum_{y=1}^{C} k^\star_s r^\star_s \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \tag{H.17}$$

$$= \sum_{s=1}^{S}\sum_{y=1}^{C} q^\star_{s,y} \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \quad \text{(by Equation (H.14) and } H^2(\mathcal{P}_{s,y}, \mathcal{Q}^\star_{s,y}) = 0\text{)} \tag{H.18}$$

$$= \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \tag{H.19}$$

=Optimal value of Problem 2.

Therefore, the *optimal* value of Theorem 11.2 will be larger or equal to the optimal value of Problem 2 which concludes the proof of the first part.

(2) Suppose the optimal value of Theorem 11.2 is attained with $k_s^\star$ and $r_y^\star$. We then construct $\mathcal{Q}^\star = \sum_{s=1}^S \sum_{y=1}^C k_s^\star r_y^\star \mathcal{P}_{s,y}$. We now inspect each constraint of Problem 2. The constraint $\text{dist}(\mathcal{P}, \mathcal{Q}^\star) \leqslant \rho$ is satisfied because $1 - \rho^2 - \sum_{s=1}^S \sum_{y=1}^C \sqrt{p_{s,y} k_s^\star r_y^\star} \leqslant 0$ is satisfied as a constraint of Theorem 11.2. Apparently, $\mathcal{P}_{s,y} = \mathcal{Q}_{s,y}^\star$. Then, $\mathcal{Q}^\star$ is a fair base rate distribution because

$$b_{s,y}^{\mathcal{Q}^\star} = \Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[Y = y | X_s = s] = \frac{k_s^\star r_y^\star}{k_s^\star} = r_y^\star \tag{H.20}$$

is a constant across all $s \in [S]$. Thus, $\mathcal{Q}^\star$ satisfies all constraints of Problem 2 and

$$\begin{aligned} &\text{Optimal objective of Problem 2} \\ \geqslant &\mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \\ = &\sum_{s=1}^S \sum_{y=1}^C k_s^\star r_y^\star \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \\ = &\sum_{s=1}^S \sum_{y=1}^C k_s^\star r_y^\star E_{s,y} = \text{Optimal objective of Theorem 11.2.} \end{aligned} \tag{H.21}$$

Combining with the conclusion of the first part, we know optimal values of Theorem 11.2 and Problem 2 match, i.e., the certificate is tight.

(3) Inspecting the problem definition in Theorem 11.2, we find the objective and all constraints but the last one are linear. Therefore, to prove the convexity of the optimization problem, we only need to show that the last constraint

$$1 - \rho^2 - \sum_{s=1}^S \sum_{y=1}^C \sqrt{p_{s,y} k_s r_y} \leqslant 0 \tag{H.22}$$

is convex with respect to k_s and r_y. Given two arbitrary feasible pairs of k_s and r_y satisfying Equation (H.22), namely (k_s^a, r_y^a) and (k_s^b, r_y^b), we only need to show that (k_s^m, r_y^m) also satisfies Equation (H.22), where $k_s^m = (k_s^a + k_s^b)/2$, $r_y^m = (r_y^a + r_y^b)/2$.

Indeed,

$$1-\rho^2-\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}k_s^m r_y^m} \tag{H.23}$$

$$=1-\rho^2-\frac{1}{2}\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}}\cdot\sqrt{k_s^a+k_s^b}\cdot\sqrt{r_y^a+r_y^b} \tag{H.24}$$

$$\leqslant 1-\rho^2-\frac{1}{2}\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}}\cdot\left(\sqrt{k_s^a r_y^a}+\sqrt{k_s^b r_y^b}\right) \qquad \text{(Cauchy's inequality)} \tag{H.25}$$

$$=\frac{1}{2}\left(1-\rho^2\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}k_s^a r_y^a}\right)+\frac{1}{2}\left(1-\rho^2\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}k_s^b r_y^b}\right) \tag{H.26}$$

$$\leqslant 0.$$

QED.

H.1.4 Proof of Lemma 11.1

Proof of Lemma 11.1. The proof of Lemma 11.1 is composed of two parts: (1) the optimization problem provides a fairness certificate for Problem 1; and (2) the certificate is tight. The high-level proof sketch is similar to the proof of Theorem 11.2.

(1) Suppose that the maximum of Problem 1 is attained with the test distribution $\mathcal{Q}^\star$ under the general shifting setting, then we decompose both $\mathcal{P}$ and $\mathcal{Q}^\star$ according to both the sensitive attribute and the label:

$$\mathcal{P}=\sum_{s=1}^{S}\sum_{y=1}^{C}p_{s,y}\mathcal{P}_{s,y},\quad \mathcal{Q}^\star=\sum_{s=1}^{S}\sum_{y=1}^{C}q^\star_{s,y}\mathcal{Q}^\star_{s,y}. \tag{H.27}$$

Unlike sensitive shifting setting, in general shifting setting, here the subpopulation of $\mathcal{Q}^\star$ is $\mathcal{Q}^\star_{s,y}$ instead of $\mathcal{P}_{s,y}$ due to the existence of distribution shifting within each subpopulation.

Following the same argument as in the first part proof of Theorem 11.2, since $\mathcal{Q}^\star$ is a fair base rate distribution, we can define

$$k_s^\star:=\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[X_s=s],\quad r_y^\star:=\Pr_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[Y=y], \tag{H.28}$$

and write

$$\mathcal{Q}^\star := \sum_{s=1}^{S}\sum_{y=1}^{C} k_s^\star r_y^\star \mathcal{Q}_{s,y}^\star \tag{H.29}$$

since $q_{s,y}^\star = k_s^\star r_y^\star$. We also define $\rho_{s,y}^\star = H(\mathcal{P}_{s,y}, \mathcal{Q}_{s,y}^\star)$. Now we show these $k_s^\star, r_y^\star, \mathcal{Q}_{s,y}^\star, \rho_{s,y}^\star$ along with model parameter $\boldsymbol{\theta}$ constitute a feasible point of Equation (11.11), and the objectives of Equation (11.11) and Problem 2 are the same given $\mathcal{Q}^\star$.

- (Feasibility)
  There are three constraints in Equation (11.11). By the definition of $k_s^\star$ and $r_y^\star$, naturally Equation (11.11b) is satisfied. Then, according to Equation (H.10) and the definifition of $\rho_{s,y}^\star$ above, Equation (11.11c) and Equation (11.11d) are satisfied.
- (Objective Equality)

$$\begin{aligned}
\text{Equation (11.11a)} &= \sum_{s=1}^{S}\sum_{y=1}^{C} k_s^\star r_y^\star \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}^\star}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \\
&= \sum_{s=1}^{S}\sum_{y=1}^{C} q_{s,y}^\star \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}^\star}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \\
&= \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\star}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] = \text{Optimal value of Problem 1.}
\end{aligned} \tag{H.30}$$

  As a result, the optimal value of Equation (11.11) is larger than or equal to the optimal value of Problem 1, and hence the optimization problem encoded by Equation (11.11) provides a fairness certificate.

(2) To prove the tightness of the certificate, we only need to show that the optimal value of the optimization problem in Equation (11.11) is also attainable by the original Problem 1.

Suppose that the optimal objective of Equation (11.11) is achieved by optimizable parameters $k_s^\star, r_y^\star, \mathcal{Q}^\star$, and $\rho_{s,y}^\star$. Then, we construct $\mathcal{Q}^\dagger = \sum_{s=1}^{S}\sum_{y=1}^{C} k_s^\star r_y^\star \mathcal{Q}_{s,y}^\star$. We first show that $\mathcal{Q}^\dagger$ is a feasible point of Problem 1, and then show that the objective given $\mathcal{Q}^\dagger$ is equal to the optimal objective of Equation (11.11).

- (Feasibility)
  There are two constraints in Problem 1: the bounded distance constraint and the fair base rate constraint. The bounded distance constraint is satisfied due to applying Equation (H.10) along with Equations (11.11c) and (11.11d). The fair base rate constraint is satisfied following the same deduction as in Equation (H.20).

- (Objective Equality)

$$\begin{aligned}\text{Objective Problem 1} &= \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}^\dagger}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] = \sum_{s=1}^{S}\sum_{y=1}^{C} k_s^\star r_y^\star \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}^\star}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] \\ &= \text{Optimal value of Equation (11.11).}\end{aligned} \tag{H.31}$$

Thus, the optimal value of the optimization problem in Equation (11.11) is attainable also by the original Problem 1 which concludes the tightness proof.

QED.

H.1.5 Proof of Theorem 11.3

High-level Illustration. The starting point of our proof is Lemma 11.1, where we have shown a fairness certificate for Problem 1 (general shifting setting). Then, we plug in Theorem 2.2 in [399] (stated as Theorem H.1 below) to upper bound the expected loss within each sub-population. Now, we get an optimization problem involving k_s, r_y, and $\rho_{s,y}$ that upper bounds the optimization problem in Lemma 11.1. In this optimization problem, we find k_s and r_y are bounded in $[0,1]$, and once these two variables are fixed, the optimization with respect to $x_{s,y} := (1-\rho_{s,y}^2)^2$ becomes convex. Using this observation, we propose to partition the feasible space of k_s and r_y into sub-regions and solve the convex optimization within each region bearing some degree relaxation, which yields Theorem 11.3.

Theorem 2.2 in [399]. As mentioned in Section 11.3.3, we leverage Theorem 2.2 from [399] to upper bound the expected loss of $h_{\boldsymbol{\theta}}(\cdot)$ in each shifted subpopulation $\mathcal{Q}_{s,y}$. Here we restate Theorem 2.2 for completeness.

Theorem H.1 (Theorem 2.2, [399])**.** Let $\mathcal{P}'$ and $\mathcal{Q}'$ denote two distributions supported on $\mathcal{X}\times\mathcal{Y}$, suppose that $0 \leqslant \ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y) \leqslant M$, then

$$\begin{aligned}&\max_{\mathcal{Q}',\boldsymbol{\theta}} \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] \quad \text{s.t.} \quad H(\mathcal{P}',\mathcal{Q}') \leqslant \rho \\ \leqslant &\mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] + 2C_\rho\sqrt{\mathbb{V}_{(\boldsymbol{X},Y)\sim\mathcal{P}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)]} + \\ &\rho^2(2-\rho^2)\left(M - \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] - \frac{\mathbb{V}_{(\boldsymbol{X},Y)\sim\mathcal{P}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)]}{M - \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{P}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)]}\right),\end{aligned} \tag{H.32}$$

where $C_\rho = \sqrt{\rho^2(1-\rho^2)^2(2-\rho^2)}$, for any given distance bound $\rho > 0$ that satisfies

$$\rho^2 \leqslant 1 - \left(1 + \frac{(M - \mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{P}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)])^2}{\mathbb{V}_{(\boldsymbol{X},Y)\sim\mathcal{P}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)]}\right)^{-1/2}. \tag{H.33}$$

This theorem provides a closed-form expression that upper bounds the mean loss of $h_{\boldsymbol{\theta}}(\cdot)$ on shifted distribution (namely $\mathbb{E}_{\mathcal{Q}'}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)]$), given bounded Hellinger distance $H(\mathcal{P}, \mathcal{Q})$ and the mean E and variance V of loss on $\mathcal{P}$ under two mild conditions: (1) the function is positive and bounded (denote the upper bound by M); and (2) the distance $H(\mathcal{P}, \mathcal{Q})$ is not too large (specifically, $H(\mathcal{P}, \mathcal{Q})^2 \leqslant \bar{\gamma}^2 := 1 - (1 + (M - E)^2/V)^{-\frac{1}{2}}$). Since Theorem H.1 holds for arbitrary models and loss functions $\ell(h_{\boldsymbol{\theta}}(\cdot), \cdot)$ as long as the function value is bounded by $[0, M]$, using Theorem H.1 allows us to provide a generic and succinct fairness certificate in Theorem 11.3 for general shifting case that holds for generic models including DNNs without engaging complex model architectures. Indeed, we only need to query the mean and variance under $\mathcal{P}$ for the given model to compute the certificate in Theorem H.1, and this benefit is also inherited by our certification framework expressed by Theorem 11.3. Note that there is no tightness guarantee for this bound yet, which is also inherited by our Theorem 11.3.

Proof of Theorem 11.3. The proof is done stage-wise: starting from Lemma 11.1, we apply relaxation and derive a subsequent optimization problem that upper bounds the previous one stage by stage, until we get the final expression in Theorem 11.3.

To demonstrate the proof, we first define the optimization problems at each stage, then prove the relaxations between each adjacent stage, and finally show that the last optimization problem contains a finite number of $\mathbf{C}$'s values where each $\mathbf{C}$ is a convex optimization, so that the final optimization problem provides a computable fairness certificate.

We define these quantities, for $s \in [S], y \in [C]$:

$$\begin{aligned}
&E_{s,y} = \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)], \quad V_{s,y} = \mathbb{V}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)], \\
&p_{s,y} = \mathop{\Pr}_{(\boldsymbol{X},Y)\sim\mathcal{P}}[X_s = s, Y = y], \quad C_{s,y} = M - E_{s,y} - \frac{V_{s,y}}{M - E_{s,y}}, \\
&\bar{\gamma}_{s,y}^2 = 1 - (1 + (M - E_{s,y})^2/V_{s,y})^{-\frac{1}{2}}.
\end{aligned} \tag{H.34}$$

Given $\rho > 0$ and the above quantities, the optimization problem definitions are:

- Lemma 11.1:

$$\max_{k_s, r_y, \mathcal{Q}, \rho_{s,y}} \quad \sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}} [\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y)] \tag{H.35a}$$

$$\text{s.t.} \quad \sum_{s=1}^{S} k_s = 1, \quad \sum_{y=1}^{C} r_y = 1, \quad k_s \geqslant 0 \quad \forall s \in [S], \quad r_y \geqslant 0 \quad \forall y \in [C], \tag{H.35b}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y}(1-\rho_{s,y}^2) \geqslant 1-\rho^2 \tag{H.35c}$$

$$H(\mathcal{P}_{s,y}, \mathcal{Q}_{s,y}) \leqslant \rho_{s,y} \quad \forall s \in [S], y \in [C]. \tag{H.35d}$$

- After applying Theorem H.1:

$$\max_{k_s, r_y, \rho_{s,y}} \quad \sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y \left(E_{s,y} + 2\sqrt{\rho_{s,y}^2(1-\rho_{s,y}^2)^2(2-\rho_{s,y}^2)}\sqrt{V_{s,y}} + \rho_{s,y}^2(2-\rho_{s,y}^2) C_{s,y} \right) \tag{H.36a}$$

$$\text{s.t.} \quad \sum_{s=1}^{S} k_s = 1, \quad \sum_{y=1}^{C} r_y = 1, \quad k_s \geqslant 0 \quad \forall s \in [S], \quad r_y \geqslant 0 \quad \forall y \in [C], \tag{H.36b}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y}(1-\rho_{s,y}^2) \geqslant 1-\rho^2, \tag{H.36c}$$

$$0 \leqslant \rho_{s,y} \leqslant \bar{\gamma}_{s,y}. \tag{H.36d}$$

- After variable transform $x_{s,y} := (1-\rho_{s,y}^2)^2$:

$$\max_{k_s, r_y, x_{s,y}} \quad \sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y \left(E_{s,y} + 2\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{V_{s,y}} + (1-x_{s,y}) C_{s,y} \right) \tag{H.37a}$$

$$\text{s.t.} \quad \sum_{s=1}^{S} k_s = 1, \quad \sum_{y=1}^{C} r_y = 1, \quad k_s \geqslant 0 \quad \forall s \in [S], \quad r_y \geqslant 0 \quad \forall y \in [C], \tag{H.37b}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y x_{s,y}} \geqslant 1-\rho^2, \tag{H.37c}$$

$$(1-\bar{\gamma}_{s,y}^2)^2 \leqslant x_{s,y} \leqslant 1 \quad \forall s \in [S], y \in [C]. \tag{H.37d}$$

- After feasible region partitioning on k_s and r_y:

$$\max_{\{i_s\in[T]:s\in[S]\},\{j_y\in[T]:y\in[C]\}} \mathsf{C}'\left(\left\{\left[\frac{i_s-1}{T},\frac{i_s}{T}\right]\right\}_{s=1}^{S},\left\{\left[\frac{j_y-1}{T},\frac{j_y}{T}\right]\right\}_{y=1}^{C}\right),\ \text{where} \tag{H.38a}$$

$$\mathsf{C}'\left(\{[\underline{k_s},\overline{k_s}]\}_{s=1}^{S},\{[\underline{r_y},\overline{r_y}]\}_{y=1}^{C}\right)= \tag{H.38b}$$

$$\max_{\underline{k_s}\leqslant k_s\leqslant\overline{k_s},\underline{r_y}\leqslant r_y\leqslant\overline{r_y},x_{s,y}} \sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y\left(E_{s,y}+2\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{V_{s,y}}+(1-x_{s,y})C_{s,y}\right)$$

$$\text{s.t.}\quad \sum_{s=1}^{S}k_s=1,\quad \sum_{y=1}^{C}r_y=1, \tag{H.38c}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}k_s r_y x_{s,y}}\geqslant 1-\rho^2, \tag{H.38d}$$

$$(1-\bar{\gamma}_{s,y}^2)^2\leqslant x_{s,y}\leqslant 1\quad \forall s\in[S],y\in[C]. \tag{H.38e}$$

- Final quantity in Theorem 11.3:

$$\max_{\{i_s\in[T]:s\in[S]\},\{j_y\in[T]:y\in[C]\}} \mathsf{C}\left(\left\{\left[\frac{i_s-1}{T},\frac{i_s}{T}\right]\right\}_{s=1}^{S},\left\{\left[\frac{j_y-1}{T},\frac{j_y}{T}\right]\right\}_{y=1}^{C}\right),\ \text{where} \tag{H.39a}$$

$$\mathsf{C}\left(\{[\underline{k_s},\overline{k_s}]\}_{s=1}^{S},\{[\underline{r_y},\overline{r_y}]\}_{y=1}^{C}\right)=\max_{x_{s,y}}\sum_{s=1}^{S}\sum_{y=1}^{C}\Big(\overline{k_s}\,\overline{r_y}\left[E_{s,y}+C_{s,y}\right]_+ + \tag{H.39b}$$

$$\underline{k_s}\,\underline{r_y}\left[E_{s,y}+C_{s,y}\right]_- +2\overline{k_s}\,\overline{r_y}\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{V_{s,y}}-\underline{k_s}\,\underline{r_y}x_{s,y}[C_{s,y}]_+-\overline{k_s}\,\overline{r_y}x_{s,y}[C_{s,y}]_-\Big)$$

$$\text{s.t.}\quad \sum_{s=1}^{S}\underline{k_s}\leqslant 1,\quad \sum_{s=1}^{S}\overline{k_s}\geqslant 1,\quad \sum_{y=1}^{C}\underline{r_y}\leqslant 1,\quad \sum_{y=1}^{C}\overline{r_y}\geqslant 1, \tag{H.39c}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}\overline{k_s}\,\overline{r_y}x_{s,y}}\geqslant 1-\rho^2, \tag{H.39d}$$

$$(1-\bar{\gamma}_{s,y}^2)^2\leqslant x_{s,y}\leqslant 1\quad \forall s\in[S],y\in[C]. \tag{H.39e}$$

We have this relation:

$$\text{Problem 1} \underbrace{\leqslant}_{\text{Lemma 11.1}} \text{(H.35)} \overbrace{\leqslant}^{\substack{\text{(when } \ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)\in[0,M]\\ \text{and } H(\mathcal{P}_{s,y},\mathcal{Q}_{s,y})\leqslant\bar{\gamma}_{s,y}\text{)}}}^{\textbf{(A)}} \text{(H.36)} \underbrace{=}_{\textbf{(B)}} \text{(H.37)} \underbrace{=}_{\textbf{(C)}} \text{(H.38)} \underbrace{\leqslant}_{\textbf{(D)}} \text{(H.39).} \tag{H.40}$$

Thus, when $H(\mathcal{P}_{s,y},\mathcal{Q}_{s,y}) \leqslant \bar{\gamma}_{s,y}$ and $\sup_{(\boldsymbol{X},Y)\in\mathcal{X}\times\mathcal{Y}} \ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y) \leqslant M$, given that ℓ is a non-negative loss by Section 11.2, we can see Equation (H.39), i.e., the expression in Theorem 11.3's statement, upper bounds Problem 1, i.e., provides a fairness certificate for Problem 1. The proofs of these equalities/inequalities are in the following parts labeled by **(A)**, **(B)**, **(C)**, and **(D)** respectively.

Now we show that each C queried by Equation (11.13) (or equally Equation (H.39a)) is a convex optimization. Inspecting C's objective, with respect to the optimizable variable $x_{s,y}$, we find that the only non-linear term in the objective is $\sum_{s=1}^{S}\sum_{y=1}^{C} 2\overline{k_s}\overline{r_y}\sqrt{V_{s,y}}\sqrt{x_{s,y}(1-x_{s,y})}$. Consider the function $f(x)=\sqrt{x(1-x)}$. Define $g(y)=\sqrt{y}$ and $h(x)=x(1-x)$, and then $f(x)=g(h(x))$. Thus, $f'(x)=g'(h(x))h'(x)$ and $f''(x)=g''(h(x))h'(x)^2+g'(h(x))h''(x)$. Notice that $g''(h(x))\leqslant 0$, $g'(h(x))>0$, and $h''(x)<0$ for $x\in(0,1]$. Thus, $f''(x)\leqslant 0$. Since f is twice differentiable in $(0,1]$, we can conclude that f is concave and so does the objective of Equation (11.13). Inspecting C's constraints, we observe that the only non-linear constraint is $\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}\overline{k_s}\overline{r_y}x_{s,y}} \geqslant 1-\rho^2$. Due to the concavity of function $x\mapsto\sqrt{x}$, we have $\sqrt{p_{s,y}\overline{k_s}\overline{r_y}(x^a_{s,y}+x^b_{s,y})/2} \geqslant \frac{1}{2}\left(\sqrt{p_{s,y}\overline{k_s}\overline{r_y}x^a_{s,y}}+\sqrt{p_{s,y}\overline{k_s}\overline{r_y}x^b_{s,y}}\right)$ for any two feasible points $x^a_{s,y}$ and $x^b_{s,y}$. Thus, this non-linear constraint defines a convex region. To this point, we have shown that C's objective is concave and C's constraints are convex, given that C is a maximization problem, C is a convex optimization. QED.

Under the assumptions that $\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)\in[0,M]$ and $H(\mathcal{P}_{s,y},\mathcal{Q}_{s,y})\leqslant\bar{\gamma}_{s,y}$:

(A) *Proof of Equation* (H.35) $\leqslant$ *Equation* (H.36).
Given Equation (H.35d), for each $\mathcal{Q}_{s,y}$, applying Theorem H.1, we get

$$\mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] \leqslant E_{s,y}+2\sqrt{\rho^2_{s,y}(1-\rho^2_{s,y})^2(2-\rho^2_{s,y})}\sqrt{V_{s,y}}+\rho^2_{s,y}(2-\rho^2_{s,y})C_{s,y}. \tag{H.41}$$

Plugging this inequality into all $\mathbb{E}_{(\boldsymbol{X},Y)\sim\mathcal{Q}_{s,y}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)]$ in Equation (H.35a), we obtain Equation (H.36). *QED.*

(B) *Proof of Equation* (H.36) $=$ *Equation* (H.37).
By Equation (H.36d), $\rho_{s,y}\in[0,1]$. Therefore, $x_{s,y}:=(1-\rho^2_{s,y})^2$ is a one-to-one mapping, and we can use $x_{s,y}$ to parameterize $\rho_{s,y}$, which yields Equation (H.37). *QED.*

(C) *Proof of Equation* (H.37) = *Equation* (H.38).
From Equation (H.37b), we notice that the feasible range of k_s and r_y is subsumed by $[0,1]$. We now partition this region $[0,1]$ for each variable to T sub-regions: $[(i-1)/T, i/T]$, $i \in [T]$, and then consider the maximum value across all the combinations of each sub-region for variables k_s and r_y, when feasible. As a result, Equation (H.37) can be written as the maximum over all such sub-problems where k_s's and r_y's enumerate all possible sub-region combinations, which is exactly encoded by Equation (H.38). *QED*.

(D) *Proof of Equation* (H.38) $\leqslant$ *Equation* (H.39).
We only need to show that when $\mathbf{C}'\left(\{[\underline{k_s}, \overline{k_s}]\}_{s=1}^{S}, \{[\underline{r_y}, \overline{r_y}]\}_{y=1}^{C}\right)$ is feasible,

$$\mathbf{C}'\left(\{[\underline{k_s}, \overline{k_s}]\}_{s=1}^{S}, \{[\underline{r_y}, \overline{r_y}]\}_{y=1}^{C}\right) \leqslant \mathbf{C}\left(\{[\underline{k_s}, \overline{k_s}]\}_{s=1}^{S}, \{[\underline{r_y}, \overline{r_y}]\}_{y=1}^{C}\right). \tag{H.42}$$

Since both $\mathbf{C}'$ and $\mathbf{C}$ are maximization problem, we only need to show that the objective of $\mathbf{C}$ upper bounds that of $\mathbf{C}'$, and the constraints of $\mathbf{C}'$ are equal or relaxations of those of $\mathbf{C}$.

For the objective, given that $\underline{k_s} \leqslant k_s \leqslant \overline{k_s}$ and $\underline{r_y} \leqslant r_y \leqslant \overline{r_y}$, for any $x_{s,y}$, We observe that

$$\begin{aligned}
k_s r_y(E_{s,y} + C_{s,y}) &\leqslant \overline{k_s}\,\overline{r_y}\left[E_{s,y} + C_{s,y}\right]_+ + \underline{k_s}\,\underline{r_y}\left[E_{s,y} + C_{s,y}\right]_-, \\
k_s r_y \cdot \left(2\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{V_{s,y}}\right) &\leqslant \overline{k_s}\,\overline{r_y} \cdot \left(2\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{V_{s,y}}\right), \\
-k_s r_y C_{s,y} x_{s,y} &\leqslant -\underline{k_s}\,\underline{r_y} x_{s,y}[C_{s,y}]_+ - \overline{k_s}\,\overline{r_y} x_{s,y}[C_{s,y}]_-,
\end{aligned} \tag{H.43}$$

and by summing up all these terms for all $s \in [S]$ and $y \in [C]$, the LHS would be the objective of $\mathbf{C}'$ and the RHS would be the objective of $\mathbf{C}$. Hence, $\mathbf{C}$'s objective upper bounds that of $\mathbf{C}'$.

For the constraints, similarly, given that $\underline{k_s} \leqslant k_s \leqslant \overline{k_s}$ and $\underline{r_y} \leqslant r_y \leqslant \overline{r_y}$, we have

$$\text{(H.38c)} \sum_{s=1}^{S} k_s = 1,\ \sum_{y=1}^{C} r_y = 1 \Longrightarrow \sum_{s=1}^{S} \underline{k_s} \leqslant 1,\ \sum_{s=1}^{S} \overline{k_s} \geqslant 1,\ \sum_{y=1}^{C} \underline{r_y} \leqslant 1,\ \sum_{y=1}^{C} \overline{r_y} \geqslant 1 \text{(H.39c)},$$

$$\text{(H.38d)} \sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y x_{s,y}} \geqslant 1-\rho^2 \Longrightarrow \sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y}\overline{k_s}\,\overline{r_y} x_{s,y}} \geqslant 1-\rho^2 \text{(H.39d)},$$

$$\text{(H.38e) is as same as (H.39e)},$$

which implies that all feasible solutions of $\mathbf{C}'$ are also feasible for $\mathbf{C}$. Combining with

the fact that for any solution of $\mathbf{C}'$, its objective value $\mathbf{C}$ is greater than or equal to that of $\mathbf{C}'$ as shown above, we have Equation (H.42) which concludes the proof. QED.

H.2 THEOREM STATEMENTS AND PROOFS FOR FINITE SAMPLING ERROR

H.2.1 Finite Sampling Confidence Intervals

Lemma H.1. Let $\hat{P}$ be set of i.i.d. finite samples from $\mathcal{P}$, and let $\hat{P}_{s,y} := \{(X_i, Y_i) \in \hat{P} : (X_i)_s = s, Y_i = y\}$ for any $s \in [S], y \in [C]$. Let $\ell : (\hat{y}, y) \to [0, M]$ be a loss function. We define $\hat{L}_n = \frac{1}{|\hat{P}_{s,y}|} \sum_{(X_i,Y_i)\in\hat{P}_{s,y}} \ell(h_{\boldsymbol{\theta}}(X_i), Y_i)$, $s_n^2 = \frac{1}{n(n-1)} \sum_{1\leqslant i<j\leqslant n} \left(\ell(h_{\boldsymbol{\theta}}(X_i), Y) - \ell(h_{\boldsymbol{\theta}}(X_j), Y)\right)^2$, and $\hat{P}_{s,y} := \{(X_i, Y_i) \in \hat{P} : (X_i)_s = s, Y_i = y\}$. Then for $\delta > 0$, with respect to the random draw of $\hat{P}$ from $\mathcal{P}$, we have

$$\Pr\left(\hat{L}_n - M\sqrt{\frac{\ln(2/\delta)}{2|\hat{P}_{s,y}|}} \leqslant \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell\left(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y\right)] \leqslant \hat{L}_n + M\sqrt{\frac{\ln(2/\delta)}{2|\hat{P}_{s,y}|}}\right) \geqslant 1-\delta, \tag{H.44}$$

$$\Pr\left(\sqrt{s_n^2} - M\sqrt{\frac{2\ln(2/\delta)}{|\hat{P}_{s,y}|-1}} \leqslant \sqrt{\mathop{\mathbb{V}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell\left(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y\right)]} \leqslant \sqrt{s_n^2} + M\sqrt{\frac{2\ln(2/\delta)}{|\hat{P}_{s,y}|-1}}\right) \geqslant 1-\delta, \tag{H.45}$$

$$\Pr\left(\frac{|\hat{P}_{s,y}|}{|\hat{P}|} - \sqrt{\frac{\ln(2/\delta)}{2|\hat{P}|}} \leqslant \Pr_{(\boldsymbol{X},Y)\sim\mathcal{P}}[X_s = s, Y = y] \leqslant \frac{|\hat{P}_{s,y}|}{|\hat{P}|} + \sqrt{\frac{\ln(2/\delta)}{2|\hat{P}|}}\right) \geqslant 1-\delta. \tag{H.46}$$

Proof of Lemma H.1. We can get Equation (H.45) according to Theorem 10 in [257]. Here, we will provide proofs for Equation (H.44) and Equation (H.46), respectively. The general idea is to use Hoeffding's inequality to get the high-confidence interval.

We will prove Equation (H.44) first. From Hoeffding's inequality, for all $t > 0$, we have:

$$\Pr\left(\left|\hat{L}_n - \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}[\ell\left(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y\right)]\right| \geqslant t\right) \leqslant 2\exp\left(-\frac{2|\hat{P}_{s,y}|^2t^2}{|\hat{P}_{s,y}|M^2}\right) \tag{H.47}$$

Since we want to get an interval with confidence $1-\delta$, we let $2\exp\left(-\frac{2|\hat{P}_{s,y}|^2t^2}{|\hat{P}_{s,y}|M^2}\right) = \delta$, from which we can derive that

$$t = M\sqrt{\frac{\ln(2/\delta)}{2|\hat{P}_{s,y}|}} \tag{H.48}$$

Plugging Equation (H.48) into Equation (H.47), we can get:

$$\Pr\left(\hat{L}_n - M\sqrt{\frac{\ln(2/\delta)}{2|\hat{P}_{s,y}|}} \leqslant \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{P}_{s,y}}\left[\ell\left(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y\right)\right] \leqslant \hat{L}_n + M\sqrt{\frac{\ln(2/\delta)}{2|\hat{P}_{s,y}|}}\right) \geqslant 1-\delta \quad \text{(H.49)}$$

Then we will prove Equation (H.46). From Hoeffding's inequality, for all $t > 0$, we have:

$$\Pr\left(\left|\frac{|\hat{P}_{s,y}|}{|\hat{P}|} - \Pr_{(\boldsymbol{X},Y)\sim\mathcal{P}}[X_s = s, Y = y]\right| \geqslant t\right) \leqslant 2\exp\left(-\frac{2|\hat{P}|^2t^2}{|\hat{P}|}\right) \quad \text{(H.50)}$$

Since we want to get an interval with confidence $1-\delta$, we let $2\exp\left(-\frac{2|\hat{P}|^2t^2}{|\hat{P}|}\right) = \delta$, from which we can derive that

$$t = \sqrt{\frac{\ln(2/\delta)}{2|\hat{P}|}} \quad \text{(H.51)}$$

Plugging Equation (H.51) into Equation (H.50), we can get:

$$\Pr\left(\frac{|\hat{P}_{s,y}|}{|\hat{P}|} - \sqrt{\frac{\ln(2/\delta)}{2|\hat{P}|}} \leqslant \Pr_{(\boldsymbol{X},Y)\sim\mathcal{P}}[X_s = s, Y = y] \leqslant \frac{|\hat{P}_{s,y}|}{|\hat{P}|} + \sqrt{\frac{\ln(2/\delta)}{2|\hat{P}|}}\right) \geqslant 1-\delta \quad \text{(H.52)}$$

QED.

H.2.2 Fairness Certification Statements with Finite Sampling

Theorem H.2 (Theorem 11.2 with finite sampling)**.** Given a distance bound $\rho > 0$ and any $\delta > 0$, the following constrained optimization, which is **convex**, when feasible, provides a fairness certificate for Problem 2 with probability at least $1 - 2SC\delta$:

$$\max_{k_s, r_y, p_{s,y}} \quad \sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y \overline{E_{s,y}} \quad \text{(H.53a)}$$

$$\text{s.t.} \quad \sum_{s=1}^{S} k_s = 1, \quad \sum_{y=1}^{C} r_y = 1, \quad k_s \geqslant 0 \quad \forall s \in [S], \quad r_y \geqslant 0 \quad \forall y \in [C], \quad \text{(H.53b)}$$

$$1 - \rho^2 - \sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}k_s r_y} \leqslant 0, \quad \text{(H.53c)}$$

$$\underline{p_{s,y}} \leqslant p_{s,y} \leqslant \overline{p_{s,y}}, \quad \forall s \in [S], \quad \forall y \in [C] \quad \text{(H.53d)}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C} p_{s,y} = 1 \tag{H.53e}$$

where $\overline{E_{s,y}} := \hat{L}_n + M\sqrt{\ln(2/\delta)/\left(2|\hat{P}_{s,y}|\right)}$, $\underline{p_{s,y}} := |\hat{P}_{s,y}|/|\hat{P}| - \sqrt{\ln(2/\delta)/\left(2|\hat{P}|\right)}$, $\overline{p_{s,y}} := |\hat{P}_{s,y}|/|\hat{P}| + \sqrt{\ln(2/\delta)/\left(2|\hat{P}|\right)}$ are constants computed with Lemma H.1.

Theorem H.3. If for any $s \in [S]$ and $y \in [Y]$, $H(\mathcal{P}_{s,y}, \mathcal{Q}_{s,y}) \leqslant \bar{\gamma}_{s,y}$ and $0 \leqslant \sup_{(\boldsymbol{X},Y)\in\mathcal{X}\times\mathcal{Y}} \ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}), Y) \leqslant M$, given a distance bound $\rho > 0$ and any $\delta > 0$, for any region granularity $T \in \mathbb{N}_+$, the following expression provides a fairness certificate for Problem 1 with probability at least $1 - 3SC\delta$:

$$\bar{\ell} = \max_{\{i_s\in[T]:s\in[S]\},\{j_y\in[T]:y\in[C]\}} \mathbf{C}\left(\left\{\left[\frac{i_s-1}{T}, \frac{i_s}{T}\right]\right\}_{s=1}^{S}, \left\{\left[\frac{j_y-1}{T}, \frac{j_y}{T}\right]\right\}_{y=1}^{C}\right), \text{ where} \tag{H.54}$$

$$\mathbf{C}\left(\{[\underline{k_s}, \overline{k_s}]\}_{s=1}^{S}, \{[\underline{r_y}, \overline{r_y}]\}_{y=1}^{C}\right) = \max_{x_{s,y},p_{s,y}} \sum_{s=1}^{S}\sum_{y=1}^{C}\left(\overline{k_s}\,\overline{r_y}\left[\overline{E_{s,y}} + \overline{C_{s,y}}\right]_+ + \underline{k_s}\,\underline{r_y}\left[\overline{E_{s,y}} + \overline{C_{s,y}}\right]_-\right.$$
$$\left. +2\overline{k_s}\,\overline{r_y}\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{\overline{V_{s,y}}} - \underline{k_s}\,\underline{r_y}x_{s,y}[\underline{C_{s,y}}]_+ - \overline{k_s}\,\overline{r_y}x_{s,y}[\underline{C_{s,y}}]_-\right) \tag{H.55a}$$

$$\text{s.t.} \quad \sum_{s=1}^{S}\underline{k_s} \leqslant 1, \quad \sum_{s=1}^{S}\overline{k_s} \geqslant 1, \quad \sum_{y=1}^{C}\underline{r_y} \leqslant 1, \quad \sum_{y=1}^{C}\overline{r_y} \geqslant 1, \tag{H.55b}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}\overline{k_s}\,\overline{r_y}x_{s,y}} \geqslant 1-\rho^2, \quad \left(1-\overline{\bar{\gamma}_{s,y}^2}\right)^2 \leqslant x_{s,y} \leqslant 1, \tag{H.55c}$$

$$\underline{p_{s,y}} \leqslant p_{s,y} \leqslant \overline{p_{s,y}}, \quad \sum_{s=1}^{S}\sum_{y=1}^{C} p_{s,y} = 1 \tag{H.55d}$$

where $\underline{E_{s,y}} := \hat{L}_n - M\sqrt{\ln(2/\delta)/\left(2|\hat{P}_{s,y}|\right)}$, $\overline{E_{s,y}} := \hat{L}_n + M\sqrt{\ln(2/\delta)/\left(2|\hat{P}_{s,y}|\right)}$, $\underline{V_{s,y}} = \left(\sqrt{s_n^2} - M\sqrt{2\ln(2/\delta)/\left(|\hat{P}_{s,y}|-1\right)}\right)^2$, $\overline{V_{s,y}} = \left(\sqrt{s_n^2} + M\sqrt{2\ln(2/\delta)/\left(|\hat{P}_{s,y}|-1\right)}\right)^2$, $\underline{p_{s,y}} := |\hat{P}_{s,y}|/|\hat{P}| - \sqrt{\ln(2/\delta)/\left(2|\hat{P}|\right)}$, $\overline{p_{s,y}} := |\hat{P}_{s,y}|/|\hat{P}| + \sqrt{\ln(2/\delta)/\left(2|\hat{P}|\right)}$ computed with Lemma H.1, and $\underline{C_{s,y}} = M - \overline{E_{s,y}} - \overline{V_{s,y}}/(M - \overline{E_{s,y}})$, $\overline{C_{s,y}} = M - \underline{E_{s,y}} - \underline{V_{s,y}}/(M - \underline{E_{s,y}})$, $\overline{\bar{\gamma}_{s,y}^2} = 1 - (1 + (M - \underline{E_{s,y}})^2/\overline{V_{s,y}})^{-\frac{1}{2}}$. Equation (H.54) only takes $\mathbf{C}$'s value when it is feasible, and each $\mathbf{C}$ queried by Equation (H.54) is a **convex optimization**.

H.2.3 Proofs of Fairness Certification with Finite Sampling

High-level Illustration. We use Hoeffding's inequality to bound the finite sampling error of statistics and add the high confidence box constraints to the optimization problems, which can still be proved to be convex.

Proof of Theorem H.2. The proof of Theorem H.2 is composed of two parts: (1) the optimization problem provides a fairness certificate for Problem 2; (2) the optimization problem is convex.

(1) We prove that Theorem H.2 provides a fairness certificate for Problem 2 in this part. Since Theorem 11.2 provides a fairness certificate for Problem 2, we only need to prove: (a) the feasible region of the optimization problem in Theorem 11.2 is a subset of the feasible region of the optimization problem in Theorem H.2, and (b) the optimization objective in Theorem 11.2 can be upper bounded by that in Theorem H.2.

To prove (a), we first equivalently transform the optimization problem in Theorem 11.2 into the following optimization problem by adding p_{sy} to the decision variables:

$$\max_{k_s, r_y, p_{s,y}} \quad \sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y E_{s,y} \tag{H.56a}$$

$$\text{s.t.} \quad \sum_{s=1}^{S} k_s = 1, \quad \sum_{y=1}^{C} r_y = 1, \quad k_s \geqslant 0 \quad \forall s \in [S], \quad r_y \geqslant 0 \quad \forall y \in [C], \tag{H.56b}$$

$$1 - \rho^2 - \sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y} \leqslant 0, \tag{H.56c}$$

$$p_{s,y} = |\hat{P}_{s,y}|/|\hat{P}|, \quad \forall s \in [S], \quad \forall y \in [C] \tag{H.56d}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C} p_{s,y} = 1 \tag{H.56e}$$

For decision variables $k_{s,y}$ and $r_{s,y}$, optimization H.53 and H.56 has the same constraints (Equation (H.53b) and Equation (H.56b)). For decision variables $p_{s,y}$, the feasible region of $p_{s,y}$ in optimization H.53 (decided by Equations (H.53d) and (H.53e)) is a subset of the feasible region of $p_{s,y}$ in optimization H.56 (decided by Equations (H.56d) and (H.56e)), since $\underline{p_{s,y}} \leqslant |\hat{P}_{s,y}|/|\hat{P}| \leqslant \overline{p_{s,y}}$. Therefore, the feasible region with respect to $k_{s,y}$, $r_{s,y}$, and $p_{s,y}$ of the optimization problem in Theorem 11.2 is a subset of that in Theorem H.2.

To prove (b), we only need to show that the objective in Equation (H.53a) can be upper bounded by the objective in Equation (H.56a). The statement $\sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y E_{s,y} \leqslant$

$\sum_{s=1}^{S}\sum_{y=1}^{C} k_s r_y \overline{E_{s,y}}$ consistently holds because $E_{s,y} \leqslant \overline{E_{s,y}}$ and $k_s, r_y \geqslant 0$.

Combining the proofs of (a) and (b), we prove that Theorem H.2 provides a fairness certificate for Problem 2.

(2) Inspecting that the objective and all the constraints in optimization problem in Equation (H.53) are linear with respect to k_s, r_y, $p_{s,y}$ but the one in Equation (H.53c). Therefore, we only need to prove that the following constraint is convex with respect to k_s, r_y, $p_{s,y}$:

$$1 - \rho^2 - \sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y} \leqslant 0 \tag{H.57}$$

We define a function f with respect to vector $\mathbf{p} := [p_{s,y}]_{s\in[S],y\in[C]}$: $f(\mathbf{p}) = 1 - \rho^2 - \sum_{s=1}^{S}\sum_{y=1}^{C} \sqrt{p_{s,y} k_s r_y}$. Then we can derive that:

$$\frac{\partial^2 f}{\partial \mathbf{p}^2} = \sum_{s=1}^{S}\sum_{y=1}^{C} \frac{\sqrt{k_s r_y}}{4} {p_{s,y}}^{-\frac{3}{2}} \geqslant 0 \tag{H.58}$$

Therefore, the function f is convex with respect to $p_{s,y}$. Similarly, we can prove the convexity with respect to k_s and r_y. Finally, we can conclude that the constraint in Equation (H.57) is convex with respect to k_s, r_y, $p_{s,y}$ and the optimization problem defined in Theorem H.2 is convex.

Since we use the union bound to bound $E_{s,y}$ and $p_{s,y}$ for all $s \in [S], y \in [C]$ simultaneously, the confidence is $1 - 2SC\delta$. QED.

Proof of Theorem H.3. The proof of Theorem H.3 includes two parts: (1) the optimization problem provides a fairness certificate for Problem 1; (2) each $\mathbf{C}$ queried by Equation (H.54) is a convex optimization.

(1) Since Theorem 11.3 provides a fairness certificate for Problem 1, we only need to prove: (a) the feasible region of the optimization problem in Theorem 11.3 is a subset of that in Theorem H.3, and (b) the optimization objective in Theorem 11.3 can be upper bounded by that in Theorem H.3.

To prove (a), we first equivalently transform the optimization problem in Theorem 11.3 into the following optimization problem by adding p_{sy} to the decision variables:

$$\mathbf{C}\left(\{[\underline{k_s}, \overline{k_s}]\}_{s=1}^{S}, \{[\underline{r_y}, \overline{r_y}]\}_{y=1}^{C}\right) = \max_{x_{s,y}, p_{s,y}} \sum_{s=1}^{S}\sum_{y=1}^{C} \left(\overline{k_s}\,\overline{r_y}\left[E_{s,y} + C_{s,y}\right]_+ + \underline{k_s}\,\underline{r_y}\left[E_{s,y} + C_{s,y}\right]_-\right.$$

$$+2\overline{k_s}\overline{r_y}\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{V_{s,y}}-\underline{k_s}\underline{r_y}x_{s,y}[C_{s,y}]_+-\overline{k_s}\overline{r_y}x_{s,y}[C_{s,y}]_-\Big) \tag{H.59a}$$

$$\text{s.t.}\quad \sum_{s=1}^{S}\underline{k_s}\leqslant 1,\quad \sum_{s=1}^{S}\overline{k_s}\geqslant 1,\sum_{y=1}^{C}\underline{r_y}\leqslant 1,\quad \sum_{y=1}^{C}\overline{r_y}\geqslant 1, \tag{H.59b}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}\overline{k_s}\overline{r_y}x_{s,y}}\geqslant 1-\rho^2,\quad \left(1-\bar{\gamma}_{s,y}^2\right)^2\leqslant x_{s,y}\leqslant 1, \tag{H.59c}$$

$$p_{s,y}=|\hat{P}_{s,y}|/|\hat{P}|,\quad \forall s\in[S],\quad \forall y\in[C] \tag{H.59d}$$

$$\sum_{s=1}^{S}\sum_{y=1}^{C}p_{s,y}=1 \tag{H.59e}$$

For decision varibales $x_{s,y}$, since $\sqrt{p_{s,y}\overline{k_s}\overline{r_y}x_{s,y}}\leqslant\sqrt{\overline{p_{s,y}}\overline{k_s}\overline{r_y}x_{s,y}}$ and $\left(1-\bar{\gamma}_{s,y}^2\right)^2\geqslant\left(1-\overline{\bar{\gamma}_{s,y}^2}\right)^2$, the feasible region of $x_{s,y}$ in Equation (H.59) is a subset of that in Equation (H.55). For decision variables $p_{s,y}$, since $\underline{p_{s,y}}\leqslant|\hat{P}_{s,y}|/|\hat{P}|\leqslant\overline{p_{s,y}}$, the feasible region of $p_{s,y}$ in Equation (H.59) is also a subset of that in Equation (H.55). Therefore, the feasible region of the optimization problem in Theorem 11.3 is a subset of that in Theorem H.3.

To prove (b), we only need to show that the objective in Equation (H.59a) can be upper bounded by the objective in Equation (H.55a). Since $\underline{k_s},\overline{k_s},\underline{r_y},\overline{r_y}\geqslant 0$ and $0\leqslant x_{s,y}\leqslant 1$ hold, we can observe that $\forall s\in[S],\forall y\in[C]$,

$$\begin{aligned}&\overline{k_s}\overline{r_y}\left[E_{s,y}+C_{s,y}\right]_+ +\underline{k_s}\underline{r_y}\left[E_{s,y}+C_{s,y}\right]_- +2\overline{k_s}\overline{r_y}\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{V_{s,y}}-\underline{k_s}\underline{r_y}x_{s,y}[C_{s,y}]_+-\\ &\overline{k_s}\overline{r_y}x_{s,y}[C_{s,y}]_- \quad\leqslant\quad \overline{k_s}\overline{r_y}\left[\overline{E_{s,y}}+\overline{C_{s,y}}\right]_+ +\underline{k_s}\underline{r_y}\left[\overline{E_{s,y}}+\overline{C_{s,y}}\right]_- +2\overline{k_s}\overline{r_y}\sqrt{x_{s,y}(1-x_{s,y})}\sqrt{\overline{V_{s,y}}}\\ &-\underline{k_s}\underline{r_y}x_{s,y}[\underline{C_{s,y}}]_+-\overline{k_s}\overline{r_y}x_{s,y}[\underline{C_{s,y}}]_-\end{aligned} \tag{H.60}$$

Therefore, we prove that the optimization in Theorem H.3 provides a fairness certificate for Problem 1.

(2) We will prove that each $\mathbf{C}$ queried by Equation (H.54) is a convex optimization with respect to decision variables $x_{s,y}$ and $p_{s,y}$ in this part. In the proof of Theorem 11.3, we provide the proof of convexity with respect to $x_{s,y}$, so we only need to prove that the optimization problem is convex with respect to $p_{s,y}$. We can observe that the constraints of $p_{s,y}$ in Equation (H.55d) is linear, and thus we only need to prove that $\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}\overline{k_s}\overline{r_y}x_{s,y}}\geqslant 1-\rho^2$ (the constraint in Equation (H.55c)) is convex with respect to $p_{s,y}$. Here, we define a function f with respect to vector $\boldsymbol{p}:=[p_{s,y}]_{s\in[S],y\in[C]}$:

$f(\boldsymbol{p}) = 1 - \rho^2 - \sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}\overline{k_s}\overline{r_y}}$. Then we can derive that:

$$\left(\frac{\partial^2 f}{\partial \boldsymbol{p}^2}\right)_{sy,s'y'} = \sum_{s=1}^{S}\sum_{y=1}^{C}\frac{\sqrt{\overline{k_s}\overline{r_y}}}{4}{p_{s,y}}^{-\frac{3}{2}} \cdot \mathbb{I}[s = s', y = y'] \geqslant 0 \tag{H.61}$$

Thus, the function f is convex and $f(\boldsymbol{p}) \leqslant 0$ defines a convex set with respect to $p_{s,y}$. Then, we prove that the constraint $\sum_{s=1}^{S}\sum_{y=1}^{C}\sqrt{p_{s,y}\overline{k_s}\overline{r_y}x_{s,y}} \geqslant 1 - \rho^2$ is convex with respect to $p_{s,y}$.

Since we use the union bound to bound $E_{s,y}$, $V_{s,y}$ and $p_{s,y}$ for all $s \in [S], y \in [C]$ simultaneously, the confidence is $1 - 3SC\delta$. QED.

H.3 EXPERIMENT DETAILS

H.3.1 Datasets

We validate our certified fairness on *six* real-world datasets: Adult [14], Compas [11], Health [161], Lawschool [403], Crime [14], and German [14]. All the used datasets contain categorical data.

In Adult dataset, we have 14 attributes of a person as input and try to predict whether the income of the person is over 50k \$/year. The sensitive attribute in Adult is selected as the sex.

In Compas dataset, given the attributes of a criminal defendent, the task is to predict whether he/she will re-offend in two years. The sensitive attribute in Compas is selected as the race.

In Health dataset, given the physician records and and insurance claims of the patients, we try to predict ten-year mortality by binarizing the Charlson Index, taking the median value as a cutoff. The sensitive attribute in Health is selected as the age.

In Lawschool dataset, we try to predict whether a student passes the exam according to the appication records of different law schools. The sensitive attribute in Lawschool is the race.

In Crime dataset, we try to predict whether a specific community is above or below the median number of violent crimes per population. The sensitive attribute in Crime is selected as the race.

In German dataset, each person is classified as good or bad credit risks according to the set of attributes. The sensitive attribute in German is selected as the sex.

Following [313], we consider the scenario where sensitive attributes and labels take binary values, and we also follow their standard data processing steps: (1) normalize the numerical values of all attributes with the mean value 0 and variance 1, (2) use one-hot encodings to represent categorical attributes, (3) drop instances and attributes with missing values, and (4) split the datasets into training set, validation set, and test set.

Training Details. We directly train a ReLU network composed of two hidden layers on the training set of six datasets respectively following the setting in [313]. Concretely, we train the models for 100 epochs with a batch size of 256. We adopt the binary cross-entropy loss and use the Adam optimizer with weight decay 0.01 and dynamic learning rate scheduling (ReduceLROnPlateau in [289]) based on the loss on the validation set starting at 0.01 with the patience of 5 epochs.

H.3.2 Fair Base Rate Distribution Generation Protocol

To evaluate how well our certificates capture the fairness risk in practice, we compare our certification bound with the empirical loss evaluated on randomly generated $30,000$ fairness constrained distributions $\mathcal{Q}$ shifted from $\mathcal{P}$. Now, we introduce the protocols to generate fairness distributions $\mathcal{Q}$ for sensitive shifting and general shifting, respectively. Note that the protocols are only valid when the sensitive attributes and labels take binary values.

Fair base rate distributions generation steps in the *sensitive shifting* scenario:

(1) Sample the proportions of subpopulations of the generated distribution $q_{0,0}, q_{0,1}, q_{1,0}, q_{1,1}$: uniformly sample two real values in the interval $[0, 1]$, and do the assignment: $q_{0,0} := kr$, $q_{0,1} := k(1-r)$, $q_{1,0} := (1-k)r$, $q_{1,1} := (1-k)(1-r)$.

(2) Determine the sample size of every subpopulation: first determine the subpopulation which requires the largest sample size, use all the samples in that subpopulation, and then calculate the sample size in other subpopulations according to the proportions.

(3) Uniformly sample in each subpopulation based on the sample size.

(4) Calculate the Hellinger distance $H(\mathcal{P}, \mathcal{Q}) = \sqrt{1 - \sum_{s=0}^{1} \sum_{y=0}^{1} \sqrt{p_{s,y}} \sqrt{q_{s,y}}}$. Suppose that the support of $\mathcal{P}$ and $\mathcal{Q}$ is $\mathcal{X} \times \mathcal{Y}$ and the densities of $\mathcal{P}$ and $\mathcal{Q}$ with respect to a suitable measure are $f_{\mathcal{P}}$ and $f_{\mathcal{Q}}$, respectively. Since we consider sensitive shifting here, we have $f_{\mathcal{Q}_{s,y}} = \lambda_{s,y} f_{\mathcal{P}_{s,y}}, \quad s, y \in \{0, 1\}$ where $\lambda_{s,y}$ is a scalor. The derivation of the distance

calculation formula is shown as follows,

$$H^2(\mathcal{P},\mathcal{Q}) = 1 - \iint\limits_{\mathcal{X}\times\mathcal{Y}} \sqrt{f_{\mathcal{P}}(\boldsymbol{x},y)}\sqrt{f_{\mathcal{Q}}(\boldsymbol{x},y)}\mathrm{d}\boldsymbol{x}\mathrm{d}y \tag{H.62a}$$

$$= 1 - \sum_{s=0}^{1}\sum_{y=0}^{1} \iint\limits_{f_{\mathcal{P}_{s,y}}(\boldsymbol{x},y)>0} \sqrt{f_{\mathcal{P}_{s,y}}(\boldsymbol{x},y)}\sqrt{\lambda_{s,y}f_{\mathcal{P}_{s,y}}}\mathrm{d}\boldsymbol{x}\mathrm{d}y \tag{H.62b}$$

$$= 1 - \sum_{s=0}^{1}\sum_{y=0}^{1} \sqrt{\lambda_{s,y}} \iint\limits_{f_{\mathcal{P}_{s,y}}(x,y)>0} f_{\mathcal{P}_{s,y}}(\boldsymbol{x},y)\mathrm{d}\boldsymbol{x}\mathrm{d}y \tag{H.62c}$$

$$= 1 - \sum_{s=0}^{1}\sum_{y=0}^{1} \sqrt{\lambda_{s,y}}p_{s,y} \tag{H.62d}$$

$$= 1 - \sum_{s=0}^{1}\sum_{y=0}^{1} \sqrt{p_{s,y}}\sqrt{q_{s,y}}. \tag{H.62e}$$

Fair base rate distribution generation steps in the *general shifting* scenario:

(1) Construct a data distribution $\mathcal{Q}'$ that is disjoint with the training data distribution $\mathcal{P}$ by changing the distribution of non-sensitive values given the sensitive attributes and labels.

(2) Sample mixing parameters $\alpha_{s,y}$ and $\alpha'_{s,y}$ in the interval $[0,1]$ satisfying $\frac{p_{00}\alpha_{00}+q_{00}\alpha'_{00}}{p_{01}\alpha_{01}+q_{01}\alpha'_{01}} = \frac{p_{10}\alpha_{10}+q_{10}\alpha'_{10}}{p_{11}\alpha_{11}+q_{11}\alpha'_{11}}$ (base rate parity) and $p_{00}\alpha_{00} + q_{00}\alpha'_{00} + p_{01}\alpha_{01} + q_{01}\alpha'_{01} + p_{10}\alpha_{10} + q_{10}\alpha'_{10} + p_{11}\alpha_{11} + q_{11}\alpha'_{11} = 1$.

(3) Determine the proportion of every subpopulation in distribution $\mathcal{Q}$: $q_{s,y} := \alpha_{s,y}p_{s,y} + \alpha'_{s,y}q'_{s,y}, \quad s,y \in \{0,1\}$.

(4) Determine the sample size of every subpopulation in $\mathcal{P}$ and $\mathcal{Q}'$: first determine the subpopulation which requires the largest sample size, use all the samples in that subpopulation, and then calculate the sample size in other subpopulations according to the proportions.

(5) Calculate the Hellinger distance between distribution $\mathcal{P}$ and $\mathcal{Q}$:

$$H(\mathcal{P},\mathcal{Q}) = \sqrt{1 - \sum_{s=0}^{1}\sum_{y=0}^{1} \sqrt{\alpha_{s,y}}p_{s,y}}. \tag{H.63}$$

Suppose that the support of $\mathcal{P}$ and $\mathcal{Q}$ is $\mathcal{X} \times \mathcal{Y}$ and the densities of $\mathcal{P}$ and $\mathcal{Q}$ with respect to a suitable measure are $f_{\mathcal{P}}$ and $f_{\mathcal{Q}}$, respectively. The derivation of the distance calculation formula is shown as follows,

$$H^2(\mathcal{P}, \mathcal{Q}) = 1 - \iint\limits_{\mathcal{X}\times\mathcal{Y}} \sqrt{f_{\mathcal{P}}(\boldsymbol{x}, y)}\sqrt{f_{\mathcal{Q}}(\boldsymbol{x}, y)}\mathrm{d}\boldsymbol{x}\mathrm{d}y \tag{H.64a}$$

$$= 1 - \iint\limits_{\mathcal{X}\times\mathcal{Y}} \sqrt{\sum_{s=0}^{1}\sum_{y=0}^{1} f_{\mathcal{P}_{s,y}}(\boldsymbol{x}, y)}\sqrt{\sum_{s=0}^{1}\sum_{y=0}^{1}\Big(\alpha_{s,y} f_{\mathcal{P}_{s,y}}(\boldsymbol{x}, y) + \alpha'_{s,y} f_{\mathcal{Q}'_{s,y}}(\boldsymbol{x}, y)\Big)}\mathrm{d}x\mathrm{d}y \tag{H.64b}$$

$$= 1 - \sum_{s=0}^{1}\sum_{y=0}^{1}\sqrt{\alpha_{s,y}} \iint\limits_{f_{\mathcal{P}_{s,y}}(\boldsymbol{x},y)>0} f_{\mathcal{P}_{s,y}}(\boldsymbol{x}, y)\sqrt{1 + \frac{\alpha'_{s,y} f_{\mathcal{Q}_{s,y}}(\boldsymbol{x}, y)}{\alpha_{s,y} f_{\mathcal{P}_{s,y}}(\boldsymbol{x}, y)}}\mathrm{d}x\mathrm{d}y \tag{H.64c}$$

$$= 1 - \sum_{s=0}^{1}\sum_{y=0}^{1}\sqrt{\alpha_{s,y}} \iint\limits_{f_{\mathcal{P}_{s,y}}(\boldsymbol{x},y)>0} f_{\mathcal{P}_{s,y}}(\boldsymbol{x}, y)\mathrm{d}\boldsymbol{x}\mathrm{d}y \tag{H.64d}$$

$$= 1 - \sum_{s=0}^{1}\sum_{y=0}^{1}\sqrt{\alpha_{s,y}} p_{s,y}. \tag{H.64e}$$

H.3.3 Implementation Details of Our Fairness Certification

We conduct vanilla training and then calculate our certified fairness according to our certification framework. Concretely, in the training step, we train a ReLU network composed of 2 hidden layers of size 20 for 100 epochs with binary cross entropy loss (BCE loss) using an Adam optimizer. The initial learning rate is 0.05 for Crime and German datasets, while for other datasets, the initial learning rate is set 0.001. We reduce the learning rate with a factor of 0.5 on the plateau measured by the loss on the validation set with patience of 5 epochs. In the fairness certification step, we set the region granularity to be 0.005 for certification in the general shifting scenario. We use 90% confidence interval when considering finite sampling error. The codes we used follow the MIT license. All experiments are conducted on a 1080 Ti GPU with 11,178 MB memory.

H.3.4 Implementation Details of WRM

The optimization problem of tackling distributional robustness is formulated as:

$$\max_{\mathcal{Q}} \mathop{\mathbb{E}}_{(\boldsymbol{X},Y)\sim\mathcal{Q}}[\ell(h_{\boldsymbol{\theta}}(\boldsymbol{X}),Y)] \quad \text{s.t.} \quad \text{dist}(\mathcal{P},\mathcal{Q}) \leqslant \rho \tag{H.65}$$

where $\text{dist}(\cdot,\cdot)$ is a predetermined distribution distance metric. Note that the optimization is the same as our certified fairness optimization in Problem 1 except for the fairness constraint.

WRM [347] proposes to use the dual reformulation of the Wasserstein worst-case risk to provide the distributional robustness certificate, which is formulated in the following proposition.

Proposition H.1 ([347], Proposition 1)**.** Let $\ell : \Theta \times \mathcal{Z} \to \mathbb{R}$ and $c : \Theta \times \mathcal{Z} \to \mathbb{R}_+$ be continuous and let $\phi_\gamma(\boldsymbol{\theta}; z_0) := \sup_{z\in\mathcal{Z}}\{\ell(z;\boldsymbol{\theta}) - \gamma c(z;\boldsymbol{\theta})\}$. Then, for any distribution $\mathcal{P}$ and any $\rho > 0$,

$$\sup_{\mathcal{Q}:W_c(\mathcal{P},\mathcal{Q})\leqslant\rho} \mathop{\mathbb{E}}_{\mathcal{Q}}[\ell(\boldsymbol{\theta};Z)] = \inf_{\gamma\geqslant 0}\{\gamma\rho + \mathop{\mathbb{E}}_{\mathcal{P}}[\phi_\gamma(\boldsymbol{\theta};Z)]\} \tag{H.66}$$

where $W_c(\mathcal{P},\mathcal{Q}) := \inf_{\pi\in\Pi(\mathcal{P},\mathcal{Q})}\int_{\mathcal{Z}} c(z,z')d\pi(z,z')$ is the 1-Wasserstein distance between $\mathcal{P}$ and $\mathcal{Q}$.

One requirement for the certificate to be tractable is that the surrogate function ϕ_γ is concave with respect to Z, which holds when γ is larger than the Lipschitz constant L of the gradient of ℓ with respect to Z. Since we use the ELU network with JSD loss, we can efficiently calculate γ iteratively as shown in Appendix D of [399].

We select Gaussian mixture data for fair comparison. The Gaussian mixture data can be formulated as $P(x|\boldsymbol{\theta}) = \sum_{k=1}^{K}\alpha_k\phi(x|\boldsymbol{\theta}_k)$ where K is the number of Gaussian data, α_k is the proportion of the k-th Gaussian, and $\boldsymbol{\theta}_k = (\mu_k, \sigma_k^2)$. In our evaluation, we use 2-dimension Gaussian and mixture data composed of 2 Gaussian ($K = 2$) labeled 0 and 1, respectively. Concretely, we let $\mu_1 = (-2,-0.5), \sigma_1 = 1.0, \alpha_1 = 0.5$ and $\mu_2 = (2, 0.5), \sigma_2 = 1.0, \alpha_2 = 0.5$. The second dimension of input vector is selected as the sensitive attribute X_s, and the base rate constraint becomes: $\Pr(Y = 0|X_s < 0) = \Pr(Y = 1|X_s > 0)$. Given the perturbation $\delta \in \mathbb{R}^2$ that induces $X \mapsto X + \delta$, the Wasserstein distance and Hellinger distance can be formulated as follows:

$$W_2(\mathcal{P},\mathcal{Q}) = \|\delta\|_2, \quad H(\mathcal{P},\mathcal{Q}) = \sqrt{1 - e^{-\|\delta\|_2^2/8}}. \tag{H.67}$$

H.3.5 More Results of Certified Fairness with Sensitive Shifting and General shifting

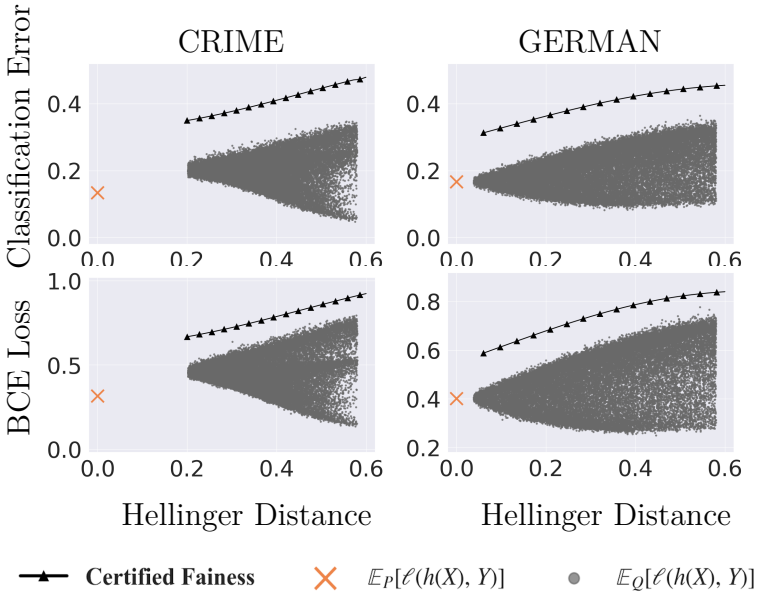


Figure H.1: Certified fairness with sensitive shifting on Crime and German.

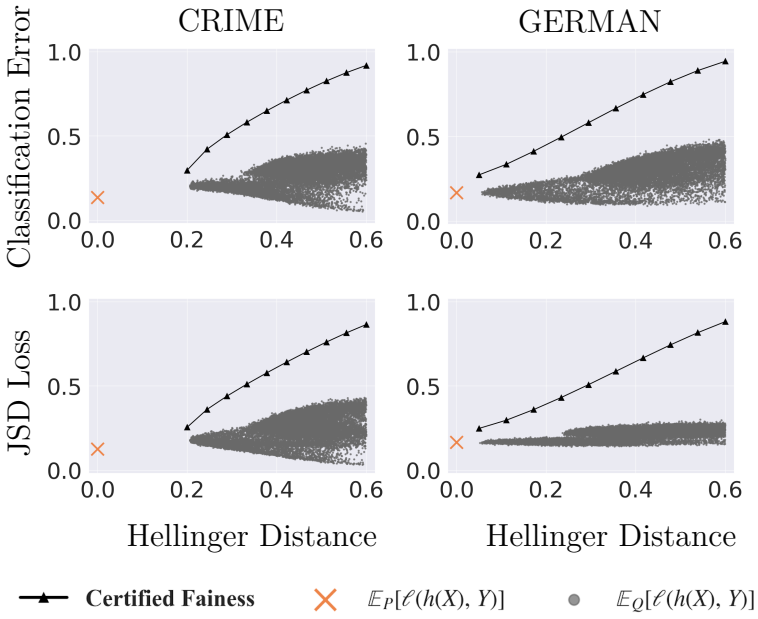


Figure H.2: Certified fairness with general shifting on Crime and German.

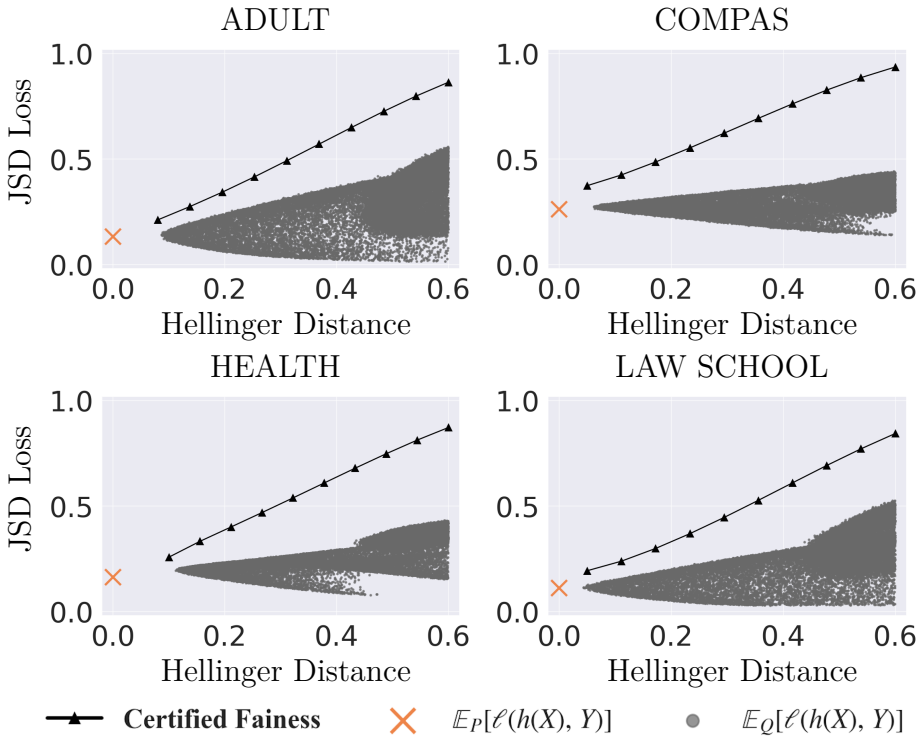


Figure H.3: Certified fairness with general shifting using JSD loss on Adult, Compas, Health, and Lawschool.

H.3.6 More Results of Certified Fairness with Additional Non-Skewness Constraints

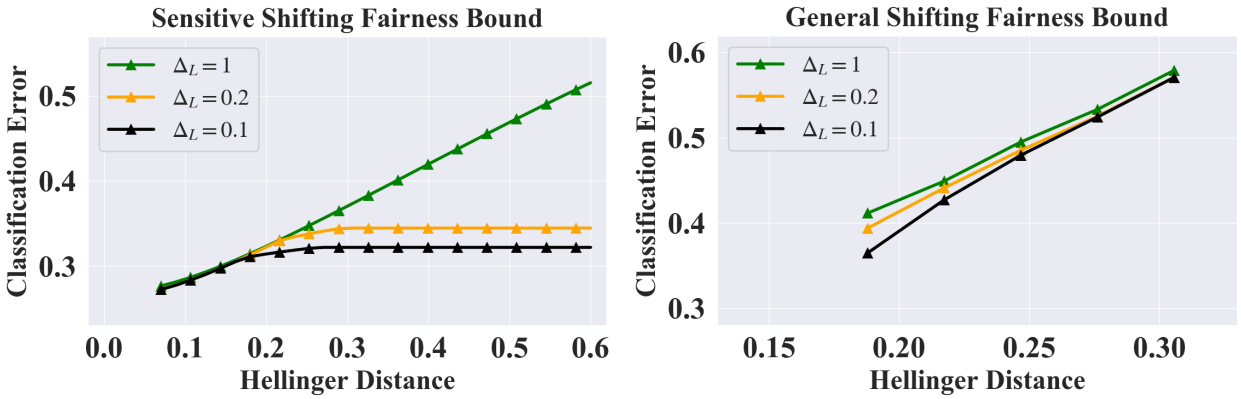


Figure H.4: Certified fairness upper bounds with additional non-skewness constraints of labels on Adult. ($|\Pr_{(\boldsymbol{X},Y)\sim P}[Y=0] - \Pr_{(\boldsymbol{X},Y)\sim P}[Y=1]| \leqslant \Delta_L$)

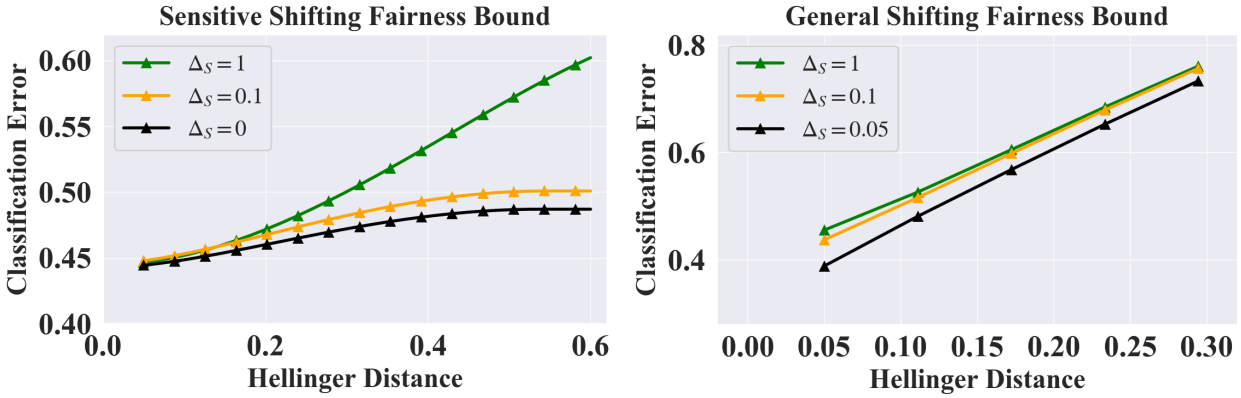


Figure H.5: Certified fairness upper bounds with additional non-skewness constraints of sensitive attributes on Compas. ($|\Pr_{(\boldsymbol{X},Y)\sim P}[X_s=0] - \Pr_{(\boldsymbol{X},Y)\sim P}[X_s=1]| \leqslant \Delta_s$)

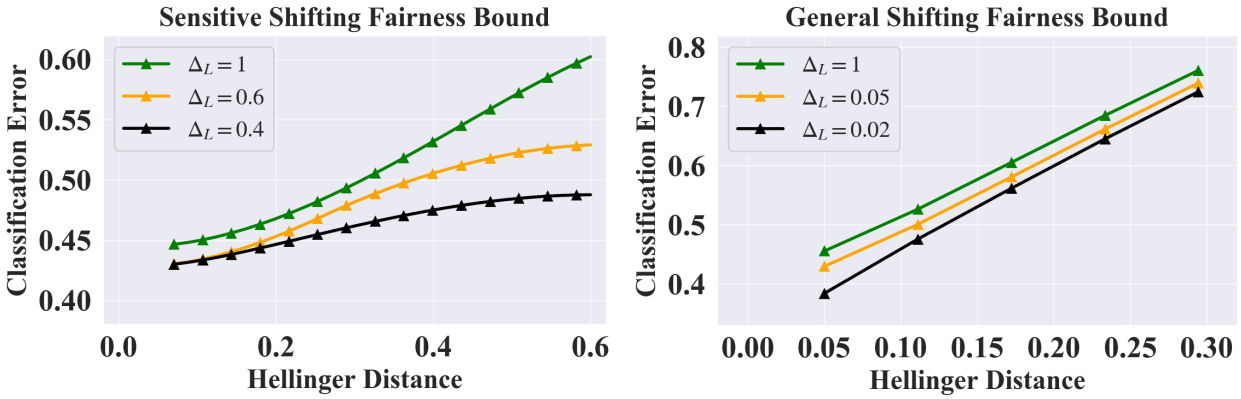


Figure H.6: Certified fairness upper bounds with additional non-skewness constraints of labels on Compas. ($|\Pr_{(\boldsymbol{X},Y)\sim P}[Y=0] - \Pr_{(\boldsymbol{X},Y)\sim P}[Y=1]| \leqslant \Delta_L$)

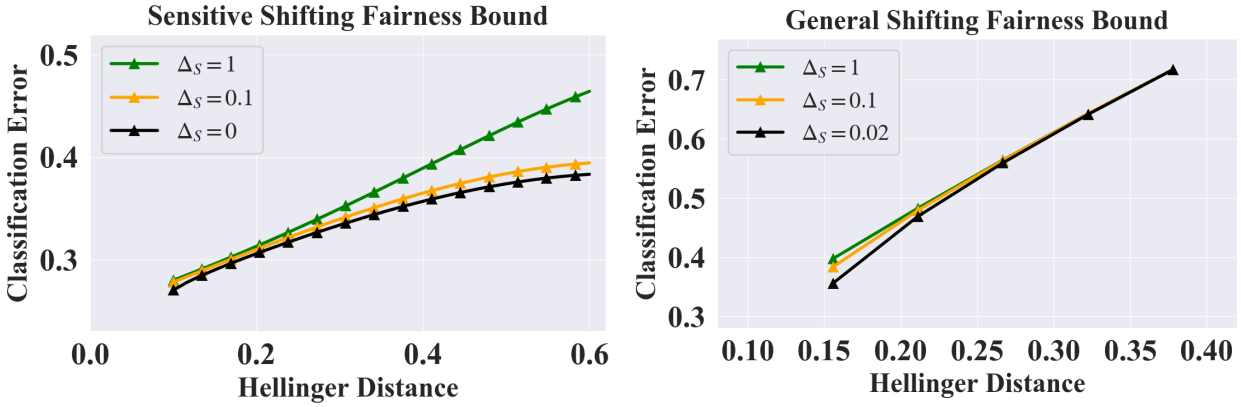


Figure H.7: Certified fairness upper bounds with additional non-skewness constraints of sensitive attributes on Health. ($|\Pr_{(\boldsymbol{X},Y)\sim P}[X_s=0] - \Pr_{(\boldsymbol{X},Y)\sim P}[X_s=1]| \leqslant \Delta_s$)

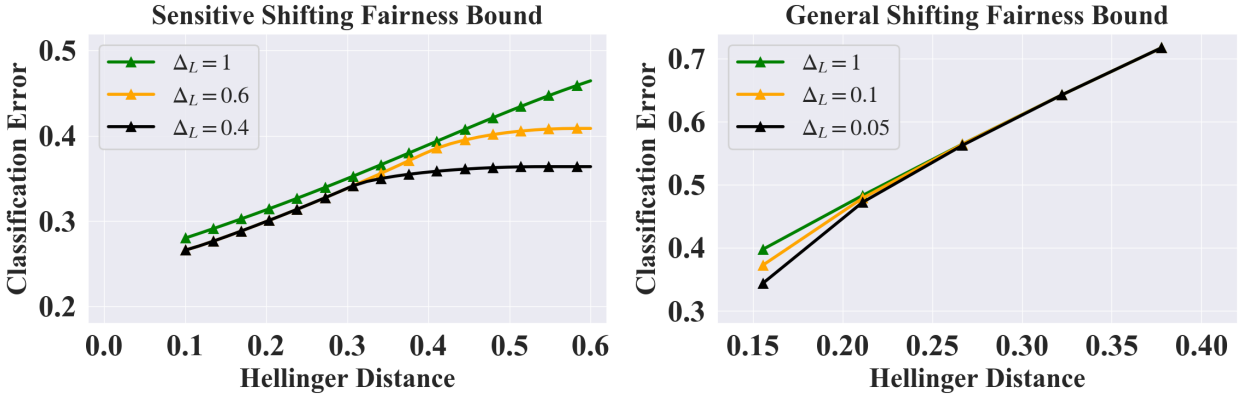


Figure H.8: Certified fairness upper bounds with additional non-skewness constraints of labels on Health. ($|\Pr_{(\boldsymbol{X},Y)\sim P}[Y=0] - \Pr_{(\boldsymbol{X},Y)\sim P}[Y=1]| \leqslant \Delta_L$)

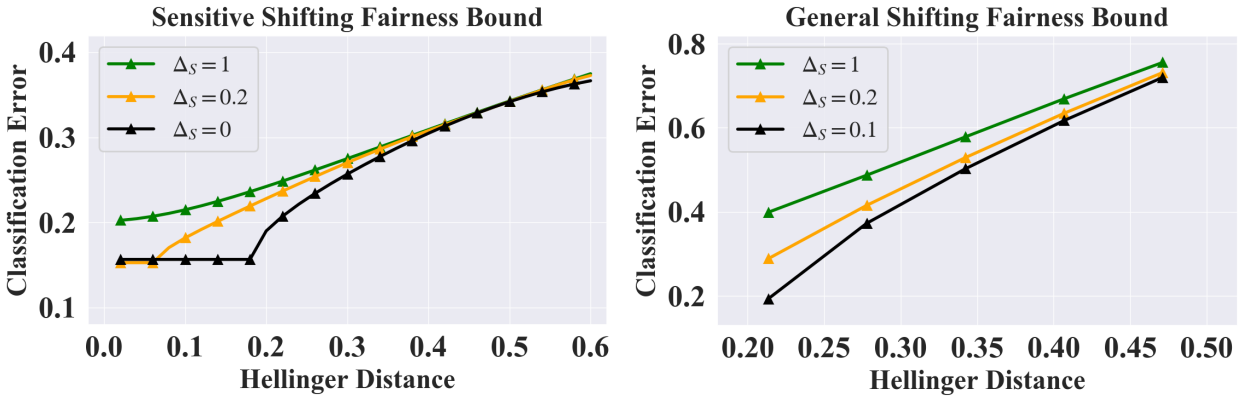


Figure H.9: Certified fairness upper bounds with additional non-skewness constraints of sensitive attributes on Lawschool. ($|\Pr_{(\boldsymbol{X},Y)\sim P}[X_s = 0] - \Pr_{(\boldsymbol{X},Y)\sim P}[X_s = 1]| \leqslant \Delta_s$)

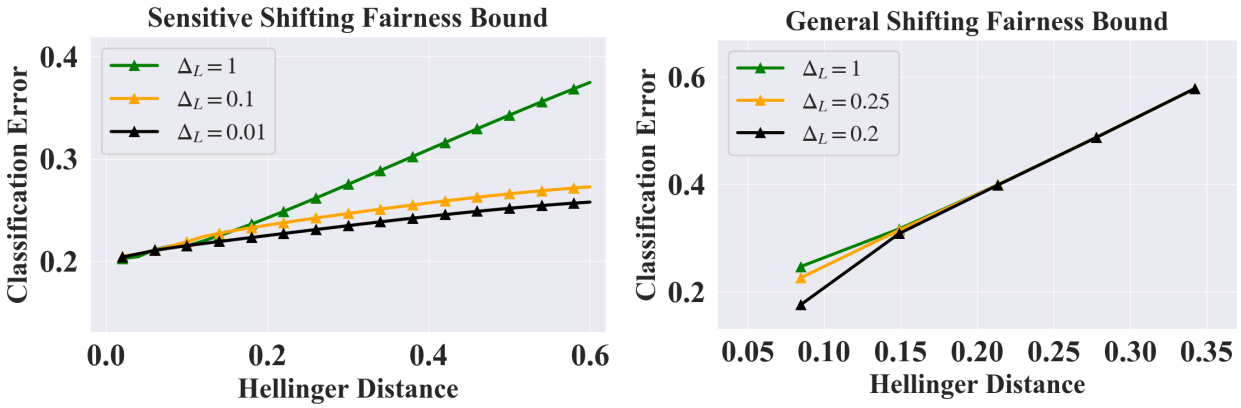


Figure H.10: Certified fairness upper bounds with additional non-skewness constraints of labels on Lawschool. ($|\Pr_{(\boldsymbol{X},Y)\sim P}[Y = 0] - \Pr_{(\boldsymbol{X},Y)\sim P}[Y = 1]| \leqslant \Delta_L$)

H.3.7 Fair Classifier Achieves High Certified Fairness

We compare the fairness certificate of the vanilla model and the perfectly fair model on Adult dataset to demonstrate that our defined certified fairness in Problem 1 and Problem 2 can indicate the fairness in realistic scenarios. In Adult dataset, we have 14 attributes of a person as input and try to predict whether the income of the person is over 50k $/year. The sensitive attribute in Adult is selected as the sex. We consider four subpopulations in the scenario: 1) male with salary below 50k, 2) male with salary above 50k, 3) female with salary below 50k, and 4) female with salary above 50k. We take the overall 0-1 error as the loss. The vanilla model is real, and trained with standard training loss on the Adault dataset. The perfectly fair model is hypothetical and simulated by enforcing the loss within each subpopulation to be the same as the vanilla trained classifier's overall expected loss for fair comparison with the vanilla model. From the experiment results in Table H.1 and Table H.2, we observe that our fairness certificates correlate with the actual fairness level of the model and verify that our certificates can be used as model's fairness indicator: the certified fairness of perfectly fair models are consistently higher than those for the unfair model, for both the general shifting scenario and the sensitive shifting scenario. These findings demonstrate the practicality of our fairness certification.

Table H.1: Comparison of the fairness certificate of the vanilla model (an "unfair" model) and the perfectly fair model (a "fair" model) for sensitive shifting. 0-1 error is selected as the loss in the evaluation.

Hellinger Distance ρ	0.1	0.2	0.3	0.4	0.5
Vanilla Model Fairness Certificate	0.182	0.243	0.297	0.349	0.397
Fair Model Fairness Certificate	0.148	0.148	0.148	0.148	0.148

Table H.2: Comparison of the fairness certificate of the vanilla model (an "unfair" model) and the perfectly fair model (a "fair" model) for general shifting. 0-1 error is selected as the loss in the evaluation.

Hellinger Distance ρ	0.1	0.2	0.3	0.4	0.5
Vanilla Model Fairness Certificate	0.274	0.414	0.559	0.701	0.828
Fair Model Fairness Certificate	0.266	0.407	0.553	0.695	0.824

APPENDIX I: APPENDIX FOR CHAPTER 12

I.1 LIST OF SUPPORTED OPERATORS

In this section, we provide a list of the 84 supported operators in the RANUM static analysis framework. These operators cover common operators used in DNNs. To the best of our knowledge, RANUM provides abstractions for the largest number of operator types among existing DNN abstraction frameworks. We believe that the framework can be further extended to support other types of analysis, testing, and debugging for general DL systems.

In particular, when compared to the state-of-the-art tool DEBAR [460], the underlined 17 operators are newly supported by RANUM.

Sub	Add	Mul	Div	Pow	MatMul
Gemm	MaxPool	GlobalMaxPool	GlobalAveragePool	Cos	Sin
AveragePool	Conv	ConvTranspose	Pad	Reciprocal	Sqrt
Tanh	Relu	Softplus	LeakyRelu	Softsign	Sigmoid
Neg	Exp	Log	Softmax	LogSoftmax	Abs
Ceil	Floor	Sign	Reshape	Flatten	Transpose
Shape	Cast	Slice	Gather	GatherND	Squeeze
Unsqueeze	ScatterElements	Expand	Concat	Split	Identity
ConstantOfShape	RandomNormalLike	RandomUniformLike	RandomNormal	RandomUniform	Range
Constant	Less	LessOrEqual	Greater	GreaterOrEqual	Equal
Not	Or	Min	Max	Clip	Sum
ReduceMin	ReduceMax	ReduceSum	ReduceMean	ArgMax	ArgMin
Tile	NegativeLogLikelihoodLoss	Loop	SequenceInsert	BatchNormalization	OneHot
NonZero	Resize	ReduceProd	ReduceSumSquare	IsInf	IsNaN

Figure I.1: List of supported operators in RANUM.

I.2 LIST OF OPERATORS WITH POTENTIAL NUMERICAL DEFECTS

In this section, we provide a full list of DNN operators that may contain numerical defects, along with their invalid ranges $\mathcal{I}_{n_0,\mathsf{invalid}}$ (see definition of the numerical defect in Definition 12.1), respectively. In the table, $U_{\min}$ and $U_{\max}$ stand for the minimum and maximum positive number of the input tensor's data type, respectively.

Table I.1: List of operators with potential numerical defects.

Op. Type	$\mathcal{I}_{n_0,\mathsf{invalid}}$
Pow	$[-U_{\min}, U_{\min}] \times (-\infty, -U_{\min}]$
Div	$\mathbb{R} \times [-U_{\min}, U_{\min}]$
Reciprocal	$[-U_{\min}, U_{\min}]$
Sqrt	$(-\infty, U_{\min}]$
Exp	$[\ln U_{\max}, \infty)$
Log	$(-\infty, U_{\min}]$
Range	$\mathbb{R} \times \mathbb{R} \times [-U_{\min}, U_{\min}]$
NegativeLogLikelihoodLoss	$[0, 0]$ for number of non-zero cells using mean reduction

I.3 DETAILS OF RANUM STATIC ANALYSIS FRAMEWORK

We present the omitted details in Section 12.2.1.

I.3.1 Abstraction Domain and Characteristics

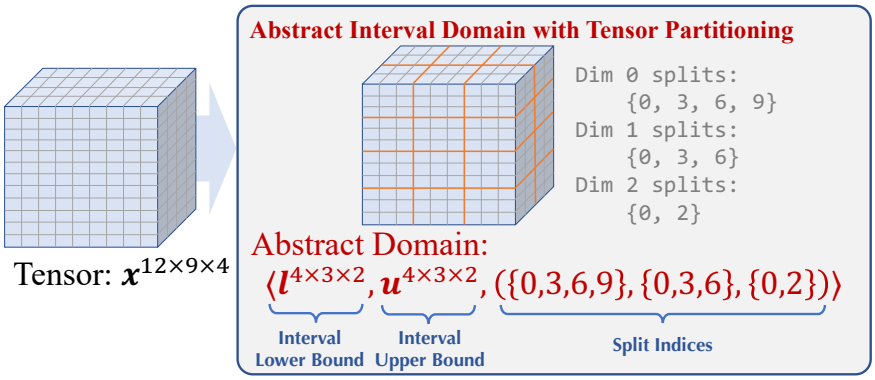


Figure I.2: Example of tensor partitioning. Tensor partitioning reduces the size of tensors in the abstract domain by sharing one interval bound among all elements inside one subblock.

We first formally define our abstraction domain: interval with tensor partitioning. Following the notation in abstract interpretation literature [81], suppose the tensor has m dimensions with shape $n_1 \times n_2 \times \cdots \times n_m$, we define the abstract domain as such:

$$\mathbb{A} := \{\langle \boldsymbol{l}, \boldsymbol{u}, (S_i)_{i=1}^m \rangle : \boldsymbol{l}, \boldsymbol{u} \in \mathbb{R}^{n'_1 \times \cdots \times n'_m} | S_i | = n'_i, \forall s \in S_i, s \in \mathbb{N}, 0 \leqslant s < n_i\}. \tag{I.1}$$

Each element in $\mathbb{A}$ is a triplet $a = \langle \boldsymbol{l}, \boldsymbol{u}, (S_i)_{i=1}^m \rangle$, where the first two elements are subblock-wise interval lower bound and upper bound, respectively, and the last element $(S_1, S_2, \ldots, S_m)$ contains m sets, where each set S_i corresponds to the *0-indexed* split indices for the i-th dimension to form subblocks. We use $S_i[j] \in \mathbb{N}$ to represent the j-th element of split index set S_i sorted in ascending order and define $S_i[|S_i|] = n_i$. Figure I.2 illustrates this abstraction domain.

To define the correspondence between the abstract domain and concrete domain $\mathbb{C} := 2^{\mathbb{R}^{n_1 \times \cdots \times n_m}}$, we form Galois connection $\langle \mathbb{C}, \subseteq \rangle \underset{\gamma}{\overset{\alpha}{\rightleftharpoons}} \langle \mathbb{A}, \sqsubseteq \rangle$, where an abstraction function α :

$\mathbb{C} \to \mathbb{A}$ and the concretization function $\gamma : \mathbb{A} \to \mathbb{C}$ are defined as follows:

$$\begin{aligned}
\alpha(\mathcal{X}) &= \langle \boldsymbol{l}, \boldsymbol{u}, (S_i)_{i=1}^m \rangle \quad \text{where} \\
\boldsymbol{l}_{i_1,i_2,\ldots,i_m} &= \min_{\boldsymbol{x}\in\mathcal{X}} \min_{\substack{1\leqslant k\leqslant m \\ S_k[i_k]\leqslant j_k<S_k[i_k+1]}} \boldsymbol{x}_{j_1,j_2,\ldots,j_m}, \\
\boldsymbol{u}_{i_1,i_2,\ldots,i_m} &= \max_{\boldsymbol{x}\in\mathcal{X}} \max_{\substack{1\leqslant k\leqslant m \\ S_k[i_k]\leqslant j_k<S_k[i_k+1]}} \boldsymbol{x}_{j_1,j_2,\ldots,j_m},
\end{aligned} \tag{I.2a}$$

$$\begin{aligned}
\gamma(\langle \boldsymbol{l}, \boldsymbol{u}, (S_i)_{i=1}^m \rangle) = \{\boldsymbol{x} : &\boldsymbol{l}_{i_1,i_2,\cdots,i_m} \leqslant \boldsymbol{x}_{j_1,j_2,\cdots,j_m} \leqslant \boldsymbol{u}_{i_1,i_2,\cdots,i_m}, \\
&S_k[i_k] \leqslant j_k < S_k[i_k+1], 1 \leqslant k \leqslant m\}.
\end{aligned} \tag{I.2b}$$

In Equation (I.2a), the split indices $(S_i)_{i=1}^m$ can be arbitrarily chosen but need to be kept consistent with those in Equation (I.2b). Take Figure I.2 as the example, to abstract a set of tensors $\mathcal{X}$ with shape $12 \times 9 \times 4$, we define split indices for each dimension, respectively, then impose interval constraints on each subblock of the tensor. For example, $[\boldsymbol{l}_{0,0,0}, \boldsymbol{u}_{0,0,0}]$ constrain any element in $\boldsymbol{x}_{0:2,0:2,0:1}$, $[\boldsymbol{l}_{3,2,1}, \boldsymbol{u}_{3,2,1}]$ constrain any element in $\boldsymbol{x}_{9:11,6:8,2:3}$.

Abstraction Characteristics. There are multiple ways to compute the abstractions, and we design our particular computational algorithms to achieve soundness, tightness, and differentiability.

(1) **Soundness**: We guarantee that all abstractions are sound. Formally, suppose $a_{\boldsymbol{x}}$ and $a_{\boldsymbol{w}}$ are the abstractions for input and weights, respectively, we guarantee $f_n^*(\gamma(a_{\boldsymbol{x}}); \gamma(a_{\boldsymbol{w}})) \subseteq \gamma(T_n^*(a_{\boldsymbol{x}}; a_{\boldsymbol{w}}))$ where $* \in \{\mathsf{in}, \mathsf{out}\}$, n is any node in the architecture, and $T_n^*(a_{\boldsymbol{x}}; a_{\boldsymbol{w}})$ is our computed abstract domain for input or output of node n. The soundness property theoretically guarantees that our detection approach is a certification approach of numerical reliability, i.e., flag all potential numerical defects, and the validity of generated preconditions (see Section 12.1.2).

(2) **Tightness**: For most operators in DNN, given abstractions for its inputs, we compute the tightest possible interval abstraction for the output. Formally, for any atomic DNN operator *op* and any abstract domain $i \in \mathbb{A}$ of *op*'s input, if $op(\gamma(i)) \subseteq [\boldsymbol{l}, \boldsymbol{u}]$, i.e., $[\boldsymbol{l}, \boldsymbol{u}]$ is an interval abstraction of *op*'s output, then $\gamma(T_{op}(i)) \subseteq [\boldsymbol{l}, \boldsymbol{u}]$, i.e., our generated abstract domain $T_{op}(i)$ is always tighter or equal to any interval abstraction $[\boldsymbol{l}, \boldsymbol{u}]$. Such tightness can reduce false positives for detecting potential numerical defects and increase the span of generated failure-exhibiting intervals and fix intervals, which improves the quality of generated tests and preconditions.

(3) **Differentiability**: We compute differentiable abstractions. Concretely, if the input of node n_2 is deferentially dependent on output of node n_1, we can compute out gradi-

ents $\nabla_i o$ for any $i \in \{\boldsymbol{l}_{n_1}, \boldsymbol{u}_{n_1}\}$ and any $o \in \{\boldsymbol{l}_{n_2}, \boldsymbol{u}_{n_2}\}$, where $\langle \boldsymbol{l}_{n_1}, \boldsymbol{u}_{n_1}, (S_i^{n_1})_{i=1}^{\dim(n_1.\mathsf{out})} \rangle$ and $\langle \boldsymbol{l}_{n_2}, \boldsymbol{u}_{n_2}, (S_i^{n_2})_{i=1}^{\dim(n_2.\mathsf{in})} \rangle$ are abstractions of $f_{n_1}^{\mathsf{out}}(\cdot;\cdot)$ and $f_{n_2}^{\mathsf{in}}(\cdot;\cdot)$, respectively. When tightness and differentiability cannot be achieved at the same time (e.g., for `floor` operator, tight abstraction $\langle \boldsymbol{l}, \boldsymbol{u} \rangle \mapsto \langle \lfloor \boldsymbol{l} \rfloor, \lfloor \boldsymbol{u} \rfloor \rangle$ is not generally differentiable, and differentiable abstraction $\langle \boldsymbol{l}, \boldsymbol{u} \rangle \mapsto \langle \boldsymbol{l} - \mathbf{1}, \boldsymbol{u} \rangle$ is not tight), we implement two abstract algorithms to achieve tightness and differentiability, respectively.

The existing approach DEBAR [460] also proposes a static analysis framework for DNN architectures with tensor partitioning. In contrast to our framework, the abstract domain in DEBAR contains affine equalities besides interval domains. Therefore, DEBAR can be tighter than ours in some cases and can produce fewer positives, but due to the additional complexity of affine equalities, DEBAR tends to use the coarsest abstraction (i.e., tensor partitioning) granularity and supports fewer operators than ours. At the same time, our algorithms produce the tightest interval abstractions while DEBAR has no tightness guarantee. As a result, RANUM detects more true numerical defects and has a comparable number of false positives (see Section 12.3.1), and we can leverage the feasibility confirmation support in RANUM to filter out false positives.

I.3.2 Initial Abstraction Construction with **Backward Fine-grained Node Labeling**

We construct abstract domains for initial nodes in two steps: First, we determine the tensor partitions, i.e., the split index sets $(S_i)_{i=1}^m$ (see Equation (I.1)), which determine the tightness and efficiency of our static analysis framework, because the tensor partitions of all other nodes will be solely dependent on the partitions of initial nodes as we will show in Appendix I.3.3. Second, we compute the interval bounds $\boldsymbol{l}$ and $\boldsymbol{u}$.

We use the following principle to decide the tensor partitions: For nodes that are connected to operators requiring fine-grained abstractions (e.g., the `shape` input for operator `Reshape`) with valid paths (which we will specify later), we construct tensor partitions with the finest granularity, i.e., $S_i = \{0, 1, \ldots, n_i\}$. We call these nodes fine-grained initial nodes and starting from the next paragraph we introduce our novel technique of *backward fine-grained node labeling* to find them out. For other nodes, we rely on downstream task requirements and user specifications to determine the partitions. For example, for fix generation (see Section 12.2.3), we use the coarsest granularity by default.

Backward Fine-grained Node Labeling. The fine-grained initial nodes are those starting a valid path in DNN computational graph $\mathcal{G}$, where a path is valid if and only if the path does not traverse through fine-grained stopping operators and terminates at a fine-grained

requiring operator with some specific input indices. As discussed in Section 12.2.1, the fine-grained requiring operators fall into three categories: control-flow operators (e.g., `Loop`), indexing operators (e.g., `Slice`), and shaping operators (e.g., `Reshape`). Fine-grained stopping operators are those whose output is independent of input abstraction granularity (so the granularity of preceding nodes does not matter). Detail lists of fine-grained stopping operators and fine-grained requiring operators are provided in Appendix I.3.4.

To find out fine-grained initial nodes, we propose backward fine-grained node labeling, which is similar to data dependency analysis in traditional programs: First, we invert all edges of the given computational graph $\mathcal{G} = \langle \mathcal{V}, \mathcal{E} \rangle$ to get $\mathcal{G}' = \langle \mathcal{V}, \mathcal{E}' \rangle$. Second, we attach a boolean label for each node $n \in \mathcal{V}$ and initialize them with `False`. Third, we do topology sorting on $\mathcal{G}'$. When encountering a node n with `True`, if the node does not contain a fine-grained stopping operator, we propagate this label to all its subsequent nodes; otherwise, we propagate `False`. When encountering a node n with `False`, if the node is a fine-grained requiring operator, we propagate `True` to some subsequent nodes corresponding to specific input indices and `False` to others; otherwise, propagate `False` to all subsequent nodes. Lastly, we collect all labels for initial nodes. The nodes with `True` label are fine-grained initial nodes.

To this point, we have determined the tensor partitions $(S_i)_{i=1}^m$ for all initial nodes, and we now compute $\boldsymbol{l}$ and $\boldsymbol{u}$ and thus finish the abstraction domain construction for the valid ranges of initial nodes. Initial nodes are further divided into input, weight, and constant nodes. In practice, most weight nodes and all constant nodes have their initial values stored in the ONNX file, and we directly use Equation (I.2a) with these initial values to compute $\boldsymbol{l}$ and $\boldsymbol{u}$. Otherwise, we rely on user specifications and built-in heuristics to determine $\boldsymbol{l}$ and $\boldsymbol{u}$ for initial nodes.

I.3.3 Internal Abstraction with **Dynamic Partitioning**

For all supported operator types (listed in Appendix I.1), we propose concrete algorithms to compute abstract domains for output with dynamic tensor partitioning. Formally, for each operator *op*, we construct the computable function $T_{op} : \mathbb{A} \to \mathbb{A}$ that satisfies soundness and tentatively satisfies tightness and differentiability, where $\mathbb{A}$ is the abstract domain defined in Equation (I.1). Therefore, following the computational procedure introduced at the end of Section 12.2.1, we can compute end-to-end abstractions for all nodes in the given DNN architecture. As the showcase of our abstraction algorithm, we describe the algorithms for four representative operators: `MatMul`, `Conv`, `Softmax`, and `Loop`.

`MatMul`. The `MatMul` operator computes the matrix multiplication of two operands. This operator is widely used in DNNs to express fully-connected layers. To simplify the narration, we focus on the two-dimensional case where we compute $op(\boldsymbol{A}, \boldsymbol{B}) = \boldsymbol{C} = \boldsymbol{AB}$ with $\boldsymbol{A} \in \mathbb{R}^{n\times m}$ and $\boldsymbol{B} \in \mathbb{R}^{m\times l}$. Extensions to other dimensions by abstracting the broadcasting mechanism can be found in our open-source implementation.

We denote the input abstractions of op by $a = \langle \boldsymbol{L}_a, \boldsymbol{U}_a, (S_A^1, S_A^2)\rangle$ and $b = \langle \boldsymbol{L}_b, \boldsymbol{U}_b, (S_B^1, S_B^2)\rangle$, respectively. First, we compute the union $U = S_A^2 \cup S_B^1$. Second, we dynamically partition both a and b with split points (S_A^1, U) and (U, S_B^2), respectively, and get $a' = \langle \boldsymbol{L}'_a, \boldsymbol{U}'_a, (S_A^1, U)\rangle$ and $b' = \langle \boldsymbol{L}'_b, \boldsymbol{U}'_b, (U, S_B^2)\rangle$. Note that a and a' (or b and b') correspond to the same concrete domain, but a' and b' have finer or equal partition granularity than a and b. Third, we compute output abstraction $T_{op}(a, b) = c = \langle \boldsymbol{L}_c, \boldsymbol{U}_c, (S_A^1, S_B^2)\rangle$

$$
\begin{aligned}
\text{where}\quad (\boldsymbol{L}_c)_{ij} &= \sum_{k=1}^{|U|} \boldsymbol{v}_k \min_{\boldsymbol{A}\in\{\boldsymbol{L}'_a,\boldsymbol{U}'_a\},\boldsymbol{B}\in\{\boldsymbol{L}'_b,\boldsymbol{U}'_b\}} \boldsymbol{A}_{ik}\boldsymbol{B}_{kj},\\
(\boldsymbol{U}_c)_{ij} &= \sum_{k=1}^{|U|} \boldsymbol{v}_k \max_{\boldsymbol{A}\in\{\boldsymbol{L}'_a,\boldsymbol{U}'_a\},\boldsymbol{B}\in\{\boldsymbol{L}'_b,\boldsymbol{U}'_b\}} \boldsymbol{A}_{ik}\boldsymbol{B}_{kj},\\
\boldsymbol{v}_k &= U[k] - U[k-1].
\end{aligned}
\tag{I.3}
$$

This formulation can guarantee tightness but is not efficient for tensor computation due to the inner minimum and maximum. Therefore, we also implement a fast-mode abstraction trading tightness for efficiency: $T'_{op}(a, b) = c' = \langle \boldsymbol{L}'_c, \boldsymbol{U}'_c, (S_A^1, S_B^2)\rangle$ where

$$
\begin{aligned}
\boldsymbol{L}'_c =& \boldsymbol{U}'^{-}_a \boldsymbol{v} \boldsymbol{U}'^{-}_b + \boldsymbol{U}'^{0}_a \boldsymbol{v} \boldsymbol{L}'^{-}_b + \boldsymbol{U}'^{+}_a \boldsymbol{v} \boldsymbol{L}'^{-}_b +\\
& \boldsymbol{L}'^{-}_a \boldsymbol{v} \boldsymbol{U}'^{0}_b + \boldsymbol{U}'^{0}_a \boldsymbol{v} \boldsymbol{L}'^{0}_b + \boldsymbol{L}'^{0}_a \boldsymbol{v} \boldsymbol{U}'^{0}_b + \boldsymbol{U}'^{+}_a \boldsymbol{v} \boldsymbol{L}'^{0}_b +\\
& \boldsymbol{L}'^{-}_a \boldsymbol{v} \boldsymbol{U}'^{+}_b + \boldsymbol{L}'^{0}_a \boldsymbol{v} \boldsymbol{U}'^{+}_b + \boldsymbol{L}'^{+}_a \boldsymbol{v} \boldsymbol{L}'^{+}_b,\\
\boldsymbol{U}'_c =& \boldsymbol{L}'^{-}_a \boldsymbol{v} \boldsymbol{L}'^{-}_b + \boldsymbol{L}'^{0}_a \boldsymbol{v} \boldsymbol{L}'^{-}_b + \boldsymbol{L}'^{+}_a \boldsymbol{v} \boldsymbol{U}'^{-}_b +\\
& \boldsymbol{L}'^{-}_a \boldsymbol{v} \boldsymbol{L}'^{0}_b + \boldsymbol{L}'^{0}_a \boldsymbol{v} \boldsymbol{L}'^{0}_b + \boldsymbol{U}'^{0}_a \boldsymbol{v} \boldsymbol{U}'^{0}_b + \boldsymbol{U}'^{+}_a \boldsymbol{v} \boldsymbol{U}'^{0}_b +\\
& \boldsymbol{U}'^{-}_a \boldsymbol{v} \boldsymbol{L}'^{+}_b + \boldsymbol{U}'^{0}_a \boldsymbol{v} \boldsymbol{U}'^{+}_b + \boldsymbol{U}'^{+}_a \boldsymbol{v} \boldsymbol{U}'^{+}_b.
\end{aligned}
\tag{I.4}
$$

In the above equation, for any $* \in \{a, b\}$, let $\circ$ be the elementwise (Hadamard) product,

$$
\begin{aligned}
\boldsymbol{L}'^{-}_* =& \boldsymbol{L}'_* \circ \mathbb{I}[\boldsymbol{U}'_* < 0], \boldsymbol{U}'^{-}_* = \boldsymbol{U}'_* \circ \mathbb{I}[\boldsymbol{U}'_* < 0],\\
\boldsymbol{L}'^{0}_* =& \boldsymbol{L}'_* \circ \mathbb{I}[\boldsymbol{L}'_* \leqslant 0, \boldsymbol{U}'_* \geqslant 0], \boldsymbol{U}'^{0}_* = \boldsymbol{U}'_* \circ \mathbb{I}[\boldsymbol{L}'_* \leqslant 0, \boldsymbol{U}'_* \geqslant 0],\\
\boldsymbol{L}'^{+}_* =& \boldsymbol{L}'_* \circ \mathbb{I}[\boldsymbol{L}'_* > 0], \boldsymbol{U}'^{+}_* = \boldsymbol{U}'_* \circ \mathbb{I}[\boldsymbol{L}'_* > 0].
\end{aligned}
\tag{I.5}
$$

From Equation (I.4), we can observe that the abstraction can be easily implemented with tensor computations. In Appendix I.3.5, we prove the soundness and tightness of these abstractions.

`Conv`. The `Conv` operator computes the discrete convolution of two operands. This operator is widely used in convolutional neural networks (CNNs, [192]). To simplify the narration, we focus on the single-channel single-stride two-dimensional case where we compute $op(\boldsymbol{A}, \boldsymbol{W}) = \boldsymbol{C}$ with $\boldsymbol{A}$ being the input matrix and $\boldsymbol{W}$ being the convolution kernel. Extensions to general cases are provided in our open-source implementation.

We first dynamically split the kernel abstraction to the finest granularity. Second, we compute the receptive field, which is the sub-region of $\boldsymbol{A}$ that decides each output position, of each output position. Third, we inspect the alignment between receptive fields and $\boldsymbol{A}$'s partitions. If neighboring output positions have their receptive fields partitioned in the same sub-block of $\boldsymbol{A}$, it means that these positions can be abstracted by a single interval, i.e., these positions can be partitioned together. Fourth, using this principle, we derive the partitions of the output tensor, and repeat the input tensors accordingly and efficiently so that the abstract computation can be written as convolutional operations. Last, we modify the abstraction computation equations in Equation (I.4) by replacing matrix multiplication with convolution to compute the abstraction of output $\boldsymbol{C}$.

`Softmax`. The `Softmax` operator computes normalized exponential values for the given input. `Softmax` is widely deployed for classification tasks to yield normalized confidence. In one-dimensional case, for input $\boldsymbol{x} \in \mathbb{R}^n$, a `Softmax` operator outputs $op(\boldsymbol{x}) = \frac{\exp(\boldsymbol{x})}{\sum_{i=1}^{n} \exp(\boldsymbol{x})_i}$. The output abstraction of `Softmax` operator op can be thus computed: $T_{op}(\langle \boldsymbol{l}, \boldsymbol{u}, (S) \rangle) = \langle \boldsymbol{l}^o, \boldsymbol{u}^o, (S) \rangle$ where

$$\begin{aligned} \boldsymbol{l}_i^o &= \frac{\exp(\boldsymbol{l}_i)}{\exp(\boldsymbol{u})^\intercal \boldsymbol{v} - \exp(\boldsymbol{u}_i) + \exp(\boldsymbol{l}_i)}, \boldsymbol{u}_i^o = \frac{\exp(\boldsymbol{u}_i)}{\exp(\boldsymbol{l})^\intercal \boldsymbol{v} - \exp(\boldsymbol{l}_i) + \exp(\boldsymbol{u}_i)}, \\ \boldsymbol{v}_k &= S[k] - S[k-1]. \end{aligned} \tag{I.6}$$

The output abstraction's partition is dynamically decided by the input abstraction's partition. We prove the soundness and tightness of the abstraction in Appendix I.3.5.

`Loop`. The node with a `Loop` operator contains a sub-computational graph with dynamic controlling counters and conditions to represent a runtime loop. The `Loop` operator can be used to represent recurrent neural networks (RNNs) and handle the input sequence of variable length. We require the controlling counter and loop termination condition to have

the finest abstraction granularity. Then, we recursively apply our static analysis framework to the sub-graph representing the loop body and update the interval of the loop counter iteratively. When the abstraction interval of the loop counter can explicitly decide whether to terminate the loop, we continue or terminate the loop iterations, respectively; otherwise, we merge the current abstraction interval with the interval obtained after another iteration. We repeat this process until termination. Theoretically, this execution process cannot guarantee the termination, i.e., the loop body may execute for infinite times. However, in practice, on our dataset, our analysis framework already suffices to guarantee the termination in all cases. In the future, we can apply a widening operation if termination cannot be tightly abstracted.

Remark I.1. As we can see, the novel dynamic partition technique is incorporated into the computation process of each operator's abstraction. The soundness and tightness of our designed abstractions are immediately achieved by design, and the differentiability of our designed abstractions can be implemented via the auto-differentiation functionality of popular DL libraries like `PyTorch` [289] and `Tensorflow` [1] where we use `PyTorch` for implementation. The abstractions for DNNs are also implemented for other applications, e.g., for robustness certification as shown in previous chapters. However, our abstractions are tailored for the tensor-partitioned interval domain that is particularly suitable for testing and debugging the numerical defects. To the best of our knowledge, these abstractions are the first that achieve soundness, tightness, and differentiability.

I.3.4 List of Fine-grained Requiring and Stopping Operators

We use fine-grained requiring and fine-grained stopping operators in Appendix I.3.2. Among all supported operators, the fine-grained requiring operators are

`Reshape` (input #2), `Slice` (input #2, #3, #4, #5), `Squeeze` (input #2), `Unsqueeze` (input #2), `Tile` (input #2, #3), `Loop` (input #1, #2), `SequenceInsert` (input #3), `ConstantOfShape` (input #1), `Gather` (input #2), `GatherND` (input #2), `ReduceSum` (input #2), `ScatterElements` (input #2), `Expand` (input #2), `Split` (input #2), `Pad` (input #2, #3), `NegativeLogLikelihoodLoss` (input #2), `Clip` (input #2, #3), `OneHot` (input #2), `Resize` (input #2, #3, #4).

The fine-grained stopping operators are

`Shape`, `RandomNormalLike`, `RandomUniformLike`.

I.3.5 Proofs

Here we present the omitted proofs in Appendix I.3.3.

`MatMul`.

Theorem I.1 (Tightness of Abstraction by Equation (I.3) for `MatMul`)**.** Suppose *op* is the `MatMul` operator and T_{op} is as defined in Equation (I.3), then if $op(\gamma(a), \gamma(b)) \subseteq [\boldsymbol{L}, \boldsymbol{U}]$ for $a, b \in \mathbb{A}$, $\gamma(T_{op}(a, b)) \subseteq [\boldsymbol{L}, \boldsymbol{U}]$.

Proof. We use $\boldsymbol{L}_a$ and $\boldsymbol{U}_a$ to denote the element-wise interval lower and upper bounds of $\gamma(a)$; and use $\boldsymbol{L}_b$ and $\boldsymbol{U}_b$ to denote these bounds of $\gamma(b)$, respectively, where a formal definition is in Equation (I.2b). Then, there exist $\boldsymbol{A}$ and $\boldsymbol{B}$ with $\boldsymbol{L}_a \leqslant \boldsymbol{A} \leqslant \boldsymbol{U}_a$ and $\boldsymbol{L}_b \leqslant \boldsymbol{B} \leqslant \boldsymbol{U}_b$, such that

$$(\boldsymbol{AB})_{ij} = \sum_{k=1}^{l} \min\{\boldsymbol{L}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{L}_{a,ik}\boldsymbol{U}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{U}_{b,kj}\}, \tag{I.7}$$

$$\text{or } (\boldsymbol{AB})_{ij} = \sum_{k=1}^{l} \max\{\boldsymbol{L}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{L}_{a,ik}\boldsymbol{U}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{U}_{b,kj}\}. \tag{I.8}$$

By definition, we have

$$\boldsymbol{L}_{ij} \leqslant \text{Equation (I.7)}, \boldsymbol{U}_{ij} \geqslant \text{Equation (I.8)}. \tag{I.9}$$

We let $\boldsymbol{L}'_c$ and $\boldsymbol{U}'_c$ to denote the element-wise interval lower and upper bounds for $\gamma(T_{op}(a, b))$, let $S^1_A[i'] \leqslant i - 1 < S^1_A[i' + 1]$ and $S^2_B[j'] \leqslant j - 1 < S^2_B[j' + 1]$, then from Equation (I.3),

$$\begin{aligned}
&(\boldsymbol{L}'_c)_{ij} = (\boldsymbol{L}_c)_{i'j'} \\
=&\sum_{k=1}^{|U|} \sum_{k'=U[k-1]+1}^{U[k]} \min\{\boldsymbol{L}_{a,ik'}\boldsymbol{L}_{b,k'j}, \boldsymbol{L}_{a,ik'}\boldsymbol{U}_{b,k'j}, \boldsymbol{U}_{a,ik'}\boldsymbol{L}_{b,k'j}, \boldsymbol{U}_{a,ik'}\boldsymbol{U}_{b,k'j}\} \\
=&\sum_{k=1}^{l} \min\{\boldsymbol{L}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{L}_{a,ik}\boldsymbol{U}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{U}_{b,kj}\} \\
=&\text{Equation (I.7)} \geqslant \boldsymbol{L}_{ij}.
\end{aligned} \tag{I.10}$$

Similarly, $(\boldsymbol{U}'_c)_{ij} \leqslant \boldsymbol{U}_{ij}$. Thus, $\gamma(T_{op}(a, b)) \subseteq [\boldsymbol{L}, \boldsymbol{U}]$. QED.

Theorem I.2 (Soundness of Abstraction by Equation (I.3) for `MatMul`)**.** Suppose *op* is the `MatMul` opeartor and T_{op} is as defined in Equation (I.3), then $op(\gamma(a), \gamma(b)) \subseteq \gamma(T_{op}(a, b))$.

Proof. We use $\boldsymbol{L}_a$ and $\boldsymbol{U}_a$ to denote the element-wise interval lower and upper bounds of $\gamma(a)$; and use $\boldsymbol{L}_b$ and $\boldsymbol{U}_b$ to denote these bounds of $\gamma(b)$, respectively, where a formal definition is in Equation (I.2b). For any $\boldsymbol{A}$ and $\boldsymbol{B}$ such that $\boldsymbol{L}_a \leqslant \boldsymbol{A} \leqslant \boldsymbol{U}_a$ and $\boldsymbol{L}_b \leqslant \boldsymbol{B} \leqslant \boldsymbol{U}_b$,

$$\begin{aligned}(\boldsymbol{AB})_{ij} &= \sum_{k=1}^{l} \boldsymbol{A}_{ik}\boldsymbol{B}_{kj} \\ &\geqslant \min\{\boldsymbol{L}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{L}_{a,ik}\boldsymbol{U}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{U}_{b,kj}\} \\ &=: (\boldsymbol{L}'_{ab})_{ij}\end{aligned} \tag{I.11}$$

and

$$\begin{aligned}(\boldsymbol{AB})_{ij} &\leqslant \max\{\boldsymbol{L}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{L}_{a,ik}\boldsymbol{U}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{U}_{b,kj}\} \\ &=: (\boldsymbol{U}'_{ab})_{ij}.\end{aligned} \tag{I.12}$$

Thus, $op(\gamma(a), \gamma(b)) \subseteq [\boldsymbol{L}'_{ab}, \boldsymbol{U}'_{ab}]$. On the other hand, $\gamma(T_{op}(a, b)) = [\boldsymbol{L}'_{ab}, \boldsymbol{U}'_{ab}]$ as seen from Equation (I.10). Therefore, $op(\gamma(a), \gamma(b)) \subseteq \gamma(T_{op}(a, b))$. QED.

Theorem I.3 (Soundness of Abstraction by Equation (I.4) for `MatMul`)**.** Suppose op is the `MatMul` operator and T'_{op} is as defined in Equations (I.4) and (I.5), then $op(\gamma(a), \gamma(b)) \subseteq \gamma(T'_{op}(a, b))$.

Proof. We use $\boldsymbol{L}_a$ and $\boldsymbol{U}_a$ to denote the element-wise interval lower and upper bounds of $\gamma(a)$; and use $\boldsymbol{L}_b$ and $\boldsymbol{U}_b$ to denote these bounds of $\gamma(b)$, respectively, where a formal definition is in Equation (I.2b). We define $\boldsymbol{L}'_{ab}$ and $\boldsymbol{U}'_{ab}$ by Equations (I.11) and (I.12). From the proof of Theorem I.2, we have $op(\gamma(a), \gamma(b)) \subseteq [\boldsymbol{L}'_{ab}, \boldsymbol{U}'_{ab}]$. We now only need to show that $[\boldsymbol{L}'_{ab}, \boldsymbol{U}'_{ab}] \subseteq \gamma(T'_{op}(a, b))$.

$\gamma(T'_{op}(a, b))$ imposes independent interval abstractions element-wise, therefore, we study each element independently. For the element (i, j), from Equations (I.4) and (I.5), the interval lower bound of $\gamma(T'_{op}(a, b))$, namely $(\boldsymbol{L}_c)_{ij}$, satisfies

$$\begin{aligned}(\boldsymbol{L}_c)_{ij} = \sum_{k=1}^{l} &(\boldsymbol{U}_a^-)_{ik}(\boldsymbol{U}_b^-)_{kj} + (\boldsymbol{U}_a^0)_{ik}(\boldsymbol{L}_b^-)_{kj} + (\boldsymbol{U}_a^+)_{ik}(\boldsymbol{L}_b^-)_{kj} + \\ &(\boldsymbol{L}_a^-)_{ik}(\boldsymbol{U}_b^0)_{kj} + (\boldsymbol{U}_a^0)_{ik}(\boldsymbol{L}_b^0)_{kj} + (\boldsymbol{L}_a^0)_{ik}(\boldsymbol{U}_b^0)_{kj} + (\boldsymbol{U}_a^+)_{ik}(\boldsymbol{L}_b^0)_{kj} + \\ &(\boldsymbol{L}_a^-)_{ik}(\boldsymbol{U}_b^+)_{kj} + (\boldsymbol{L}_a^0)_{ik}(\boldsymbol{U}_b^+)_{kj} + (\boldsymbol{L}_a^+)_{ik}(\boldsymbol{L}_b^+)_{kj}.\end{aligned} \tag{I.13}$$

By Equation (I.5),

- when $\boldsymbol{L}_{a,ik} \leqslant \boldsymbol{U}_{a,ik} < 0$,
  $(\boldsymbol{L}_a^-)_{ik} = \boldsymbol{L}_{a,ik}, (\boldsymbol{U}_a^-)_{ik} = \boldsymbol{U}_{a,ik}, (\boldsymbol{L}_a^0)_{ik} = 0, (\boldsymbol{U}_a^0)_{ik} = 0, (\boldsymbol{L}_a^+)_{ik} = 0, (\boldsymbol{U}_a^+)_{ik} = 0$;

- when $\boldsymbol{L}_{a,ik} \leqslant 0 \leqslant \boldsymbol{U}_{a,ik}$,
  $(\boldsymbol{L}_a^-)_{ik} = 0, (\boldsymbol{U}_a^-)_{ik} = 0, (\boldsymbol{L}_a^0)_{ik} = \boldsymbol{L}_{a,ik}, (\boldsymbol{U}_a^0)_{ik} = \boldsymbol{U}_{a,ik}, (\boldsymbol{L}_a^+)_{ik} = 0, (\boldsymbol{U}_a^+)_{ik} = 0$;
- when $0 < \boldsymbol{L}_{a,ik} \leqslant \boldsymbol{U}_{a,ik}$,
  $(\boldsymbol{L}_a^-)_{ik} = 0, (\boldsymbol{U}_a^-)_{ik} = 0, (\boldsymbol{L}_a^0)_{ik} = 0, (\boldsymbol{U}_a^0)_{ik} = 0, (\boldsymbol{L}_a^+)_{ik} = \boldsymbol{L}_{a,ik}, (\boldsymbol{U}_a^+)_{ik} = \boldsymbol{U}_{a,ik}$.

Similarly for $(\boldsymbol{L}_b^-)_{kj}$, $(\boldsymbol{U}_b^-)_{kj}$, $(\boldsymbol{L}_b^0)_{kj}$, $(\boldsymbol{U}_b^0)_{kj}$, $(\boldsymbol{L}_b^+)_{kj}$ and $(\boldsymbol{U}_b^+)_{kj}$. Thus, by enumerating all cases, we have

$$\text{Equation (I.13)} \leqslant \min\{\boldsymbol{L}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{L}_{a,ik}\boldsymbol{U}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{U}_{b,kj}\} = (\boldsymbol{L}'_{ab})_{ij}. \tag{I.14}$$

Similarly, the interval lower bound of $\gamma(T'_{op}(a,b))$, namely $(\boldsymbol{U}_c)_{ij}$,

$$(\boldsymbol{U}_c)_{ij} \geqslant \max\{\boldsymbol{L}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{L}_{a,ik}\boldsymbol{U}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{L}_{b,kj}, \boldsymbol{U}_{a,ik}\boldsymbol{U}_{b,kj}\} = (\boldsymbol{U}'_{ab})_{ij}. \tag{I.15}$$

Thus, $[\boldsymbol{L}'_{ab}, \boldsymbol{U}'_{ab}] \subseteq \gamma(T'_{op}(a,b))$. QED.

`Softmax`.

Theorem I.4 (Tightness of Abstraction for `Softmax`)**.** Suppose $op : \mathbb{R}^n \to \mathbb{R}^n$ is the `Softmax` operator and T_{op} is as defined in Equation (I.6), if $op(\gamma(\langle \boldsymbol{l}, \boldsymbol{u}, (S)\rangle)) \subseteq [\boldsymbol{l}^r, \boldsymbol{u}^r]$, then $\gamma(T_{op}(\langle \boldsymbol{l}, \boldsymbol{u}, (S)\rangle)) \subseteq [\boldsymbol{l}^r, \boldsymbol{u}^r]$.

Proof. We use $\boldsymbol{l}'$ and $\boldsymbol{u}'$ to denote the element-wise interval lower and upper bounds of $\gamma(\langle \boldsymbol{l}, \boldsymbol{u}, (S)\rangle)$. Formally, $\boldsymbol{l}'_i = \boldsymbol{l}_{i'}$ and $\boldsymbol{u}'_i = \boldsymbol{u}_{i'}$ where $S[i'] \leqslant i - 1 < S[i'+1]$. Then, for each i, by setting $\boldsymbol{x}_i = \boldsymbol{l}_i$ and $\boldsymbol{x}_j = \boldsymbol{u}_j$ for all $j \neq i$, we get

$$\begin{aligned} \boldsymbol{l}_i^r &\leqslant \frac{\exp(\boldsymbol{l}'_i)}{\exp(\boldsymbol{l}'_i) + \sum_{k=1,k\neq i}^{n} \exp(\boldsymbol{u}'_k)} \\ &= \frac{\exp(\boldsymbol{l}_{i'})}{\exp(\boldsymbol{l}_{i'}) + \sum_{k=1}^{n} \exp(\boldsymbol{u}'_k) - \exp(\boldsymbol{u}'_i)} \\ &= \frac{\exp(\boldsymbol{l}_{i'})}{\exp(\boldsymbol{l}_{i'}) + \boldsymbol{v}^\intercal \exp(\boldsymbol{u}_k) - \exp(\boldsymbol{u}_{i'})} = \boldsymbol{l}_{i'}^o. \end{aligned} \tag{I.16}$$

In addition, by setting $\boldsymbol{x}_i = \boldsymbol{u}_i$ and $\boldsymbol{x}_j = \boldsymbol{l}_j$ for all $j \neq i$, we get $\boldsymbol{u}_i^r \geqslant \boldsymbol{u}_{i'}^o$. Here, $\boldsymbol{l}^o$ and $\boldsymbol{u}^o$ are as defined in Equation (I.6). Combining these two arguments, we get

$$\gamma(T_{op}(\langle \boldsymbol{l}, \boldsymbol{u}, (S)\rangle)) \subseteq [\boldsymbol{l}^r, \boldsymbol{u}^r]. \tag{I.17}$$

QED.

Theorem I.5 (Soundness of Abstraction for `Softmax`)**.** Suppose $op : \mathbb{R}^n \to \mathbb{R}^n$ is the `Softmax` operator and T_{op} is as defined in Equation (I.6), then

$$op(\gamma(\langle \boldsymbol{l}, \boldsymbol{u}, (S) \rangle)) \subseteq \gamma(T_{op}(\langle \boldsymbol{l}, \boldsymbol{u}, (S) \rangle)). \tag{I.18}$$

Proof. Leveraging the fact that the softmax function is monotonically increasing, i.e., $\frac{\mathrm{d}op(\boldsymbol{x})_i}{\mathrm{d}\boldsymbol{x}_j} > 0$, we have $\gamma(\langle \boldsymbol{l}, \boldsymbol{u}, (S) \rangle)_i \in [\boldsymbol{l}^o_{i'}, \boldsymbol{u}^o_{i'}]$, where $S[i'] \leqslant i - 1 < S[i' + 1]$. Since $[\boldsymbol{l}^o_{i'}, \boldsymbol{u}^o_{i'}] = \gamma(T_{op}(\langle \boldsymbol{l}, \boldsymbol{u}, (S) \rangle))_i$ by our definition of T_{op}, $op(\gamma(\langle \boldsymbol{l}, \boldsymbol{u}, (S) \rangle)) \subseteq \gamma(T_{op}(\langle \boldsymbol{l}, \boldsymbol{u}, (S) \rangle))$ follows. QED.

REFERENCES

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[2] M. Abramowitz and I. A. E. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, 9th printing*. New York: Dover, New York, 1972.

[3] R. Agarwal, D. Schuurmans, and M. Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, 2020.

[4] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[5] A. Albarghouthi, L. D'Antoni, S. Drews, and A. V. Nori. Fairsquare: probabilistic verification of program fairness. *Proceedings of the ACM on Programming Languages*, 1(OOPSLA):1–30, 2017.

[6] M. Alfarra, A. Bibi, P. H. Torr, and B. Ghanem. Data dependent randomized smoothing. *arXiv preprint arXiv:2012.04351*, 2020.

[7] S. Almahdi and S. Y. Yang. An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. *Expert Systems with Applications*, 87:267–279, 2017.

[8] G. Anderson, S. Pailoor, I. Dillig, and S. Chaudhuri. Optimization and abstraction: a synergistic approach for analyzing neural network robustness. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 731–744, 2019.

[9] R. Anderson, J. Huchette, C. Tjandraatmadja, and J. P. Vielma. Strong convex relaxations and mixed-integer programming formulations for trained neural networks. In *Mathematical Programming*, 2020.

[10] M. Andriushchenko and M. Hein. Provably robust boosted decision stumps and trees against adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 12997–13008, 2019.

[11] J. Angwin, J. Larson, S. Mattu, and L. Kirchner. Machine bias. In *Ethics of Data and Analytics*, pages 254–264. Auerbach Publications, 2016.

[12] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7163–7172, 2019.

[13] H. Ashtiani, V. Pathak, and R. Urner. Black-box certification and learning under adversarial perturbations. In *International Conference on Machine Learning*, pages 388–398. PMLR, 2020.

[14] A. Asuncion and D. Newman. Uci machine learning repository, 2007.

[15] A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283, 2018.

[16] P. Awasthi, H. Jain, A. S. Rawat, and A. Vijayaraghavan. Adversarial robustness via robust low rank representations. *Advances in Neural Information Processing Systems*, 33:11391–11403, 2020.

[17] M. Baader, M. Mirman, and M. Vechev. Universal approximation with certified networks. In *International Conference on Learning Representations*, 2020.

[18] S. Bak. nnenum: Verification of relu neural networks with optimized abstraction refinement. In *NASA Formal Methods Symposium*, pages 19–36. Springer, 2021.

[19] S. Bak, H.-D. Tran, K. Hobbs, and T. T. Johnson. Improved geometric path enumeration for verifying relu neural networks. In *International Conference on Computer Aided Verification*, pages 66–96. Springer, 2020.

[20] S. Bak, C. Liu, and T. Johnson. The second international verification of neural networks competition (vnn-comp 2021): Summary and results. *arXiv preprint arXiv:2109.00498*, 2021.

[21] M. Balunovic and M. Vechev. Adversarial training and provable defenses: Bridging the gap. In *International Conference on Learning Representations*, 2020.

[22] M. Balunovic, M. Baader, G. Singh, T. Gehr, and M. Vechev. Certifying geometric robustness of neural networks. In *2019 Advances in Neural Information Processing Systems (NeurIPS)*, pages 15287–15297. Curran Associates, Inc., 2019.

[23] M. Balunovic, A. Ruoss, and M. Vechev. Fair normalizing flows. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=BrFIKuxrZE`.

[24] K. Banihashem, A. Singla, and G. Radanovic. Defense against reward poisoning attacks in reinforcement learning. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL `https://openreview.net/forum?id=goPsLn3RVo`.

[25] S. Barocas and A. D. Selbst. Big data's disparate impact. *Calif. L. Rev.*, 104:671, 2016.

[26] O. Bastani, X. Zhang, and A. Solar-Lezama. Probabilistic verification of fairness properties via concentration. *Proceedings of the ACM on Programming Languages*, 3 (OOPSLA):1–27, 2019.

[27] B. Batten, P. Kouvaros, A. Lomuscio, and Y. Zheng. Efficient neural network verification via layer-based semidefinite relaxations and linear cuts. In *International Joint Conference on Artificial Intelligence*, pages 2184–2190, 2021.

[28] V. Behzadan and A. Munir. Vulnerability of deep reinforcement learning to policy induction attacks. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 262–275. Springer, 2017.

[29] V. Behzadan and A. Munir. Whatever does not kill deep reinforcement learning, makes it stronger. *arXiv preprint arXiv:1712.09344*, 2017.

[30] V. Behzadan and A. Munir. Mitigation of policy manipulation attacks on deep q-networks with parameter-space noise. In *International Conference on Computer Safety, Reliability, and Security*, pages 406–417. Springer, 2018.

[31] M. G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pages 449–458. PMLR, 2017.

[32] R. Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[33] A. Ben-Tal, D. Den Hertog, A. De Waegenaere, B. Melenberg, and G. Rennen. Robust solutions of optimization problems affected by uncertain probabilities. *Management Science*, 59(2):341–357, 2013.

[34] Y. Bengio, N. Léonard, and A. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[35] R. Berk, H. Heidari, S. Jabbari, M. Kearns, and A. Roth. Fairness in criminal justice risk assessments: The state of the art. *Sociological Methods & Research*, 50(1):3–44, 2021.

[36] D. Bertsimas and R. Weismantel. *Optimization over integers*. Athena Scientific, 2005. ISBN 978-0-97591-462-5.

[37] R. Bixby and E. Rothberg. Progress in computational mixed integer programming—a look back from the other side of the tipping point. *Annals of Operations Research*, 149 (1):37–41, 2007.

[38] A. Blum, T. Dick, N. Manoj, and H. Zhang. Random smoothing might be unable to certify ℓ_∞ robustness for high-dimensional images. *Journal of Machine Learning Research (JMLR)*, 21(211):1–21, 2020.

[39] G. Bonaert, D. I. Dimitrov, M. Baader, and M. Vechev. Fast and precise certification of transformers. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 466–481, 2021.

[40] A. Boopathy, T.-W. Weng, P.-Y. Chen, S. Liu, and L. Daniel. Cnn-cert: An efficient framework for certifying robustness of convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3240–3247, 2019.

[41] E. Botoeva, P. Kouvaros, J. Kronqvist, A. Lomuscio, and R. Misener. Efficient verification of relu-based neural networks via dependency analysis. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[42] W. Brendel, J. Rauber, and M. Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=SyZI0GWCZ`.

[43] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[44] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.

[45] S. Bubeck and M. Sellke. A universal law of robustness via isoperimetry. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[46] S. Bulusu, B. Kailkhura, B. Li, P. K. Varshney, and D. Song. Anomalous example detection in deep learning: A survey. *IEEE Access*, 8:132330–132347, 2020.

[47] R. Bunel, A. De Palma, A. Desmaison, K. Dvijotham, P. Kohli, P. H. S. Torr, and M. P. Kumar. Lagrangian decomposition for neural network verification. *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2020.

[48] R. Bunel, P. Mudigonda, I. Turkaslan, P. Torr, J. Lu, and P. Kohli. Branch and bound for piecewise linear neural network verification. *Journal of Machine Learning Research*, 21(2020), 2020.

[49] R. R. Bunel, I. Turkaslan, P. Torr, P. Kohli, and P. K. Mudigonda. A unified view of piecewise linear neural network verification. In *Advances in Neural Information Processing Systems*, pages 4790–4799, 2018.

[50] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 2267–2281, 2019.

[51] Y. Cao, N. Wang, C. Xiao, D. Yang, J. Fang, R. Yang, Q. A. Chen, M. Liu, and B. Li. Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 176–194. IEEE, 2021.

[52] N. Carlini. Is AmI (attacks meet interpretability) robust to adversarial examples? *arXiv preprint arXiv:1902.02322*, 2019.

[53] N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *ACM Workshop on Artificial Intelligence and Security*, pages 3–14, 2017.

[54] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[55] N. Carlini, F. Tramer, K. D. Dvijotham, L. Rice, M. Sun, and J. Z. Kolter. (certified!!) adversarial robustness for free! In *International Conference on Learning Representations*, 2023.

[56] Y. Carmon, A. Raghunathan, L. Schmidt, J. C. Duchi, and P. S. Liang. Unlabeled data improves adversarial robustness. In *Advances in Neural Information Processing Systems*, pages 11190–11201, 2019.

[57] H. Chacon, S. Silva, and P. Rad. Deep learning poison data attack detection. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 971–978. IEEE, 2019.

[58] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[59] H. Chen, H. Zhang, S. Si, Y. Li, D. Boning, and C.-J. Hsieh. Robustness verification of tree-based models. In *Advances in Neural Information Processing Systems*, pages 12317–12328, 2019.

[60] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

[61] S. Chen, B. Liu, C. Feng, C. Vallespi-Gonzalez, and C. Wellington. 3D point cloud processing and learning for autonomous driving: Impacting map creation, localization, and perception. *IEEE Signal Processing Magazine*, 38(1):68–86, 2020.

[62] S. Chen, E. Wong, J. Z. Kolter, and M. Fazlyab. Deepsplit: Scalable verification of deep neural networks via operator splitting. *IEEE Open Journal of Control Systems*, 1:126–140, 2022.

[63] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1907–1915, 2017.

[64] Y. Chen, S. Wang, Y. Qin, X. Liao, S. Jana, and D. Wagner. Learning security classifiers with verified global robustness properties. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, page 477–494, 2021. ISBN 9781450384544.

[65] Y. Chen, R. Raab, J. Wang, and Y. Liu. Fairness transferability subject to bounded distribution shift. *Advances in Neural Information Processing Systems*, 2022.

[66] C.-H. Cheng, G. Nührenberg, and H. Ruess. Maximum resilience of artificial neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 251–268. Springer, 2017.

[67] H. Cheng, K. Xu, C. Wang, X. Lin, B. Kailkhura, and R. Goldhahn. Mixture of robust experts (more): A flexible defense against multiple perturbations. *arXiv preprint arXiv:2104.10586*, 2021.

[68] R. Cheng, G. Orosz, R. M. Murray, and J. W. Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3387–3395, 2019.

[69] H. Chernoff and H. Scheffe. A generalization of the neyman-pearson fundamental lemma. *The Annals of Mathematical Statistics*, 23(4):213–225, 1952.

[70] P.-y. Chiang, M. Curry, A. Abdelkader, A. Kumar, J. Dickerson, and T. Goldstein. Detection as regression: Certified object detection with median smoothing. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1275–1286, 2020.

[71] Y. Choi, M. Dang, and G. Van den Broeck. Group fairness by probabilistic modeling with latent fair decisions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12051–12059, 2021.

[72] F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[73] A. Chouldechova. Fair prediction with disparate impact: A study of bias in recidivism prediction instruments. *Big data*, 5(2):153–163, 2017.

[74] P. Christiano, Z. Shah, I. Mordatch, J. Schneider, T. Blackwell, J. Tobin, P. Abbeel, and W. Zaremba. Transfer from simulation to real world through learning deep inverse dynamics model. *arXiv preprint arXiv:1610.03518*, 2016.

[75] W. Chu, L. Li, and B. Li. TPC: Transformation-specific smoothing for point cloud models. In *Proceedings of the 39th International Conference on Machine Learning*, pages 4035–4056, 2022.

[76] V. Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete mathematics*, 4(4):305–337, 1973.

[77] J. Cohen, E. Rosenfeld, and Z. Kolter. Certified adversarial robustness via randomized smoothing. In *2019 International Conference on Machine Learning (ICML)*, pages 1310–1320. PMLR, 2019.

[78] M. Conforti, G. Cornuéjols, and G. Zambelli. *Integer programming / Michele Conforti, Gérard Cornuéjols, Giacomo Zambelli.* Graduate texts in mathematics, . 271. Springer, Cham, 2014. ISBN 9783319110073.

[79] T. Contributors. torch.onnx — pytorch 1.10 documentation. `https://pytorch.org/docs/stable/onnx.html`, 2022. Accessed: 2023-02-01.

[80] P. Cousot and R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Proceedings of the 4th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 238–252, 1977.

[81] P. Cousot and R. Cousot. Static determination of dynamic properties of generalized type unions. In *ACM Conference on Language Design for Reliable Software, LDRS*, pages 77–94. ACM, 1977. doi: 10.1145/800022.808314.

[82] E. Creager, D. Madras, J.-H. Jacobsen, M. Weis, K. Swersky, T. Pitassi, and R. Zemel. Flexibly fair representation learning by disentanglement. In *International conference on machine learning*, pages 1436–1445. PMLR, 2019.

[83] F. Croce and M. Hein. Provable robustness against all adversarial l_p-perturbations for $p \geqslant 1$. In *International Conference on Learning Representations*, 2020.

[84] F. Croce, M. Andriushchenko, and M. Hein. Provable robustness of relu networks via maximization of linear regions. In K. Chaudhuri and M. Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 2057–2066. PMLR, 16–18 Apr 2019.

[85] F. Croce, M. Andriushchenko, and M. Hein. Provable robustness of relu networks via maximization of linear regions. In *the 22nd International Conference on Artificial Intelligence and Statistics*, pages 2057–2066. PMLR, 2019.

[86] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. Robustbench: a standardized adversarial robustness benchmark. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[87] H. Crowder, E. L. Johnson, and M. Padberg. Solving large-scale zero-one linear programming problems. *Operations Research*, 31(5):803–834, 1983.

[88] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos. Distributional reinforcement learning with quantile regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[89] S. Dathathri, K. Dvijotham, A. Kurakin, A. Raghunathan, J. Uesato, R. R. Bunel, S. Shankar, J. Steinhardt, I. Goodfellow, P. S. Liang, and P. Kohli. Enabling certification of verification-agnostic networks via memory-efficient semidefinite programming. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5318–5331, 2020.

[90] A. Datta, M. C. Tschantz, and A. Datta. Automated experiments on ad privacy settings. *Proceedings on privacy enhancing technologies*, 2015(1):92–112, 2015.

[91] A. De Palma, R. Bunel, A. Desmaison, K. Dvijotham, P. Kohli, P. H. Torr, and M. P. Kumar. Improved branch and bound for neural network verification via lagrangian decomposition. *arXiv preprint arXiv:2104.06718*, 2021.

[92] M. P. Deisenroth, G. Neumann, J. Peters, et al. A survey on policy search for robotics. *Foundations and trends in Robotics*, 2(1-2):388–403, 2013.

[93] A. Demontis, M. Melis, M. Pintor, M. Jagielski, B. Biggio, A. Oprea, C. Nita-Rotaru, and F. Roli. Why do adversarial attacks transfer? explaining transferability of evasion and poisoning attacks. In *28th {USENIX} Security Symposium ({USENIX} Security 19)*, pages 321–338, 2019.

[94] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[95] Y. Deng, F. Bao, Y. Kong, Z. Ren, and Q. Dai. Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems*, 28(3):653–664, 2016.

[96] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[97] I. Diakonikolas, G. Kamath, D. M. Kane, J. Li, A. Moitra, and A. Stewart. Robust estimators in high dimensions without the computational intractability. In *57th Annual Symposium on Foundations of Computer Science*, pages 655–664. Institute of Electrical and Electronics Engineers (IEEE), 2016.

[98] P. L. Donti, M. Roderick, M. Fazlyab, and J. Z. Kolter. Enforcing robust control guarantees within neural network policies. *arXiv preprint arXiv:2011.08105*, 2020.

[99] T. Du, S. Ji, L. Shen, Y. Zhang, J. Li, J. Shi, C. Fang, J. Yin, R. Beyah, and T. Wang. Cert-rnn: Towards certifying the robustness of recurrent neural networks. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 516–534, 2021.

[100] J. Duchi and H. Namkoong. Variance-based regularization with convex objectives. *The Journal of Machine Learning Research*, 20(1):2450–2504, 2019.

[101] J. C. Duchi, P. W. Glynn, and H. Namkoong. Statistics of robust optimization: A generalized empirical likelihood approach. *Mathematics of Operations Research*, 2021.

[102] S. Dutta, S. Jha, S. Sankaranarayanan, and A. Tiwari. Output range analysis for deep feedforward neural networks. In *NASA Formal Methods - 10th International Symposium*, volume 10811, pages 121–138, 2018.

[103] K. Dvijotham, S. Gowal, R. Stanforth, R. Arandjelovic, B. O'Donoghue, J. Uesato, and P. Kohli. Training verified learners with learned verifiers. *arXiv preprint arXiv:1805.10265*, 2018.

[104] K. Dvijotham, R. Stanforth, S. Gowal, T. A. Mann, and P. Kohli. A dual approach to scalable verification of deep networks. In *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence*, volume 1, pages 550–559, 2018.

[105] K. D. Dvijotham, R. Stanforth, S. Gowal, C. Qin, S. De, and P. Kohli. Efficient neural network verification with exactness characterization. In *Proc. Uncertainty in Artificial Intelligence, UAI*, page 164, 2019.

[106] K. D. Dvijotham, J. Hayes, B. Balle, J. Z. Kolter, C. Qin, A. György, K. Xiao, S. Gowal, and P. Kohli. A framework for robustness certification of smoothed classifiers using f-divergences. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[107] H. Edwards and A. Storkey. Censoring representations with an adversary. *arXiv preprint arXiv:1511.05897*, 2015.

[108] R. Ehlers. Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 269–286. Springer, 2017.

[109] F. Eiras, M. Alfarra, M. P. Kumar, P. H. S. Torr, P. K. Dokania, B. Ghanem, and A. Bibi. ANCER: anisotropic certification via sample-wise volume maximization. *arXiv preprint arXiv:2107.04570*, 2021.

[110] L. Engstrom, B. Tran, D. Tsipras, L. Schmidt, and A. Madry. Exploring the landscape of spatial robustness. In *2019 International Conference on Machine Learning (ICML)*, pages 1802–1811. PMLR, 2019.

[111] M. Everett, B. Lütjens, and J. P. How. Certifiable robustness to adversarial state uncertainty in deep reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[112] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR 2018)*, pages 1625–1634, 2018.

[113] B. Eysenbach and S. Levine. Maximum entropy rl (provably) solves some robust rl problems. *arXiv preprint arXiv:2103.06257*, 2021.

[114] M. Fazlyab, M. Morari, and G. J. Pappas. Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming. *IEEE Transactions on Automatic Control*, 2020.

[115] C. Ferrari, M. N. Mueller, N. Jovanović, and M. Vechev. Complete verification via multi-neuron relaxation guided branch-and-bound. In *International Conference on Learning Representations*, 2022.

[116] J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 64(7):2737–2752, 2018.

[117] M. Fischer, M. Mirman, S. Stalder, and M. Vechev. Online robustness training for deep reinforcement learning. *arXiv preprint arXiv:1911.00887*, 2019.

[118] M. Fischer, M. Baader, and M. Vechev. Certified defense to image transformations via randomized smoothing. In *2020 Advances in Neural Information Processing Systems (NeurIPS)*, pages 8404–8417. Curran Associates, Inc., 2020.

[119] M. Fischer, M. Baader, and M. Vechev. Certified defense to image transformations via randomized smoothing. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 8404–8417, 2020.

[120] M. Fischer, M. Baader, and M. Vechev. Scalable certified segmentation via randomized smoothing. In *International Conference on Machine Learning*, pages 3340–3351. PMLR, 2021.

[121] M. Fortunato, M. G. Azar, B. Piot, J. Menick, I. Osband, A. Graves, V. Mnih, R. Munos, D. Hassabis, O. Pietquin, et al. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295*, 2017.

[122] A. Fromherz, K. Leino, M. Fredrikson, B. Parno, and C. Pasareanu. Fast geometric projections for local robustness certification. In *International Conference on Learning Representations*, 2021.

[123] T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev. AI^2: Safety and robustness certification of neural networks with abstract interpretation. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 3–18. IEEE, 2018.

[124] A. Ghiasi, A. Shafahi, and T. Goldstein. Breaking certified defenses: Semantic adversarial examples with spoofed robustness certificates. In *2020 International Conference on Learning Representations (ICLR)*. OpenReview, 2020.

[125] P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859, 1961.

[126] P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting stock problem—part ii. *Operations research*, 11(6):863–888, 1963.

[127] A. Gleave, M. Dennis, C. Wild, N. Kant, S. Levine, and S. Russell. Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615*, 2019.

[128] J. A. Gliner. Reviewing qualitative research: Proposed criteria for fairness and rigor. *The Occupational Therapy Journal of Research*, 14(2):78–92, 1994.

[129] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *14th International Conference on Artificial Intelligence and Statistics, AISTATS*, pages 315–323. JMLR.org, 2011.

[130] T. Gokhale, R. Anirudh, B. Kailkhura, J. J. Thiagarajan, C. Baral, and Y. Yang. Attribute-guided adversarial training for robustness to natural perturbations. In *2021 AAAI Conference on Artificial Intelligence (AAAI)*, volume 35, pages 7574–7582. Association for the Advancement of Artificial Intelligence Press, 2021.

[131] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[132] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[133] L. A. Goodman. On simultaneous confidence intervals for multinomial proportions. *Technometrics*, 7(2):247–254, 1965.

[134] S. Gowal, K. D. Dvijotham, R. Stanforth, R. Bunel, C. Qin, J. Uesato, R. Arandjelovic, T. Mann, and P. Kohli. Scalable verified training for provably robust image classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4842–4851, 2019.

[135] I. C. S. S. C. W. group of the Microprocessor Standards Subcommittee and A. N. S. Institute. *IEEE standard for binary floating-point arithmetic*, volume 754. IEEE, 1985.

[136] A. Gurfinkel, T. Kahsai, A. Komuravelli, and J. A. Navas. The SeaHorn verification framework. In *27th International Conference on Computer Aided Verification, CAV*, pages 343–361. Springer, 2015.

[137] L. Gurobi Optimization. Gurobi optimizer - gurobi, 2022. URL `https://www.gurobi.com/products/gurobi-optimizer/`.

[138] H. Han, K. Xu, X. Hu, X. Chen, L. Liang, Z. Du, Q. Guo, Y. Wang, and Y. Chen. Scalecert: Scalable certified defense against adversarial patches with sparse superficial layers. *Advances in Neural Information Processing Systems*, 34, 2021.

[139] M. Hardt, E. Price, and N. Srebro. Equality of opportunity in supervised learning. *Advances in neural information processing systems*, 29, 2016.

[140] M. Hattori, S. Sawada, S. Hamaji, M. Sakai, and S. Shimizu. Semi-static type, shape, and symbolic shape inference for dynamic computation graphs. In *4th ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*, pages 11–19, 2020.

[141] J. Hayes. Extensions and limitations of randomized smoothing for robustness guarantees. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3413–3421. IEEE, 2020.

[142] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[143] M. Hein and M. Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *Advances in Neural Information Processing Systems*, pages 2266–2276, 2017.

[144] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *2018 International Conference on Learning Representations (ICLR)*. OpenReview, 2018.

[145] D. Hendrycks and T. Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=HJz6tiCqYm`.

[146] D. Hendrycks, N. Mu, E. D. Cubuk, B. Zoph, J. Gilmer, and B. Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *2020 International Conference on Learning Representations (ICLR)*. OpenReview, 2020.

[147] D. Hendrycks, K. Zhao, S. Basart, J. Steinhardt, and D. Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262–15271, 2021.

[148] P. Henriksen and A. Lomuscio. Efficient neural network verification via adaptive refinement and adversarial search. In *ECAI 2020*, pages 2513–2520. IOS Press, 2020.

[149] P. Henriksen and A. Lomuscio. Deepsplit: An efficient splitting method for neural network verification via indirect effect analysis. In *Proceedings of the 30th international joint conference on artificial intelligence (IJCAI21)*, pages 2549–2555, 2021.

[150] W. Hoeffding. Probability inequalities for sums of bounded random variables. In *The Collected Works of Wassily Hoeffding*, pages 409–426. Springer, 1994.

[151] K. L. Hoffman and M. Padberg. Solving airline crew scheduling problems by branch-and-cut. *Management science*, 39(6):657–682, 1993.

[152] M. Z. Horváth, M. N. Mueller, M. Fischer, and M. Vechev. Boosting randomized smoothing with variance reduced classifiers. In *International Conference on Learning Representations*, 2022.

[153] H. Hosseini and R. Poovendran. Semantic adversarial examples. In *2018 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1614–1619. IEEE, 2018.

[154] W. Hu and Y. Tan. Generating adversarial malware examples for black-box attacks based on gan. *arXiv preprint arXiv:1702.05983*, 02, 2017.

[155] K. Huang, H. Yang, I. King, and M. R. Lyu. Maxi–min margin machine: learning large margin classifiers locally and globally. *IEEE Transactions on Neural Networks*, 19(2):260–272, 2008.

[156] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

[157] N. Humbatova, G. Jahangirova, G. Bavota, V. Riccio, A. Stocco, and P. Tonella. Taxonomy of real faults in deep learning systems. In *42nd ACM/IEEE International Conference on Software Engineering, ICSE*, pages 1110–1121. IEEE, 2020.

[158] L. Hussenot, M. Geist, and O. Pietquin. Copycat: Taking control of neural policies with constant attacks. *arXiv preprint arXiv:1905.12282*, 2019.

[159] T. Huster, C.-Y. J. Chiang, and R. Chadha. Limitations of the lipschitz constant as a defense against adversarial examples. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 16–29. Springer, 2018.

[160] IBM. Ibm ilog cplex optimizer, 2022. URL `https://www.ibm.com/analytics/cplex-optimizer`.

[161] K. Inc. Heritage health prize kaggle. `https://www.kaggle.com/c/hhp`, 2022. URL `https://www.kaggle.com/c/hhp`.

[162] G. N. Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2):257–280, 2005.

[163] J. Jacod and P. Protter. *Probability essentials*. Springer Science & Business Media, Berlin, 2000.

[164] N. Jay, N. Rotman, B. Godfrey, M. Schapira, and A. Tamar. A deep reinforcement learning perspective on internet congestion control. In *International Conference on Machine Learning*, pages 3050–3059. PMLR, 2019.

[165] J. Jeong and J. Shin. Consistency regularization for certified robustness of smoothed classifiers. In *2020 Advances in Neural Information Processing Systems (NeurIPS)*, pages 10558–10570. Curran Associates, Inc., 2020.

[166] J. Jeong, S. Park, M. Kim, H.-C. Lee, D. Kim, and J. Shin. Smoothmix: Training confidence-calibrated smoothed classifiers for certified robustness. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[167] J. Jia, X. Cao, and N. Z. Gong. Certified robustness of nearest neighbors against data poisoning attacks. *arXiv preprint arXiv:2012.03765*, 2020.

[168] J. Jia, B. Wang, X. Cao, H. Liu, and N. Z. Gong. Almost tight l0-norm certified robustness of top-k predictions against adversarial perturbations. In *International Conference on Learning Representations*, 2022.

[169] K. Jia and M. Rinard. Exploiting verified neural networks via floating point numerical error. In *International Static Analysis Symposium*, pages 191–205. Springer, 2021.

[170] R. Jia and P. Liang. Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*, 2017.

[171] R. Jia, A. Raghunathan, K. Göksel, and P. Liang. Certified robustness to adversarial word substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 4127–4140, 2019.

[172] X. Jin, F. Barbieri, B. Kennedy, A. M. Davani, L. Neves, and X. Ren. On transferability of bias mitigation effects in language model fine-tuning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 3770–3783. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.naacl-main.296.

[173] P. G. John, D. Vijaykeerthy, and D. Saha. Verifying individual fairness in machine learning models. In *Conference on Uncertainty in Artificial Intelligence*, pages 749–758. PMLR, 2020.

[174] E. L. Johnson and M. W. Padberg. Degree-two inequalities, clique facets, and biperfect graphs. In *North-Holland Mathematics Studies*, volume 66, pages 169–187. Elsevier, 1982.

[175] M. Jordan, J. Lewis, and A. G. Dimakis. Provable certificates for adversarial examples: Fitting a ball in the union of polytopes. In *Advances in Neural Information Processing Systems*, pages 14059–14069, 2019.

[176] N. Jovanović, M. Balunovic, M. Baader, and M. Vechev. On the paradox of certified training. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. URL `https://openreview.net/forum?id=atJHLVyBi8`.

[177] M. Kang, L. Li, M. Weber, Y. Liu, C. Zhang, and B. Li. Certifying some distributional fairness with subpopulation decomposition. In *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)*, 2022.

[178] S. Kariyappa and M. K. Qureshi. Improving adversarial robustness of ensembles with diversity training. *arXiv preprint arXiv:1901.09981*, 2019.

[179] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[180] G. Katz, C. Barrett, D. L. Dill, K. Julian, and M. J. Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *International conference on computer aided verification*, pages 97–117. Springer, 2017.

[181] G. Katz, D. A. Huang, D. Ibeling, K. Julian, C. Lazarus, R. Lim, P. Shah, S. Thakoor, H. Wu, A. Zeljić, et al. The marabou framework for verification and analysis of deep neural networks. In *International Conference on Computer Aided Verification*, pages 443–452. Springer, 2019.

[182] T. Kehrenberg, M. Bartlett, O. Thomas, and N. Quadrianto. Null-sampling for interpretable and fair representations. In *European Conference on Computer Vision*, pages 565–580. Springer, 2020.

[183] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*, 2015.

[184] P. Kiourti, K. Wardega, S. Jha, and W. Li. Trojdrl: evaluation of backdoor attacks on deep reinforcement learning. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6. IEEE, 2020.

[185] J. Kleinberg, S. Mullainathan, and M. Raghavan. Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807*, 2016.

[186] E. Kloberdanz, K. G. Kloberdanz, and W. Le. Deepstability: A study of unstable numerical methods and their solutions in deep learning. In *Proceedings of the 44th International Conference on Software Engineering*, ICSE '22, page 586–597, New York,

NY, USA, 2022. Association for Computing Machinery. ISBN 9781450392211. doi: 10.1145/3510003.3510095. URL https://doi.org/10.1145/3510003.3510095.

[187] C.-Y. Ko, Z. Lyu, L. Weng, L. Daniel, N. Wong, and D. Lin. Popqorn: Quantifying robustness of recurrent neural networks. In *International Conference on Machine Learning*, pages 3468–3477. PMLR, 2019.

[188] J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[189] J. Kos and D. Song. Delving into adversarial attacks on deep policies. *arXiv preprint arXiv:1705.06452*, 2017.

[190] P. Kouvaros and A. Lomuscio. Towards scalable complete verification of relu neural networks via dependency-based branching. In *IJCAI*, pages 2643–2650, 2021.

[191] A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[192] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25, NIPS*, pages 1106–1114, 2012.

[193] A. Kumar, A. Levine, S. Feizi, and T. Goldstein. Certifying confidence via randomized smoothing. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5165–5177. Curran Associates, Inc., 2020.

[194] A. Kumar, A. Levine, S. Feizi, and T. Goldstein. Certifying confidence via randomized smoothing. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, pages 5165–5177. Curran Associates, Inc., 2020.

[195] A. Kumar, A. Levine, T. Goldstein, and S. Feizi. Curse of dimensionality on randomized smoothing for certifiable robustness. In *2020 International Conference on Machine Learning (ICML)*, pages 5458–5467. PMLR, 2020.

[196] A. Kumar, A. Levine, and S. Feizi. Policy smoothing for provably robust reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=mwdfai8NBrJ.

[197] A. Kumar, A. Levine, T. Goldstein, and S. Feizi. Certifying model accuracy under distribution shifts. *arXiv preprint arXiv:2201.12440*, 2022.

[198] R. S. S. Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia. Adversarial machine learning-industry perspectives. In *2020 IEEE Security and Privacy Workshops (SPW)*, pages 69–75. IEEE, 2020.

[199] A. Kurakin, I. Goodfellow, and S. Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[200] H. Lakkaraju, J. Kleinberg, J. Leskovec, J. Ludwig, and S. Mullainathan. The selective labels problem: Evaluating algorithmic predictions in the presence of unobservables. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 275–284, 2017.

[201] H. Lam. Robust sensitivity analysis for stochastic systems. *Mathematics of Operations Research*, 41(4):1248–1275, 2016.

[202] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[203] Y. LeCun, C. Cortes, and C. Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[204] M. Lecuyer, V. Atlidakis, R. Geambasu, D. Hsu, and S. Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672. IEEE, 2019.

[205] G.-H. Lee, Y. Yuan, S. Chang, and T. Jaakkola. Tight certificates of adversarial robustness for randomly smoothed classifiers. In *Advances in Neural Information Processing Systems*, pages 4911–4922, 2019.

[206] S. Lee, J. Lee, and S. Park. Lipschitz-certifiable training with a tight outer bound. *Advances in Neural Information Processing Systems*, 33, 2020.

[207] S. Lee, W. Lee, J. Park, and J. Lee. Towards better understanding of training certifiably robust models against adversarial examples. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[208] R. Legenstein, N. Wilbert, and L. Wiskott. Reinforcement learning on slow features of high-dimensional input streams. *PLoS computational biology*, 6(8):e1000894, 2010.

[209] K. Leino, Z. Wang, and M. Fredrikson. Globally-robust neural networks. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 6212–6222. PMLR, 18–24 Jul 2021.

[210] E. Leurent. An environment for autonomous driving decision-making. `https://github.com/eleurent/highway-env`, 2018.

[211] A. Levine and S. Feizi. (de) randomized smoothing for certifiable defense against patch attacks. *Advances in Neural Information Processing Systems*, 33:6465–6475, 2020.

[212] A. Levine and S. Feizi. Robustness certificates for sparse adversarial attacks by randomized ablation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4585–4593, 2020.

[213] A. Levine and S. Feizi. Deep partition aggregation: Provable defenses against general poisoning attacks. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YUGG2tFuPM.

[214] A. Levine and S. Feizi. Improved, deterministic smoothing for ℓ_1 certified robustness. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 6254–6264. PMLR, 2021.

[215] A. Levine, A. Kumar, T. Goldstein, and S. Feizi. Tight second-order certificates for randomized smoothing. *arXiv preprint arXiv:2010.10549*, 2020.

[216] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[217] B. Li. 3D fully convolutional network for vehicle detection in point cloud. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1513–1518. IEEE, 2017.

[218] B. Li, C. Chen, W. Wang, and L. Carin. Certified adversarial robustness with additive noise. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9459–9469. Curran Associates, Inc., 2019.

[219] H. Li, X. Xu, X. Zhang, S. Yang, and B. Li. Qeba: Query-efficient boundary-based blackbox attack. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1221–1230, 2020.

[220] H. Li, L. Li, X. Xu, X. Zhang, S. Yang, and B. Li. Nonlinear gradient estimation for query efficient blackbox attack. In *International Conference on Artificial Intelligence and Statistics (AISTATS 2021)*, Proceedings of Machine Learning Research. PMLR, 13–15 Apr 2021.

[221] L. Li, Z. Zhong, B. Li, and T. Xie. Robustra: Training provable robust neural networks over reference adversarial space. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4711–4717, 2019.

[222] L. Li, Z. Li, W. Zhang, J. Zhou, P. Wang, J. Wu, G. He, X. Zeng, Y. Deng, and T. Xie. Clustering test steps in natural language toward automating test automation. In *Proceedings of the 28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2020)*, 2020. doi: 10.1145/3368089.3417067. URL https://doi.org/10.1145/3368089.3417067.

[223] L. Li, M. Weber, X. Xu, L. Rimanic, B. Kailkhura, T. Xie, C. Zhang, and B. Li. Tss: Transformation-specific smoothing for robustness certification. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS 2021)*, 2021.

[224] L. Li, J. Zhang, T. Xie, and B. Li. Double sampling randomized smoothing. In *39th International Conference on Machine Learning (ICML)*, 2022.

[225] L. Li, T. Xie, and B. Li. Sok: Certified robustness for deep neural networks. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, 22-26 May 2023*. IEEE, 2023.

[226] L. Li, Y. Zhang, L. Ren, Y. Xiong, and T. Xie. Reliability assurance for deep neural network architectures against numerical defects. In *In submission to 45th International Conference on Software Engineering (ICSE 2023)*, 2023.

[227] Q. Li, S. Haque, C. Anil, J. Lucas, R. B. Grosse, and J.-H. Jacobsen. Preventing gradient attenuation in lipschitz constrained convolutional networks. *Advances in neural information processing systems*, 32:15390–15402, 2019.

[228] Y. Li and Y. Yuan. Convergence analysis of two-layer neural networks with ReLU activation. In *Advances in Neural Information Processing Systems 30, NIPS*, pages 597–607, 2017.

[229] Y. Li, H. Su, J. Zhu, and J. Zhou. Boosting the robustness of capsule networks with diverse ensemble. In *2020 10th International Conference on Information Science and Technology (ICIST)*, pages 247–251. IEEE, 2020.

[230] Y. Li, D. Choi, J. Chung, N. Kushman, J. Schrittwieser, R. Leblond, T. Eccles, J. Keeling, F. Gimeno, A. Dal Lago, et al. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097, 2022.

[231] J. Liao, C. Huang, P. Kairouz, and L. Sankar. Learning generative adversarial representations (gap) under fairness and censoring constraints. *arXiv preprint arXiv:1910.00411*, 1, 2019.

[232] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun. Tactics of adversarial attack on deep reinforcement learning agents. *arXiv preprint arXiv:1703.06748*, 2017.

[233] C. Liu and T. Johnson. Vnn comp 2020, 2020. URL `https://sites.google.com/view/vnn20/vnncomp`.

[234] C. Liu, Y. Feng, R. Wang, and B. Dong. Enhancing certified robustness of smoothed classifiers via weighted model ensembling. *arXiv preprint arXiv:2005.09363*, 2020.

[235] C. Liu, J. Lu, G. Li, T. Yuan, L. Li, F. Tan, J. Yang, L. You, and J. Xue. Detecting TensorFlow program bugs in real-world industrial environment. In *36th IEEE/ACM International Conference on Automated Software Engineering, ASE*, pages 55–66. IEEE, 2021.

[236] H. Liu, J. Jia, and N. Z. Gong. Pointguard: Provably robust 3d point cloud classification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6186–6195, 2021.

[237] K. Liu, B. Dolan-Gavitt, and S. Garg. Fine-pruning: Defending against backdooring attacks on deep neural networks. In *International Symposium on Research in Attacks, Intrusions, and Defenses*, pages 273–294. Springer, 2018.

[238] L. T. Liu, M. Simchowitz, and M. Hardt. The implicit fairness criterion of unconstrained learning. In *International Conference on Machine Learning*, pages 4051–4060. PMLR, 2019.

[239] F. Locatello, G. Abbati, T. Rainforth, S. Bauer, B. Schölkopf, and O. Bachem. On the fairness of disentangled representations. *Advances in Neural Information Processing Systems*, 32, 2019.

[240] A. Lomuscio and L. Maganti. An approach to reachability analysis for feed-forward relu neural networks. *arXiv preprint arXiv:1706.07351*, 2017.

[241] T. Lorenz, A. Ruoss, M. Balunović, G. Singh, and M. Vechev. Robustness certification for point cloud models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7608–7618, 2021.

[242] C. Louizos, K. Swersky, Y. Li, M. Welling, and R. Zemel. The variational fair autoencoder. *arXiv preprint arXiv:1511.00830*, 2015.

[243] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0–1 optimization. *SIAM journal on optimization*, 1(2):166–190, 1991.

[244] D. W. Lozier. NIST digital library of mathematical functions. *Annals of Mathematics and Artificial Intelligence*, 38(1-3):105–119, 2003.

[245] J. Lu and M. P. Kumar. Neural network branching for neural network verification. In *International Conference on Learning Representations*, 2020.

[246] Z. Lyu, C.-Y. Ko, Z. Kong, N. Wong, D. Lin, and L. Daniel. Fastened crown: Tightened neural network robustness certificates. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 5037–5044, 2020.

[247] Z. Lyu, M. Guo, T. Wu, G. Xu, K. Zhang, and D. Lin. Towards evaluating and training verifiably robust neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4308–4317, 2021.

[248] L. Ma, F. Juefei-Xu, F. Zhang, J. Sun, M. Xue, B. Li, C. Chen, T. Su, L. Li, Y. Liu, J. Zhao, and Y. Wang. DeepGauge: Multi-granularity testing criteria for deep learning systems. In *33rd ACM/IEEE International Conference on Automated Software Engineering, ASE*, pages 120–131. ACM, 2018.

[249] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, G. Schoenebeck, D. Song, M. E. Houle, and J. Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. In *2018 International Conference on Learning Representations (ICLR)*. OpenReview, 2018.

[250] D. Madras, E. Creager, T. Pitassi, and R. Zemel. Learning adversarially fair and transferable representations. In *International Conference on Machine Learning*, pages 3384–3393. PMLR, 2018.

[251] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[252] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[253] S. Maity, D. Mukherjee, M. Yurochkin, and Y. Sun. Does enforcing fairness mitigate biases caused by subpopulation shift? In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 25773–25784. Curran Associates, Inc., 2021.

[254] A. Mandlekar, Y. Zhu, A. Garg, L. Fei-Fei, and S. Savarese. Adversarially robust policy learning: Active construction of physically-plausible perturbations. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3932–3939. IEEE, 2017.

[255] S. Manikandan. Measures of central tendency: Median and mode. *Journal of pharmacology and pharmacotherapeutics*, 2(3):214, 2011.

[256] H. Marchand and L. A. Wolsey. Aggregation and mixed integer rounding to solve mips. *Operations research*, 49(3):363–371, 2001.

[257] A. Maurer and M. Pontil. Empirical bernstein bounds and sample variance penalization. *arXiv preprint arXiv:0907.3740*, 2009.

[258] P. McCausland. Self-driving uber car that hit and killed woman did not recognize that pedestrians jaywalk, 2022. URL `https://www.nbcnews.com/tech-news/n1079281`.

[259] D. McNamara, C. S. Ong, and R. C. Williamson. Costs and benefits of fair representation learning. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, page 263–270, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450363242. doi: 10.1145/3306618.3317964. URL `https://doi.org/10.1145/3306618.3317964`.

[260] A. Mehra, B. Kailkhura, P.-Y. Chen, and J. Hamm. How robust are randomized smoothing based defenses to data poisoning? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13244–13253, 2021.

[261] D. Michie. "memo" functions and machine learning. *Nature*, 218(5136):19–22, 1968.

[262] M. Mirman, T. Gehr, and M. Vechev. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pages 3575–3583, 2018.

[263] M. Mirman, A. Hägele, P. Bielik, T. Gehr, and M. Vechev. Robustness certification with generative models. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 1141–1154, 2021.

[264] M. Mirman, A. Hägele, P. Bielik, T. Gehr, and M. Vechev. Robustness certification with generative models. In *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*, pages 1141–1154, 2021.

[265] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[266] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[267] J. Mohapatra, C.-Y. Ko, T.-W. Weng, P.-Y. Chen, S. Liu, and L. Daniel. Higher-order certification for randomized smoothing. *Advances in Neural Information Processing Systems*, 33, 2020.

[268] J. Mohapatra, T.-W. Weng, P.-Y. Chen, S. Liu, and L. Daniel. Towards verifying robustness of neural networks against a family of semantic perturbations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 244–252. IEEE, 2020.

[269] J. Mohapatra, C.-Y. Ko, L. Weng, P.-Y. Chen, S. Liu, and L. Daniel. Hidden cost of randomized smoothing. In *International Conference on Artificial Intelligence and Statistics*, pages 4033–4041. PMLR, 2021.

[270] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2574–2582, 2016.

[271] C. Müller, F. Serre, G. Singh, M. Püschel, and M. Vechev. Scaling polyhedral neural network verification on gpus. *Proceedings of Machine Learning and Systems*, 3:733–746, 2021.

[272] M. N. Müller, C. Brix, S. Bak, C. Liu, and T. T. Johnson. The third international verification of neural networks competition (vnn-comp 2022): summary and results. *arXiv preprint arXiv:2212.10376*, 2022.

[273] M. N. Müller, G. Makarchuk, G. Singh, M. Püschel, and M. Vechev. PRIMA: Precise and general neural network certification via multi-neuron convex relaxations. *Proceedings of the ACM on Programming Languages*, 6(POPL):1–33, 2022.

[274] B. Nelson, M. Barreno, F. J. Chi, A. D. Joseph, B. I. Rubinstein, U. Saini, C. Sutton, J. D. Tygar, and K. Xia. Exploiting machine learning to subvert your spam filter. *LEET*, 8:1–9, 2008.

[275] G. L. Nemhauser and L. A. Wolsey. A recursive procedure to generate all cuts for 0–1 mixed integer programs. *Mathematical Programming*, 46(1):379–390, 1990.

[276] J. Neyman and E. S. Pearson. On the problem of the most efficient tests of statistical hypotheses. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 231:289–337, 1933.

[277] A. Nilim and L. El Ghaoui. Robust control of markov decision processes with uncertain transition matrices. *Operations Research*, 53(5):780–798, 2005.

[278] A. Odena, C. Olsson, D. Andersen, and I. Goodfellow. Tensorfuzz: Debugging neural networks with coverage-guided fuzzing. In *International Conference on Machine Learning*, pages 4901–4911. PMLR, 2019.

[279] W. H. O. of Science and T. Policy. *Blueprint for an AI bill of rights*. The White House, Washington, D.C., 2022.

[280] T. Oikarinen, T.-W. Weng, and L. Daniel. Robust deep reinforcement learning through adversarial loss. *arXiv preprint arXiv:2008.01976*, 2020.

[281] R. Olivier and B. Raj. Sequential randomized smoothing for adversarially robust speech recognition. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6372–6386, 2021.

[282] L. Oneto, M. Donini, M. Pontil, and A. Maurer. Learning fair and transferable representations with theoretical guarantees. In *2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA)*, pages 30–39, 2020. doi: 10.1109/DSAA49011.2020.00015.

[283] OpenCV. Opencv: Transformations of images. `https://docs.opencv.org/master/dd/d52/tutorial_js_geometric_transformations.html`, 2020.

[284] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198, 1999.

[285] M. W. Padberg, T. J. Van Roy, and L. A. Wolsey. Valid linear inequalities for fixed charge problems. *Operations Research*, 33(4):842–861, 1985.

[286] A. D. Palma, H. Behl, R. R. Bunel, P. Torr, and M. P. Kumar. Scaling the convex barrier with active sets. In *International Conference on Learning Representations*, 2021.

[287] T. Pang, K. Xu, C. Du, N. Chen, and J. Zhu. Improving adversarial robustness via promoting ensemble diversity. In *International Conference on Machine Learning*, pages 4970–4979. PMLR, 2019.

[288] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 582–597. IEEE, 2016.

[289] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

[290] A. Pattanaik, Z. Tang, S. Liu, G. Bommannan, and G. Chowdhary. Robust deep reinforcement learning with adversarial attacks. In *17th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2018*, pages 2040–2042. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2018.

[291] K. Pei, Y. Cao, J. Yang, and S. Jana. Deepxplore: Automated whitebox testing of deep learning systems. In *2017 Symposium on Operating Systems Principles (SOSP)*, pages 1–18. Association for Computing Machinery, 2017.

[292] K. Pei, Y. Cao, J. Yang, and S. Jana. Towards practical verification of machine learning: The case of computer vision systems. *arXiv preprint arXiv:1712.01785*, 2017.

[293] N. Peri, N. Gupta, W. R. Huang, L. Fowl, C. Zhu, S. Feizi, T. Goldstein, and J. P. Dickerson. Deep k-nn defense against clean-label data poisoning attacks. In *European Conference on Computer Vision*, pages 55–70. Springer, 2020.

[294] M. Peychev, A. Ruoss, M. Balunović, M. Baader, and M. Vechev. Latent space smoothing for individually fair representations. In *European Conference on Computer Vision*, pages 535–554. Springer, 2022.

[295] H. V. Pham, S. Qian, J. Wang, T. Lutellier, J. Rosenthal, L. Tan, Y. Yu, and N. Nagappan. Problems and opportunities in training deep learning software systems: An analysis of variance. In *35th IEEE/ACM International Conference on Automated Software Engineering, ASE*, pages 771–783. IEEE, 2020.

[296] A. S. Polydoros and L. Nalpantidis. Survey of model-based reinforcement learning: Applications on robotics. *Journal of Intelligent & Robotic Systems*, 86(2):153–173, 2017.

[297] L. Powell. The problem with artificial intelligence in security, 2022. URL `https://www.darkreading.com/threat-intelligence/the-problem-with-artificial-intelligence-in-security`.

[298] L. Pulina and A. Tacchella. An abstraction-refinement approach to verification of artificial neural networks. In *International Conference on Computer Aided Verification*, pages 243–257. Springer, 2010.

[299] L. Pulina and A. Tacchella. Challenging smt solvers to verify neural networks. *Ai Communications*, 25(2):117–135, 2012.

[300] PyTorch. torchvision.models — torchvison 0.10.0 documentation. `https://pytorch.org/vision/stable/models.html`, 2021.

[301] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[302] H. Qi, Z. Wang, Q. Guo, J. Chen, F. Juefei-Xu, L. Ma, and J. Zhao. ArchRepair: Block-level architecture-oriented repairing for deep neural networks. *CoRR*, abs/2111.13330, 2021.

[303] C. Qin, B. O'Donoghue, R. Bunel, R. Stanforth, S. Gowal, J. Uesato, G. Swirszcz, P. Kohli, et al. Verification of non-linear specifications for neural networks. *arXiv preprint arXiv:1902.09592*, 2019.

[304] H. Qiu, C. Xiao, L. Yang, X. Yan, H. Lee, and B. Li. Semanticadv: Generating adversarial examples via attribute-conditioned image editing. In A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, editors, *16th European Conference on Computer Vision, ECCV 2020, Glasgow, UK, August 23-28, 2020, Proceedings, Part XIV*, volume 12359 of *Lecture Notes in Computer Science*, pages 19–37. Springer, 2020.

[305] M. Qu and J. Tang. Probabilistic logic neural networks for reasoning. *Advances in neural information processing systems*, 32, 2019.

[306] R. Raab and Y. Liu. Unintended selection: Persistent qualification rate disparities and interventions. *Advances in Neural Information Processing Systems*, 34, 2021.

[307] A. Raghunathan, J. Steinhardt, and P. Liang. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018.

[308] A. Raghunathan, J. Steinhardt, and P. S. Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems*, pages 10877–10887, 2018.

[309] A. Reisizadeh, F. Farnia, R. Pedarsani, and A. Jadbabaie. Robust federated learning: The case of affine distribution shifts. *Advances in Neural Information Processing Systems*, 33:21554–21565, 2020.

[310] Y. Roh, K. Lee, S. Whang, and C. Suh. Sample selection for fair and robust training. *Advances in Neural Information Processing Systems*, 34, 2021.

[311] L. Rokach. Ensemble-based classifiers. *Artificial intelligence review*, 33(1):1–39, 2010.

[312] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[313] A. Ruoss, M. Balunovic, M. Fischer, and M. Vechev. Learning certified individually fair representations. *Advances in Neural Information Processing Systems*, 33:7584–7596, 2020.

[314] A. Russo and A. Proutiere. Optimal attacks on reinforcement learning policies. *arXiv preprint arXiv:1907.13548*, 2019.

[315] W. Ryou, J. Chen, M. Balunovic, G. Singh, A. Dan, and M. Vechev. Scalable polyhedral verification of recurrent neural networks. In *International Conference on Computer Aided Verification*, pages 225–248. Springer, 2021.

[316] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

[317] H. Salman, J. Li, I. P. Razenshteyn, P. Zhang, H. Zhang, S. Bubeck, and G. Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11289–11300. Curran Associates, Inc., 2019.

[318] H. Salman, G. Yang, H. Zhang, C.-J. Hsieh, and P. Zhang. A convex relaxation barrier to tight robustness verification of neural networks. In *2019 Advances in Neural Information Processing Systems (NeurIPS)*, pages 9835–9846. Curran Associates, Inc., 2019.

[319] H. Salman, M. Sun, G. Yang, A. Kapoor, and J. Z. Kolter. Denoised smoothing: A provable defense for pretrained classifiers. *Advances in Neural Information Processing Systems*, 33:21945–21957, 2020.

[320] H. Salman, S. Jain, E. Wong, and A. Madry. Certified patch robustness via smoothed vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15137–15147, 2022.

[321] P. Samangouei, M. Kabkab, and R. Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations*, 2018.

[322] M. H. Sarhan, N. Navab, A. Eslami, and S. Albarqouni. Fairness by learning orthogonal disentangled representations. In *European Conference on Computer Vision*, pages 746–761. Springer, 2020.

[323] J. Schuchardt, T. Wollschläger, A. Bojchevski, and S. Günnemann. Localized randomized smoothing for collective robustness certification, 2022. URL `https://openreview.net/forum?id=mF122BuAnnW`.

[324] A. Schwarzschild, M. Goldblum, A. Gupta, J. P. Dickerson, and T. Goldstein. Just how toxic is data poisoning? a unified benchmark for backdoor and data poisoning attacks. In *International Conference on Machine Learning*, pages 9389–9398. PMLR, 2021.

[325] S. Segal, Y. Adi, B. Pinkas, C. Baum, C. Ganesh, and J. Keshet. Fairness in the eyes of the data: Certifying machine-learning models. In *Proceedings of the 2021 AAAI/ACM Conference on AI, Ethics, and Society*, pages 926–935, 2021.

[326] V. Sehwag, S. Wang, P. Mittal, and S. Jana. Hydra: Pruning adversarially robust neural networks. *Advances in Neural Information Processing Systems*, 33:19655–19666, 2020.

[327] T. Serra, C. Tjandraatmadja, and S. Ramalingam. Bounding and counting linear regions of deep neural networks. In *35th International Conference on Machine Learning, ICML*, pages 4565–4573. PMLR, 2018.

[328] S. Shalev-Shwartz, S. Shammah, and A. Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

[329] C. E. Shannon. Communication theory of secrecy systems. *The Bell system technical journal*, 28(4):656–715, 1949.

[330] D. K. Sharma, S. K. Dhurandher, I. Woungang, R. K. Srivastava, A. Mohananey, and J. J. Rodrigues. A machine learning-based protocol for efficient routing in opportunistic networks. *IEEE Systems Journal*, 12(3):2207–2213, 2016.

[331] S. Shekhar, G. Fields, M. Ghavamzadeh, and T. Javidi. Adaptive sampling for minimax fair classification. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 24535–24544. Curran Associates, Inc., 2021.

[332] Q. Shen, Y. Li, H. Jiang, Z. Wang, and T. Zhao. Deep reinforcement learning with robust and smooth policy. In *International Conference on Machine Learning*, pages 8707–8718. PMLR, 2020.

[333] Q. Shen, H. Ma, J. Chen, Y. Tian, S. Cheung, and X. Chen. A comprehensive study of deep learning compiler bugs. In *29th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE*, pages 968–980. ACM, 2021.

[334] Z. Shen, J. Liu, Y. He, X. Zhang, R. Xu, H. Yu, and P. Cui. Towards out-of-distribution generalization: A survey. *arXiv preprint arXiv:2108.13624*, 2021.

[335] H. D. Sherali and W. P. Adams. *A reformulation-linearization technique for solving discrete and continuous nonconvex problems*, volume 31. Springer Science & Business Media, 2013.

[336] Z. Shi, H. Zhang, K.-W. Chang, M. Huang, and C.-J. Hsieh. Robustness verification for transformers. In *International Conference on Learning Representations*, 2020.

[337] Z. Shi, Y. Wang, H. Zhang, J. Yi, and C.-J. Hsieh. Fast certified robust training with short warmup. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[338] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[339] G. Singh, M. Püschel, and M. T. Vechev. Fast polyhedra abstract domain. In *44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL*, pages 46–59. ACM, 2017.

[340] G. Singh, T. Gehr, M. Mirman, M. Püschel, and M. Vechev. Fast and effective robustness certification. In *Advances in Neural Information Processing Systems*, pages 10802–10813, 2018.

[341] G. Singh, T. Gehr, M. Püschel, and M. Vechev. Boosting robustness certification of neural networks. In *International Conference on Learning Representations*, 2018.

[342] G. Singh, R. Ganvir, M. Püschel, and M. Vechev. Beyond the single neuron convex barrier for neural network certification. In *Advances in Neural Information Processing Systems*, pages 15072–15083, 2019.

[343] G. Singh, T. Gehr, M. Püschel, and M. Vechev. An abstract domain for certifying neural networks. *Proc. ACM Program. Lang.*, 3(POPL), Jan. 2019.

[344] S. Singla and S. Feizi. Second-order provable defenses against adversarial attacks. In *International Conference on Machine Learning*, 2020.

[345] S. Singla and S. Feizi. Fantastic four: Differentiable and efficient bounds on singular values of convolution layers. In *International Conference on Learning Representations*, 2021.

[346] S. Singla, S. Singla, and S. Feizi. Improved deterministic l2 robustness on CIFAR-10 and CIFAR-100. In *International Conference on Learning Representations*, 2022.

[347] A. Sinha, H. Namkoong, and J. Duchi. Certifying some distributional robustness with principled adversarial training. In *International Conference on Learning Representations*, 2018.

[348] J. Song, P. Kalluri, A. Grover, S. Zhao, and S. Ermon. Learning controllable fair representations. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2164–2173. PMLR, 2019.

[349] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[350] M. Sotoudeh and A. V. Thakur. Provable repair of deep neural networks. In *42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation, PLDI*, pages 588–603. ACM, 2021.

[351] T. Steerneman. On the total variation and hellinger distance between signed measures; an application to product measures. *Proceedings of the American Mathematical Society*, 88(4):684–688, 1983.

[352] C. Stein et al. A bound for the error in the normal approximation to the distribution of a sum of dependent random variables. In *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability, Volume 2: Probability Theory*. The Regents of the University of California, 1972.

[353] J. Steinhardt, P. W. Koh, and P. Liang. Certified defenses for data poisoning attacks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 3520–3532, 2017.

[354] P. Súkeník, A. Kuvshinov, and S. Günnemann. Intriguing properties of input-dependent randomized smoothing. In *International Conference on Machine Learning*, pages 20697–20743. PMLR, 2022.

[355] J. Sun, Y. Cao, Q. A. Chen, and Z. M. Mao. Towards robust lidar-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 877–894, 2020.

[356] J. Sun, Y. Cao, C. Choy, Z. Yu, A. Anandkumar, Z. Mao, and C. Xiao. Adversarially robust 3d point cloud recognition using self-supervisions. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL `https://openreview.net/forum?id=6_sF7BuscXe`.

[357] J. Sun, K. Koenig, Y. Cao, Q. A. Chen, and Z. M. Mao. On adversarial robustness of 3d point cloud classification under adaptive attacks. In *The 32nd British Machine Vision Conference, BMVC 2021*, 2021.

[358] Y. Sun, R. Zheng, Y. Liang, and F. Huang. Who is the strongest enemy? towards optimal and efficient evasion attacks in deep RL. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=JM2kFbJvvI`.

[359] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL `http://arxiv.org/abs/1312.6199`.

[360] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[361] F. Tambon, A. Nikanjam, L. An, F. Khomh, and G. Antoniol. Silent bugs in deep learning frameworks: An empirical study of Keras and TensorFlow. *CoRR*, abs/2112.13314, 2021.

[362] C. Tan, Y. Zhu, and C. Guo. Building verified neural networks with specifications for systems. In *Proceedings of the 12th ACM SIGOPS Asia-Pacific Workshop on Systems*, pages 42–47, 2021.

[363] R. Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160, 1972.

[364] J. Teng, G.-H. Lee, and Y. Yuan. ℓ_1 adversarial robustness certificates: a randomized smoothing approach, 2020. URL `https://openreview.net/forum?id=H1lQIgrFDS`.

[365] The Linux Foundation. ONNX home. `https://onnx.ai/`, 2022. Accessed: 2023-02-01.

[366] Y. Tian, K. Pei, S. Jana, and B. Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*, pages 303–314, 2018.

[367] C. Tjandraatmadja, R. Anderson, J. Huchette, W. Ma, K. K. PATEL, and J. P. Vielma. The convex relaxation barrier, revisited: Tightened single-neuron relaxations for neural network verification. *Advances in Neural Information Processing Systems*, 33:21675–21686, 2020.

[368] V. Tjeng, K. Y. Xiao, and R. Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2019.

[369] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[370] F. Tramèr, A. Kurakin, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel. Ensemble adversarial training: Attacks and defenses. In *2018 International Conference on Learning Representations (ICLR)*. OpenReview, 2018.

[371] F. Tramer, N. Carlini, W. Brendel, and A. Madry. On adaptive attacks to adversarial example defenses. *Advances in Neural Information Processing Systems*, 33:1633–1645, 2020.

[372] H.-D. Tran, X. Yang, D. M. Lopez, P. Musau, L. V. Nguyen, W. Xiang, S. Bak, and T. T. Johnson. Nnv: The neural network verification tool for deep neural networks and learning-enabled cyber-physical systems. In *International Conference on Computer Aided Verification*, pages 3–17. Springer, 2020.

[373] A. Trockman and J. Z. Kolter. Orthogonalizing convolutional layers with the cayley transform. In *International Conference on Learning Representations*, 2021.

[374] Y. Tsuzuku, I. Sato, and M. Sugiyama. Lipschitz-margin training: scalable certification of perturbation invariance for deep neural networks. In *Advances in Neural Information Processing Systems*, 2018.

[375] C. Urban, M. Christakis, V. Wüstholz, and F. Zhang. Perfectly parallel fairness certification of neural networks. *Proceedings of the ACM on Programming Languages*, 4 (OOPSLA):1–30, 2020.

[376] US Government. Promoting the use of trustworthy artificial intelligence in the federal government. `https://www.federalregister.gov/d/2020-27065`, 2020. Accessed: 2021-05-20.

[377] P. van Emde Boas. Preserving order in a forest in less than logarithmic time and linear space. *Information processing letters*, 6(3):80–82, 1977.

[378] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5998–6008, 2017.

[379] R. Veerapaneni, J. D. Co-Reyes, M. Chang, M. Janner, C. Finn, J. Wu, J. Tenenbaum, and S. Levine. Entity abstraction in visual model-based reinforcement learning. In *Conference on Robot Learning*, pages 1439–1456. PMLR, 2020.

[380] V. Voráček and M. Hein. Provably adversarially robust nearest prototype classifiers. In *International Conference on Machine Learning*, pages 22361–22383. PMLR, 2022.

[381] V. Voráček and M. Hein. Sound randomized smoothing in floating-point arithmetics. *arXiv preprint arXiv:2207.07209*, 2022.

[382] C. Wan, S. Liu, H. Hoffmann, M. Maire, and S. Lu. Are machine learning cloud APIs used correctly? In *43rd IEEE/ACM International Conference on Software Engineering, ICSE*, pages 125–137. IEEE, 2021.

[383] C. Wan, S. Liu, S. Xie, Y. Liu, H. Hoffmann, M. Maire, and S. Lu. Automated testing of software that uses machine learning APIs. In *44th IEEE/ACM International Conference on Software Engineering, ICSE*, pages 212–224. ACM, 2022.

[384] B. Wang, C. Xu, S. Wang, Z. Gan, Y. Cheng, J. Gao, A. H. Awadallah, and B. Li. Adversarial GLUE: A multi-task benchmark for robustness evaluation of language models. In J. Vanschoren and S. Yeung, editors, *Proceedings of the Neural Information*

Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual, volume 1, 2021.

[385] L. Wang, Z. Javed, X. Wu, W. Guo, X. Xing, and D. Song. Backdoorl: Backdoor attack against competitive reinforcement learning. *arXiv preprint arXiv:2105.00579*, 2021.

[386] S. Wang, Y. Chen, A. Abdou, and S. Jana. Mixtrain: Scalable training of verifiably robust neural networks. *arXiv preprint arXiv:1811.02625*, 2018.

[387] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana. Efficient formal safety analysis of neural networks. In *Advances in Neural Information Processing Systems*, pages 6367–6377, 2018.

[388] S. Wang, K. Pei, J. Whitehouse, J. Yang, and S. Jana. Formal security analysis of neural networks using symbolic intervals. In *27th USENIX Security Symposium (USENIX) Security 18)*, pages 1599–1614, 2018.

[389] S. Wang, H. Zhang, K. Xu, X. Lin, S. Jana, C.-J. Hsieh, and J. Z. Kolter. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. *Advances in Neural Information Processing Systems*, 34, 2021.

[390] Y. Wang, Y. Li, Q. Zhang, J. Hu, and X. Kuang. Evading pdf malware classifiers with generative adversarial network. In *2019 International Symposium on Cyberspace Safety and Security (CSS)*, pages 374–387. Springer International Publishing, 2019.

[391] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5): 1–12, 2019.

[392] Y. Wang, E. Sarkar, W. Li, M. Maniatakos, and S. E. Jabari. Stop-and-go: Exploring backdoor attacks on deep reinforcement learning-based traffic congestion control systems. *arXiv preprint arXiv:2003.07859*, 2020.

[393] Y. Wang, H. Zhang, H. Chen, D. Boning, and C.-J. Hsieh. On lp-norm robustness of ensemble decision stumps and trees. In *International Conference on Machine Learning*, pages 10104–10114. PMLR, 2020.

[394] Y. Wang, Z. Shi, Q. Gu, and C.-J. Hsieh. On the convergence of certified robust training with interval bound propagation. In *International Conference on Learning Representations*, 2022.

[395] Z. Wang, M. Yan, J. Chen, S. Liu, and D. Zhang. Deep learning library testing via effective model generation. In *28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE*, pages 788–799. ACM, 2020.

[396] Z. Wang, A. Albarghouthi, G. Prakriya, and S. Jha. Interval universal approximation for neural networks. *Proceedings of the ACM on Programming Languages*, 6(POPL): 1–29, 2022.

[397] M. Wardat, W. Le, and H. Rajan. DeepLocalize: Fault localization for deep neural networks. In *43rd IEEE/ACM International Conference on Software Engineering, ICSE*, pages 251–262. IEEE, 2021.

[398] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[399] M. Weber, L. Li, B. Wang, Z. Zhao, B. Li, and C. Zhang. Certifying out-of-domain generalization for blackbox functions. In *International Conference on Machine Learning*. PMLR, 2022.

[400] M. Weber, X. Xu, B. Karlas, C. Zhang, and B. Li. Rab: Provable robustness against backdoor attacks. In *44th IEEE Symposium on Security and Privacy, SP 2023, San Francisco, CA, USA, 22-26 May 2023*. IEEE, 2023.

[401] K. Weierstrass. Über die analytische darstellbarkeit sogenannter willkürlicher functionen einer reellen veränderlichen. *Sitzungsberichte der Königlich Preußischen Akademie der Wissenschaften zu Berlin*, 2:633–639, 1885.

[402] L. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, L. Daniel, D. Boning, and I. Dhillon. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pages 5276–5285, 2018.

[403] L. F. Wightman. Lsac national longitudinal bar passage study. lsac research report series. *Law School Admission Council*, 1998.

[404] R. Winter, F. Montanari, F. Noé, and D.-A. Clevert. Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. *Chemical science*, 10(6):1692–1701, 2019.

[405] E. Wong and J. Z. Kolter. Learning perturbation sets for robust machine learning. In *International Conference on Learning Representations*, 2020.

[406] E. Wong and Z. Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5286–5295, 2018.

[407] E. Wong, F. Schmidt, J. H. Metzen, and J. Z. Kolter. Scaling provable adversarial defenses. In *Advances in Neural Information Processing Systems*, pages 8400–8409, 2018.

[408] F. Wu, L. Li, Z. Huang, Y. Vorobeychik, D. Zhao, and B. Li. CROP: Certifying robust policies for reinforcement learning through functional smoothing. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=HOjLHrlZhmx`.

[409] F. Wu, L. Li, C. Xu, H. Zhang, B. Kailkhura, K. Kenthapadi, D. Zhao, and B. Li. COPA: Certifying robust policies for offline reinforcement learning against poisoning attacks. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=psh0oeMSBiF`.

[410] Y. Wu, A. Bojchevski, A. Kuvshinov, and S. Günnemann. Completing the picture: Randomized smoothing suffers from the curse of dimensionality for a large family of distributions. In A. Banerjee and K. Fukumizu, editors, *The 24th International Conference on Artificial Intelligence and Statistics, AISTATS 2021, April 13-15, 2021, Virtual Event*, volume 130 of *Proceedings of Machine Learning Research*, pages 3763–3771. PMLR, 2021.

[411] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[412] C. Xiang, C. R. Qi, and B. Li. Generating 3d adversarial point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9136–9144, 2019.

[413] C. Xiang, A. N. Bhagoji, V. Sehwag, and P. Mittal. PatchGuard: A provably robust defense against adversarial patches via small receptive fields and masking. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2237–2254, 2021.

[414] C. Xiang, S. Mahloujifar, and P. Mittal. PatchCleanser: Certifiably robust defense against adversarial patches for any image classifier. In *31st USENIX Security Symposium (USENIX Security)*, 2022.

[415] T. Xiang, C. Zhang, Y. Song, J. Yu, and W. Cai. Walk in the cloud: Learning curves for point clouds shape analysis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 915–924, 2021.

[416] C. Xiao, J.-Y. Zhu, B. Li, W. He, M. Liu, and D. Song. Spatially transformed adversarial examples. In *2018 International Conference on Learning Representations (ICLR)*. OpenReview, 2018.

[417] C. Xiao, D. Yang, B. Li, J. Deng, and M. Liu. Meshadv: Adversarial meshes for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6898–6907, 2019.

[418] C. Xiao, Z. Chen, K. Jin, J. Wang, W. Nie, M. Liu, A. Anandkumar, B. Li, and D. Song. Densepure: Understanding diffusion models for adversarial robustness. In *International Conference on Learning Representations*, 2023.

[419] K. Y. Xiao, V. Tjeng, N. M. M. Shafiullah, and A. Madry. Training for faster adversarial robustness verification via inducing reLU stability. In *International Conference on Learning Representations*, 2019.

[420] Y. Xiao, I. Beschastnikh, D. S. Rosenblum, C. Sun, S. Elbaum, Y. Lin, and J. S. Dong. Self-checking deep neural networks in deployment. In *43rd IEEE/ACM International Conference on Software Engineering, ICSE*, pages 372–384. IEEE, 2021.

[421] D. Xie, Y. Li, M. Kim, H. V. Pham, L. Tan, X. Zhang, and M. W. Godfrey. Leveraging documentation to test deep learning library functions. *CoRR*, abs/2109.01002, 2021.

[422] Y. Xie, Z. Xu, M. S. Kankanhalli, K. S. Meel, and H. Soh. Embedding symbolic knowledge into deep networks. *Advances in neural information processing systems*, 32, 2019.

[423] Y. Xiong, Y. Tian, Y. Liu, and S. Cheung. Toward actionable testing of deep learning models. *Science China, Information Sciences*, 2022. Online first: 2022-09-26.

[424] K. Xu, Z. Shi, H. Zhang, Y. Wang, K.-W. Chang, M. Huang, B. Kailkhura, X. Lin, and C.-J. Hsieh. Automatic perturbation analysis for scalable certified robustness and beyond. *Advances in Neural Information Processing Systems*, 33, 2020.

[425] K. Xu, H. Zhang, S. Wang, Y. Wang, S. Jana, X. Lin, and C.-J. Hsieh. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In *International Conference on Learning Representations*, 2021.

[426] M. Yan, J. Chen, X. Zhang, L. Tan, G. Wang, and Z. Wang. Exposing numerical bugs in deep learning via gradient back-propagation. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 627–638, 2021.

[427] G. Yang, T. Duan, J. E. Hu, H. Salman, I. P. Razenshteyn, and J. Li. Randomized smoothing of all shapes and sizes. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 10693–10705. PMLR, 2020.

[428] H. Yang, J. Zhang, H. Dong, N. Inkawhich, A. Gardner, A. Touchet, W. Wilkes, H. Berry, and H. Li. Dverge: Diversifying vulnerabilities for enhanced robust generation of ensembles. *Advances in Neural Information Processing Systems*, 33, 2020.

[429] Z. Yang, L. Li, X. Xu, S. Zuo, Q. Chen, P. Zhou, B. I. P. Rubinstein, C. Zhang, and B. Li. Trs: Transferability reduced ensemble via promoting gradient diversity and model smoothness. In *Advances in Neural Information Processing Systems*, 2021.

[430] Z. Yang, L. Li, X. Xu, B. Kailkhura, T. Xie, and B. Li. On the certified robustness for ensemble models and beyond. In *International Conference on Learning Representations*, 2022.

[431] Z. Yang, L. Li, X. Xu, B. Kailkhura, T. Xie, and B. Li. On the certified robustness for ensemble models and beyond. In *International Conference on Learning Representations (ICLR)*, 2022. URL `https://openreview.net/forum?id=tUa4REjGjTf`.

[432] M. Ye, C. Gong, and Q. Liu. Safer: A structure-free approach for certified robustness to adversarial word substitutions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3465–3475, 2020.

[433] W. Ye, S. Liu, T. Kurutach, P. Abbeel, and Y. Gao. Mastering atari games with limited data. *Advances in Neural Information Processing Systems*, 34, 2021.

[434] Y. Ye, H. Pei, B. Wang, P.-Y. Chen, Y. Zhu, J. Xiao, and B. Li. Reinforcement-learning based portfolio management with augmented asset movement prediction states. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1112–1119, 2020.

[435] S. Yeom and M. Fredrikson. Individual fairness revisited: transferring techniques from adversarial robustness. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence*, pages 437–443, 2021.

[436] R. Zemel, Y. Wu, K. Swersky, T. Pitassi, and C. Dwork. Learning fair representations. In *International conference on machine learning*, pages 325–333. PMLR, 2013.

[437] R. Zhai, C. Dan, D. He, H. Zhang, B. Gong, P. Ravikumar, C.-J. Hsieh, and L. Wang. Macer: Attack-free and scalable robust training via maximizing certified radius. In *International Conference on Learning Representations*, 2020.

[438] B. Zhang, T. Cai, Z. Lu, D. He, and L. Wang. Towards certifying l-infinity robustness using neural networks with l-inf-dist neurons. In M. Meila and T. Zhang, editors, *International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 12368–12379. PMLR, 18–24 Jul 2021.

[439] B. Zhang, D. Jiang, D. He, and L. Wang. Boosting the certified robustness of l-infinity distance nets. In *International Conference on Learning Representations*, 2022.

[440] B. Zhang, D. Jiang, D. He, and L. Wang. Boosting the certified robustness of l-infinity distance nets. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=Q76Y7wkiji`.

[441] B. Zhang, D. Jiang, D. He, and L. Wang. Rethinking lipschitz neural networks and certified robustness: A boolean function perspective. *Advances in Neural Information Processing Systems*, 35:19398–19413, 2022.

[442] D. Zhang, M. Ye, C. Gong, Z. Zhu, and Q. Liu. Black-box certification with randomized smoothing: A functional optimization based framework. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 2316–2326. Curran Associates, Inc., 2020.

[443] H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel. Efficient neural network robustness certification with general activation functions. In *Advances in neural information processing systems*, pages 4939–4948, 2018.

[444] H. Zhang, M. Cheng, and C.-J. Hsieh. Enhancing certifiable robustness via a deep model ensemble. *arXiv preprint arXiv:1910.14655*, 2019.

[445] H. Zhang, P. Zhang, and C.-J. Hsieh. Recurjac: An efficient recursive algorithm for bounding jacobian matrix of neural networks and its applications. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 5757–5764, 2019.

[446] H. Zhang, H. Chen, C. Xiao, S. Gowal, R. Stanforth, B. Li, D. Boning, and C.-J. Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *International Conference on Learning Representations*, 2020.

[447] H. Zhang, H. Chen, D. S. Boning, and C.-J. Hsieh. Robust reinforcement learning on state observations with learned optimal adversary. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=sCZbhBvqQaU`.

[448] H. Zhang, S. Wang, K. Xu, L. Li, B. Li, S. Jana, C.-J. Hsieh, and J. Z. Kolter. General cutting planes for bound-propagation-based neural network verification. In *Advances in Neural Information Processing Systems 35 (NeurIPS)*, 2022.

[449] J. Zhang, L. Li, H. Li, X. Zhang, S. Yang, and B. Li. Progressive-scale boundary blackbox attack via projective gradient estimation. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning (ICML 2021)*, volume 139 of *Proceedings of Machine Learning Research*, pages 12479–12490. PMLR, 18–24 Jul 2021.

[450] J. Zhang, L. Li, C. Zhang, and B. Li. CARE: Certifiably robust learning with reasoning via variational inference. In *First IEEE Conference on Secure and Trustworthy Machine Learning*, 2023. URL `https://openreview.net/forum?id=1n6oWTTV1n`.

[451] J. M. Zhang, M. Harman, L. Ma, and Y. Liu. Machine learning testing: Survey, landscapes and horizons. *IEEE Transactions on Software Engineering*, 48(2):1–36, 2022.

[452] T. Zhang, C. Gao, L. Ma, M. Lyu, and M. Kim. An empirical study of common challenges in developing deep learning applications. In *2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE)*, pages 104–115. IEEE, 2019.

[453] X. Zhang, Y. Ma, A. Singla, and X. Zhu. Adaptive reward-poisoning attacks against reinforcement learning. In *International Conference on Machine Learning*, pages 11225–11234. PMLR, 2020.

[454] X. Zhang, R. Tu, Y. Liu, M. Liu, H. Kjellstrom, K. Zhang, and C. Zhang. How do fair decisions fare in long-term qualification? *Advances in Neural Information Processing Systems*, 33:18457–18469, 2020.

[455] X. Zhang, J. Zhai, S. Ma, and C. Shen. AutoTrainer: An automatic DNN training problem detection and repair system. In *43rd IEEE/ACM International Conference on Software Engineering, ICSE*, pages 359–371. IEEE, 2021.

[456] X. Zhang, Y. Chen, X. Zhu, and W. Sun. Corruption-robust offline reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pages 5757–5773. PMLR, 2022.

[457] Y. Zhang, Y. Chen, S.-C. Cheung, Y. Xiong, and L. Zhang. An empirical study on tensorflow program bugs. In *Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 129–140, 2018.

[458] Y. Zhang, X. Chen, Y. Yang, A. Ramamurthy, B. Li, Y. Qi, and L. Song. Efficient probabilistic logic reasoning with graph neural networks. In *International Conference on Learning Representations*, 2019.

[459] Y. Zhang, A. Albarghouthi, and L. D'Antoni. Robustness to programmable string transformations via augmented abstract training. In *International Conference on Machine Learning*, 2020.

[460] Y. Zhang, L. Ren, L. Chen, Y. Xiong, S. Cheung, and T. Xie. Detecting numerical bugs in neural network architectures. In *28th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE*, pages 826–837. ACM, 2020.

[461] Y. Zhang, A. Albarghouthi, and L. D'Antoni. Certified robustness to programmable transformations in LSTMs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1068–1083, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

[462] H. Zhao and G. Gordon. Inherent tradeoffs in learning fair representations. *Advances in neural information processing systems*, 32, 2019.

[463] H. Zhao, A. Coston, T. Adel, and G. J. Gordon. Conditional learning of fair representations. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=Hkekl0NFPr`.

[464] Y. Zhao, H. Zhu, R. Liang, Q. Shen, S. Zhang, and K. Chen. Seeing isn't believing: Towards more robust adversarial attack against real world object detectors. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 1989–2004, 2019.

[465] Y. Zhou and O. Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4490–4499, 2018.

[466] L. Zhu, Z. Liu, and S. Han. Deep leakage from gradients. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14747–14756, 2019.

[467] Q. Zhu, Y. Li, H. Hu, and B. Wu. Robust point cloud classification based on multi-level semantic relationships for urban scenes. *ISPRS journal of photogrammetry and remote sensing*, 129:86–102, 2017.

[468] D. Zombori, B. Bánhelyi, T. Csendes, I. Megyeri, and M. Jelasity. Fooling a complete neural network verifier. In *International Conference on Learning Representations*, 2020.