# Provably Robust Machine Learning through Structure-Aware Computation

by

Brendon G. Anderson

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering—Mechanical Engineering

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Associate Professor Somayeh Sojoudi, Chair
Associate Professor Javad Lavaei
Professor Kameshwar Poolla

Spring 2024

Provably Robust Machine Learning through Structure-Aware Computation

Abstract

Provably Robust Machine Learning through Structure-Aware Computation

by

Brendon G. Anderson

Doctor of Philosophy in Engineering—Mechanical Engineering

University of California, Berkeley

Associate Professor Somayeh Sojoudi, Chair

Standard machine learning (ML) algorithms exhibit catastrophic failures when subject to uncertainties in their input data, such as attacks generated by an adversary. Robustness against such uncertainties must be guaranteed in order to reliably deploy ML in safety-critical settings, such as autonomous driving, healthcare, and the operation of power systems. This dissertation presents theoretical and computational advancements in provably robust machine learning. We both introduce efficient optimization methods to certify the robustness of prior ML models, as well as design novel ML models endowed with mathematical proof of robustness. By exploiting key structures in the underlying certification problems, the proposed methods achieve state-of-the-art robustness and efficiency.

In the first part of this dissertation, we consider certifying the robustness of given, pretrained machine learning models. This robustness certification problem amounts to solving a difficult nonconvex optimization problem, and therefore a more tractable approach for generating safety guarantees is to lower-bound the optimization. We begin by considering a branch-and-bound approach to computing such lower bounds. In doing so, we leverage the piecewise linear structure of ReLU neural networks to develop branching schemes that minimize the looseness of the desired lower bounds in a worst-case sense. Next, we show that ReLU neural networks may be rewritten in a min-max affine form. We prove that this min-max affine structure allows us to efficiently solve the nonconvex robustness certification problem to global optimality using off-the-shelf convex solvers. We also consider certifying the robustness of models in the case where the inputs are subject to random noise. A data-driven, convex optimization-based method is developed that simultaneously localizes neural network outputs and verifies their safety, all with high-probability guarantees. We show that our data-driven method's sample complexity can be dramatically reduced by leveraging the compositional structure of neural networks.

In the second part of this dissertation, we consider the design of robust machine learning models that are capable of withstanding uncertainties and attacks in their inputs, and

are amenable to robustness certification. First, we propose *feature-convex neural networks*, which consist of the composition of a Lipschitz continuous feature map followed by a convex neural network. We utilize this composite convex structure of our model to derive deterministic, closed-form robustness certificates that match or outperform prior provably robust ML methods. Finally, we introduce *locally biased randomized smoothing* as a means to robustify an otherwise general, non-robust pretrained classifier. Our model inherits a nonlinear interpolation structure from which we prove certified robustness guarantees, and empirically show an enhancement in robustness against adversarial attacks.

*To my parents.*

# Contents

# List of Figures

# List of Tables

# Acknowledgments

This dissertation would not have been possible without the unwavering support of my mentors, colleagues, friends, and family.

I would first like to extend my deepest gratitude to my advisor, Somayeh Sojoudi. Somayeh has always encouraged me to explore research ideas that genuinely sparked my interests, making my time at UC Berkeley refreshing and full of intellectual excitement. Beyond her incredible guidance as a research mentor, Somayeh has went above and beyond in supporting me through my professional development and through navigating the academic process, and for this I am forever grateful. Above all, I would like to thank Somayeh for her compassion throughout all of the professional and personal challenges a Ph.D. has to offer.

I am also deeply indebted to Professors Javad Lavaei and Murat Arcak, both of whom have greatly supported me in achieving my academic goals. I also thank Javad for kindly bringing me into his optimization course's teaching team, where I was able to learn new pedagogical skills and progress as an educator. I thank Murat for introducing me to the field of evolutionary game theory, and for enthusiastically guiding me through the broadening of my research agenda to include this area of interest. I would also like to express my utmost gratitude to Professor Kameshwar Poolla, for graciously serving on my dissertation committee and supporting my success as a graduate student. As important as my mentors at UC Berkeley are my incredibly talented colleagues and peers with whom I have worked over the years: Samuel Pfrommer, Ziye Ma, Jingqi Li, Yatong Bai, Tanmay Gautam, Eli Brock, Hyunin Lee, Elizabeth Glista, Fernando Gama, Salar Fattahi, Richard Y. Zhang, Cédric Josz, Nan Tian, and Siddharth Nair.

I would also like to acknowledge Professor Robert M'Closkey at UCLA, for piquing my initial interest in control theory and dynamical systems during my undergraduate studies, inviting me to work in his research lab, and motivating me to pursue graduate school at UC Berkeley. This Ph.D. would not have been possible without his encouragement and uplifting support. I would additionally like to thank Professor Matt Haberland at Cal Poly. While we were both at UCLA, Matt was generous enough to mentor me and a wonderful group of other undergraduate students on a summer research project. It was under Matt's mentorship that I learned what fundamental research is all about—an experience which solified my desire to continue on to a Ph.D. I also express my utmost gratitude to Professor Matthew Powell-Palm at Texas A&M, for not only posing fun, thermodynamics-driven mathematics problems to me that broaden my horizons, but also for being a part of my professional support system and always creating a cheerful and friendly environment.

My time at UC Berkeley was not only some of my most fulfilling academically, but personally as well. I thank all of my amazing friends for being my biggest advocates and for filling my time during graduate school with uncountably many fond memories: Paul, Austin, Augie, Tony, Colin, Jared, Page, Max, Elliot, David, and the rest of the 510 family. To my parents, brother, and family: words cannot describe how much I appreciate the unconditional love and support you have given me throughout my life. None of my opportunities for growth and success would have been possible without you by my side. I owe the world to my parents.

Last but not least, I express my deepest gratitude to my love, Lauren Giggy. I cannot thank you enough for your unwavering fortification. Building such a special life with you is hands-down my most fulfilling accomplishment of graduate school.

# Chapter 1

# Introduction

Real-world data is inherently uncertain. Such uncertainty takes a variety of forms, including corruptions, random measurement noise, and adversarial attacks [57, 20, 71]. Recent research has found that the performance of machine learning models can be highly sensitive to these uncertainties in the input data [130, 47, 128]. For example, Kumar et al. [83] showed that maliciously designed, human-imperceptible perturbations to image data are capable of fooling autonomous vehicle perception systems into misclassifying a stop sign as a yield sign. More recently, publicly available AI systems have been broken: Zou et al. [174] persuaded Chat GPT into generating a step-by-step plan to destroy humanity, in spite of the system's built-in safeguards. Examples of this sort demonstrate that, even though machine learning models may attain state-of-the-art performance on complex data-driven tasks, they are susceptible of exhibiting *catastrophic* performance in the presence of an adversary or corrupted input data. Such sensitivity is unacceptable in safety-critical machine learning applications such as autonomous driving [23, 151] and the operation of power systems [78, 104, 110]. This has motivated the need for provably robust machine learning systems, i.e., systems with mathematical guarantees that they perform reliably in the presence of input uncertainties.

To formalize robustness of machine learning systems, consider a classifier given by $f(x) \in \arg\max_{i\in\{1,\dots,n\}} g_i(x)$ with $g\colon \mathbb{R}^d \to \mathbb{R}^n$, and a region $X \subseteq \mathbb{R}^d$. The region $X$ represents a threat model; it is the set of all possible uncertain or attacked inputs that $f$ may receive. Formally, the classifier $f$ is said to be *certifiably robust over* $X$ if there exists some class $y \in \{1,\dots,n\}$ such that $f(x) = y$ for all $x \in X$. That is, robustness of the model amounts to consistent prediction over the set $X$ of viable input data. Commonly, the input uncertainty set $X$ is taken as an $\ell_p$-norm ball centered around some nominal input data point $\overline{x} \in \mathbb{R}^d$, to model the situation in which an adversary is designing a "stealth," or human-imperceptible attack.

Verifying that a classifier is robust over an input uncertainty set $X$ typically amounts to solving the *robustness certification problem*, given by the following optimization:

$$p_i^\star \coloneqq \inf_{x\in X} \big(g_y(x) - g_i(x)\big).$$

If $p_i^\star \geq 0$ for all $i \neq y$, then indeed $f(x) = y$ for all $x \in X$, and hence $f$ is certifiably robust over $X$. Such optimization-based approaches to robustness certification can be extended to regression settings and situations in which the input data is corrupted by random noise—these alternative notions of robustness will be formalized in later chapters. In the case that $f$ is certifiably robust over a norm-ball input uncertainty set $X = \{x \in \mathbb{R}^d : \|x - \overline{x}\| \leq r\}$, we call $r$ a *certified radius*. In general, we hope for our systems to withstand as much uncertainty as possible (without sacrificing predictive performance on clean data), and hence seek to certify robustness over the largest possible uncertainty set $X$.

There are two key challenges that arise when dealing with certifiable robustness of machine learning systems:

1. **Computing robustness certificates.** The robustness certification problem is, in general, a nonconvex optimization problem, and is therefore challenging to solve to global optimality. If, however, one computes a lower bound $\tilde{p}_i \leq p_i^\star$ and verifies that $\tilde{p}_i \geq 0$, then the model is still guaranteed to be robust over $X$. A handful of approaches have been popularized to compute such lower bounds, ranging from branch-and-bound techniques [25, 39, 143], to bounding the Lipschitz constant of $g$ [65, 51], to developing convex relaxations of the nonconvex certification problem [148, 150, 116]. However, these methods have been found to span an efficiency-conservatism tradeoff. For example, branch-and-bound schemes can offer very tight lower bounds on $p_i^\star$, but their worst-case computational cost grows exponentially with the size of the problem. On the other hand, off-the-shelf linear programming (LP) solvers can efficiently solve LP relaxations of the certification problem, but the resulting lower bounds are over-conservative (loose) in high-dimensional, deep learning settings [123]. If such a lower bound is too loose, it may become negative, voiding the robustness certificate altogether. This motivates the need for novel techniques to efficiently lower-bound the certification problem with as minimal conservatism as possible.

2. **Designing robust models.** Even if the certification problem was able to be solved to global optimality, this would not imply that the model under consideration is actually robust. That is, $p_i^\star$ may be negative for an overly sensitive model. Designing models that simultaneously exhibit high performance on clean data as well as robust prediction on uncertain data has remained a major challenge in the field. Various approaches have been proposed, such as adversarial training (training on attacked data) [95], and Jacobian regularization [67] and randomized smoothing [85, 35] methods that aim to smooth out the sensitivities in the decision landscape of a classifier. However, current approaches to robustifying models have been shown to suffer from an accuracy-robustness tradeoff [137, 167], resulting in the call for new models that attain high accuracy in both the clean and uncertain data regimes.

This dissertation aims at tackling these two key challenges. The underlying methodology taken in this work is to leverage structure in models and the robustness certification problem as a means to resolve the efficiency-conservatism and accuracy-robustness tradeoffs.

## Leveraging Structure in Robust Machine Learning

Much of machine learning is highly structured. For instance, neural networks are defined by compositional structures, ReLU activation functions have piecewise linear structure, and even data distributions encode input-output structures that can be learned from. This dissertation makes use of these structures to develop novel computational techniques for solving and lower-bounding the robustness certification problem (Part I), as well as novel machine learning models with robustness-enhancing structures built into them (Part II). This structure-aware approach to computation is the key theme underlying each of the following chapters. In Chapter 2, the piecewise linear structure of ReLU neural networks is utilized to develop novel branch-and-bound techniques that efficiently tighten lower bounds on the certification problem. Min-max affine structures for ReLU networks are introduced in Chapter 3, and are shown to result in globally optimal solutions to the certification problem. By leveraging the input-output structure of samples through a neural network in conjunction with its compositional structure, an efficient data-driven approach to certification is developed in Chapter 4. In Chapter 5, a new model is introduced, whose composite convex structure results in closed-form certified radii that meet or exceed prior state-of-the-art methods. Finally, a nonlinear interpolation structure is introduced in Chapter 6 to robustify general, unstructured models. The primary contributions of each of these chapters are now summarized.

## Summary of Contributions

This dissertation is organized into two parts, each focusing on one of the primary challenges of robust machine learning introduced above. The first part, comprised of three chapters, is focused on computing robustness certificates. The common underlying contributions of these chapters include novel optimization methods that leverage structure in network architecture and data distributions to efficiently compute state-of-the-art lower bounds on the robustness certification problem. The second part, comprised of two chapters, centers around the design of robust machine learning models. These chapters focus on endowing models with robustness-enhancing structures that are amenable to efficient certification.

### Part I: Computing Robustness Certificates

In Chapter 2, branch-and-bound methods for lower-bounding the robustness certification problem for ReLU neural networks are considered. We consider the popular linear and semidefinite programming-based methods for bounding the problem. The focus of our work in this chapter is on determining a branching scheme that results in minimal relaxation error when employing these bounding methods. We show that computing an optimal branching strategy is NP-hard. Consequently, we leverage the structure of the ReLU activation functions to derive upper-bounds on the relaxation error incurred by these bounding methods in

a worst-case sense. We then derive worst-case optimal branching schemes that minimize our upper bounds in closed form.

In Chapter 3, we ask ourselves whether we can do even better than just tightening lower bounds on the certification problem. Namely, can we *exactly* solve the nonconvex certification problem to global optimality? In this chapter, we show that this is possible by changing our representation of ReLU neural networks into a min-max affine form. Specifically, we leverage the min-max structure to prove that the nonconvex certification problem over a convex input uncertainty set is equivalent to a readily-solvable convex optimization problem, under mild technical assumptions. As a result, we are able to solve the certification problem for such min-max representations to global optimality using off-the-shelf convex solvers, which we show outperform branch-and-bound methods both in terms of certification performance and speed.

In Chapter 4, we consider data-driven robustness certification in the case that models are subject to random uncertainties in their inputs. We develop a scenario optimization approach that is capable of simultaneously localizing model outputs as well as certifying their safety with high probability, given input-output samples from the model. Conditions for the convexity of the resulting optimization problem are proven. We show that, by exploiting the compositional structure of neural networks, one may apply our method to shallower "surrogate" models to dramatically reduce certification runtime, while maintaining high-performing robustness certificates for the true model under consideration.

**Part II: Designing Robust Models**

In Chapter 5, we propose *feature-convex models*, which consist of the composition of a Lipschitz continuous feature map followed by a convex function. By leveraging the composite convex structure of these models, we derive closed-form certified radii that scale up with model confidence and scale down with the Lipschitz constant of the feature map. This form of the robustness certificate motivates us to choose simple, low-Lipschitz feature maps, and to learn the convex portion of the model from data, using a nonnegatively weighted ReLU model with feedthrough layers. We characterize the decision region geometry of these feature-convex neural networks and prove that they are universal approximators of convex classifiers. We also show that their composite convex structure is not restrictive in practice, as they are able to achieve 100% training accuracy on the benchmark CIFAR-10 cats-dogs image dataset, and are even able to separate highly unstructured, uniformly distributed data on a cube with probability that increases exponentially in the data's underlying dimension. Our model's certified radii are shown to meet or exceed prior state-of-the-art methods on benchmark image and malware classification datasets.

In Chapter 6, we propose *locally biased randomized smoothing*, which robustifies a given pretrained classifier by manipulating its decision boundaries via convolution with a "smoothing distribution." We show that prior smoothing approaches, which use unbiased smoothing distributions that are independent of the input, cannot robustify even the simplest linear classifiers, and always work to lower the model's Lipschitz constant, even if the underlying

data distribution indicates the need for an increase in nonlinearity to robustify predictions. We propose to generalize smoothing to allow for biased, input-dependent smoothing distributions. We optimize such a smoothing distribution for first-order approximations of the base classifier in closed-form—without the need for sample expectations—which we show is biased and locally pushes the classifier's decision boundary along the span of its gradient. Upon approximating the correct direction of this decision boundary manipulation using a 1-nearest neighbor classifier, we are able to prove interpretable certified radii for our models. The structure of our model is shown to result in nonlinear interpolation between the base classifier and the 1-nearest neighbor classifier, which we empirically find enhances robustness on benchmark image datasets.

### Related Publications

This dissertation contains materials from the following publications.

#### Chapter 2.

[5] Brendon G. Anderson, Ziye Ma, Jingqi Li, and Somayeh Sojoudi, "Towards optimal branching of linear and semidefinite relaxations for neural network robustness certification," *Submitted*, 2023.

[4] Brendon G. Anderson, Ziye Ma, Jingqi Li, and Somayeh Sojoudi, "Tightened convex relaxations for neural network robustness certification," *IEEE Conference on Decision and Control (CDC)*, 2020.

#### Chapter 3.

[6] Brendon G. Anderson, Samuel Pfrommer, and Somayeh Sojoudi, "Tight certified robustness via min-max representations of ReLU neural networks," *IEEE Conference on Decision and Control (CDC)*, 2023.

#### Chapter 4.

[9] Brendon G. Anderson and Somayeh Sojoudi, "Data-driven certification of neural networks with random input noise," *IEEE Transactions on Control of Network Systems*, 2022.

#### Chapter 5.

[113] Samuel Pfrommer*, Brendon G. Anderson*, Julien Piet, and Somayeh Sojoudi, "Asymmetric certified robustness via feature-convex neural networks," *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. *Co-first author and equal contribution.

**Chapter 6.**

[8] Brendon G. Anderson and Somayeh Sojoudi, "Certified robustness via locally biased randomized smoothing," *Learning for Dynamics and Control (L4DC)*, 2022.

See also the following publications that are related to Chapter 6:

[16] Yatong Bai, Brendon G. Anderson, Aerin Kim, and Somayeh Sojoudi, "Improving the accuracy-robustness trade-off of classifiers via adaptive smoothing," *SIAM Journal on Mathematics of Data Science*, 2024.

[17] Yatong Bai, Brendon G. Anderson, and Somayeh Sojoudi, "Mixing classifiers to alleviate the accuracy-robustness trade-off," *Learning for Dynamics and Control (L4DC)*, 2024.

[111] Samuel Pfrommer, Brendon G. Anderson, and Somayeh Sojoudi, "Projected randomized smoothing for certified adversarial robustness," *Transactions on Machine Learning Research*, 2023.

[7] Brendon G. Anderson, Samuel Pfrommer, and Somayeh Sojoudi, "Towards optimal randomized smoothing: A semi-infinite linear programming approach," *ICML Workshop on Formal Verification of Machine Learning (WFVML)*, 2022.

# Part I

# Computing Robustness Certificates

# Chapter 2

# Towards Optimal Branching for ReLU Neural Networks

In this chapter, we study certifying the robustness of ReLU neural networks against adversarial input perturbations. To diminish the relaxation error suffered by the popular linear programming (LP) and semidefinite programming (SDP) certification methods, we take a branch-and-bound approach to propose partitioning the input uncertainty set and solving the relaxations on each part separately. We show that this approach reduces relaxation error, and that the error is eliminated entirely upon performing an LP relaxation with a partition intelligently designed to exploit the nature of the ReLU activations. To scale this approach to large networks, we consider using a coarser partition whereby the number of parts in the partition is reduced. We prove that computing such a coarse partition that directly minimizes the LP relaxation error is NP-hard. By instead minimizing the worst-case LP relaxation error, we develop a closed-form branching scheme. We extend the analysis to the SDP, where the feasible set geometry is exploited to design a branching scheme that minimizes the worst-case SDP relaxation error. Numerical simulations on MNIST, CIFAR-10, and Wisconsin breast cancer diagnosis classifiers demonstrate significant increases in the percentages of test samples certified. By independently increasing the input size and the number of layers, we empirically illustrate under which regimes the branched LP and branched SDP are best applied.

This chapter is based on the following submitted and published works:

[5] Brendon G. Anderson, Ziye Ma, Jingqi Li, and Somayeh Sojoudi, "Towards optimal branching of linear and semidefinite relaxations for neural network robustness certification," *Submitted*, 2023.

[4] Brendon G. Anderson, Ziye Ma, Jingqi Li, and Somayeh Sojoudi, "Tightened convex relaxations for neural network robustness certification," *IEEE Conference on Decision and Control (CDC)*, 2020.

## 2.1 Introduction

The two primary settings taken in the robustness certification literature consider either random input uncertainty or adversarial uncertainty. In the former, the neural network input is assumed to be random and follow a known probability distribution. For instance, the works Weng et al. [147] and Anderson and Sojoudi [9] (see also Chapter 4) derive high-probability guarantees that this randomness causes no misclassification or unsafe output. In the adversarial setting, the input is assumed to be unknown but contained in a prescribed input uncertainty set. The goal here is to certify that all possible inputs from the uncertainty set are mapped to outputs that the network operator deems as safe [150, 120]. In this chapter, we take the latter, worst-case perspective. We remark that this approach is more generally applicable than the techniques for random inputs. Indeed, certifying that a network is robust against adversarial inputs immediately also certifies it is robust against random inputs distributed over the same uncertainty set; if the worst-case input cannot cause a failure, then neither will randomly selected ones.

The problem of adversarial robustness certification amounts to proving that all possible outputs in the output set, i.e., the image of the input uncertainty set under the mapping of the network, are contained in the safe set. However, this output set is generally nonconvex, even when the input set is convex. Consequently, the certification decision problem has been shown to be NP-complete, and the optimization-based formulation for solving it is an NP-hard, nonconvex optimization problem [73, 148]. To make the problem more tractable, researchers have proposed various ways to over-approximate the nonconvex output set with a convex one. Performing the certification over the convex surrogate reduces the problem to an easy-to-solve convex relaxation, and if the relaxation issues a robustness certificate for the outer approximation, it also issues a certificate for the true set of outputs. Such convex relaxation-based certification algorithms are sound, but generally incomplete; the network may be robust even if the algorithm is unable to verify it. Thus, a line of works has been developed in order to increase the percentage of test inputs that convex relaxation-based methods are able to certify [150, 116, 50, 134, 103, 21, 143].

Perhaps the simplest and most popular outer approximation technique is based on a linear programming (LP) relaxation of the ReLU activation function [45, 150]. However, this method has been shown to yield relatively loose outer approximations, making it possible for the approximating set to contain unsafe outputs, even if the true output set is entirely safe. If this occurs, the convex relaxation fails to issue a certificate of robustness, despite the fact the network is indeed robust. A semidefinite programming (SDP) relaxation was proposed in Raghunathan, Steinhardt, and Liang [116] and shown to yield tighter outer approximations when compared to the LP method. Other methods, such as the quadratically-constrained semidefinite program [50], use sophisticated relaxations to tighten the approximation, but these SDP-based methods inevitably gain accuracy at the expense of enlarging the computational cost, and are still susceptible to relaxation error. The work Tjandraatmadja et al. [134] introduces a joint ReLU LP relaxation that is tighter than the per-neuron approach, but may require an exponential number of linear inequalities and is weaker overall in com-

parison to the per-neuron relaxation when embedded into a branch-and-bound scheme [143]. We follow the certification literature's emphasis on ReLU networks and focus our attention in this chapter on such models, which, aside from their certifiable structure, are extremely popular for their non-vanishing gradient property and their quick training times [154].

## Branch-and-Bound Certification

As eluded to above, instead of resorting to exorbitantly costly relaxation techniques to lower the approximation error, e.g., using heuristic variants of Lasserre's hierarchy [32], state-of-the-art convex relaxation-based verifiers have turned to branch-and-bound methods, where the nonconvex optimization domain is recursively partitioned and outer-approximated by smaller convex sets. In fact, the verifier that won the 2021 and 2022 VNN certification competition is $\alpha, \beta$-CROWN [18, 143], which is a branch-and-bound method that uses linear programming relaxations in the bounding steps. $\alpha, \beta$-CROWN uses $\alpha$-CROWN to generate linear upper and lower bounds on the neural network output, together with the heuristic per-neuron branching methods BaBSR [25] and filtered smart branching (FSB) [39].

The success of $\alpha, \beta$-CROWN is interesting, since the BaBSR and FSB branching strategies are designed in the context of Ehler's per-neuron "triangle" convex relaxations [45], which is a different and, when activation bounds are adequately chosen, tighter bounding approach than $\alpha$-CROWN's linear bounding of the output, albeit computationally more expensive. FSB remains the state-of-the-art branching heuristic, outperforming BaBSR [143, 39], which in turn beat a state-of-the-art mixed-integer approach and all prior ReLU neuron splitting methods: Reluplex, Planet, and Neurify [25, 73, 45, 142]. FSB works by assigning scores to each neuron that are computed by quickly estimating the improvement in the optimization objective upon splitting that neuron. In this chapter, we propose two branch-and-bound certification methods, one utilizing the triangle relaxation of Ehlers [45] as a bounding method and the other utilizing the SDP relaxation of Raghunathan, Steinhardt, and Liang [116], and for these bounding methods we derive novel branching schemes that minimize the worst-case relaxation error.

It is worth briefly mentioning here that branch-and-bound certification techniques can make use of parallel computing when solving independent subproblems arising from the feasible set partitioning [92, 153, 157]. This improves the scalability and efficiency of neural network certification. Our proposed branch-and-bound methods are naturally able to leverage such parallelizations.

## Contributions

In this chapter, we fully exploit the piecewise linear structure of ReLU networks in order to tighten convex relaxations for robustness certification. A condensed summary of our main contributions is as follows:

1. For the LP relaxation, we show that a methodically designed finite partition of the input set allows one to attain zero relaxation error. We then use the structure of this motivating partition to derive a closed-form branching scheme that minimizes worst-case relaxation error. This novel branching scheme makes theoretically principled strides towards optimal LP branching, in contrast to heuristic-based prior methods.

2. We prove that computing the branching neuron that minimizes the actual LP relaxation error is NP-hard, in turn theoretically justifying our approach of minimizing the worst-case relaxation error.

3. For the SDP relaxation, we develop a geometrically interpretable measure for how far the solution is from being rank-1. We then show that, when branching along a given direction, this rank-1 gap is minimized by a uniform grid. Finally, we derive the branching direction that minimizes the worst-case relaxation error.

4. We embed our branching scheme into a branch-and-bound framework, and empirically validate the effectiveness of our approach on the MNIST, CIFAR-10, and Wisconsin breast cancer diagnosis benchmarks. Specifically, we experimentally show on these datasets that our LP branching scheme outperforms the state-of-the-art filtered smart branching, the branching heuristic used in $\alpha, \beta$-CROWN to win the 2021 and 2022 VNN certification competitions. Our SDP branching scheme is found to yield even higher certified percentages.

## Outline

This chapter is organized as follows. Section 2.2 introduces the ReLU robustness certification problem as well as its basic LP and SDP relaxations. In Section 2.3, we study the effect of partitioning the input uncertainty set for the LP relaxation. After developing formal guarantees for its effectiveness, we develop a branching strategy that optimally reduces the worst-case relaxation error. Similarly, in Section 2.4 we study the partitioned SDP relaxation and propose a worst-case optimal branching scheme. Section 2.6 demonstrates the developed methods on numerical examples and studies the effectiveness of branching on the LP and SDP as the network grows in width and depth. Finally, we conclude in Section 2.7.

## Notations

For the reader's convenience, we list our commonly used symbols for this chapter in Table 2.1, some of which will be more rigorously defined in the following sections.

For an index set $\mathcal{I} \subseteq \{1, 2, \ldots, n\}$, the symbol $\mathbf{1}_{\mathcal{I}}$ is used to denote the $n$-vector with $(\mathbf{1}_{\mathcal{I}})_i = 1$ for $i \in \mathcal{I}$ and $(\mathbf{1}_{\mathcal{I}})_i = 0$ for $i \in \mathcal{I}^c = \{1, 2, \ldots, n\} \setminus \mathcal{I}$. For a function $f \colon \mathbb{R}^n \to \mathbb{R}^m$ and the point $x \in \mathbb{R}^n$, we denote the $i$th element of the $m$-vector $f(x)$ by $f_i(x)$. For a matrix $X \in \mathbb{R}^{m \times n}$, we use two indices to denote an element of $X$ and one index to denote a column of $X$, unless otherwise stated. In particular, the $(i, j)$ element and the $i$th column of $X$

are denoted by $X_{ij}$ and $X_i$ respectively, or, when necessary for clarity, by $(X)_{ij}$ and $(X)_i$ respectively. Furthermore, for a function $f\colon \mathbb{R} \to \mathbb{R}$, we define $f(X)$ to be an $m \times n$ matrix whose $(i, j)$ element is equal to $f(X_{ij})$ for all $i \in \{1, 2, \ldots, m\}$ and all $j \in \{1, 2, \ldots, n\}$. If $Z$ is a square $n \times n$ matrix, we use the notation $\operatorname{diag}(Z)$ to mean the $n$-vector $(Z_{11}, Z_{22}, \ldots, Z_{nn})$ and $\operatorname{tr}(Z)$ to mean the trace of $Z$. When $Z \in \mathbb{S}^n$, we write $Z \succeq 0$ to mean that $Z$ is positive semidefinite.

We use $\mathbb{I}$ to denote the indicator function, i.e., $\mathbb{I}(A) = 1$ if event $A$ holds and $\mathbb{I}(A) = 0$ if event $A$ does not hold. For a set $\mathcal{T} \subseteq \mathbb{R}$, we respectively denote its infimum and supremum by $\inf \mathcal{T}$ and $\sup \mathcal{T}$. For a function $f\colon \mathbb{R}^n \to \mathbb{R}$ and a set $\mathcal{X} \subseteq \mathbb{R}^n$, we write $\inf_{x \in \mathcal{X}} f(x)$ to mean $\inf\{f(x) : x \in \mathcal{X}\}$ and similarly for suprema. Finally, we assume that all infima and suprema throughout the paper are attained.

Table 2.1: List of commonly used symbols in Chapter 2.

| Symbol | Meaning |
|---|---|
| $\mathbb{R}^n$ | Set of $n$-vectors with real elements |
| $\mathbb{S}^n$ | Set of $n \times n$ matrices with real elements |
| $e_i$ | $i$th standard basis vector of $\mathbb{R}^n$ |
| $\mathbf{1}_n$ | $n$-vector of all ones |
| $X \leq Y$ | The array $X$ is element-wise less than or equal to the array $Y$ |
| $X \odot Y$ | Element-wise (Hadamard) product between arrays $X$ and $Y$ |
| $X \oslash Y$ | Element-wise (Hadamard) division of array $X$ by array $Y$ |
| $\lvert\mathcal{S}\rvert$ | Cardinality of the set $\mathcal{S}$ |
| $\lVert\cdot\rVert_*$ | Dual norm of the norm $\lVert\cdot\rVert$ |
| $d_{\lVert\cdot\rVert}(\mathcal{X})$ | Diameter of $\mathcal{X} \subseteq \mathbb{R}^n$ with respect to norm $\lVert\cdot\rVert$; $d_{\lVert\cdot\rVert}(\mathcal{X}) = \sup_{x,x' \in \mathcal{X}} \lVert x - x'\rVert$ |
| $f$ | Neural network |
| ReLU | Rectified linear unit; $\mathrm{ReLU}(\cdot) = \max\{0, \cdot\}$ |
| $x$, $z$ | Neural network input and output, respectively |
| $x^{[k]}$, $\hat{z}^{[k]}$ | Neural network activations and preactivations at layer $k$, respectively |
| $W^{[k]}$ | Neural network weights at layer $k$ |
| $l^{[k]}$, $u^{[k]}$ | Neural network preactivation lower and upper bounds at layer $k$, respectively |
| $\mathcal{X}$ | Neural network input uncertainty set |
| $f(\mathcal{X})$ | Neural network output uncertainty set given the input uncertainty set $\mathcal{X}$ |
| $\phi(\mathcal{X})$ | Optimal value of certification problem over the input uncertainty set $\mathcal{X}$ |

## 2.2 Problem Statement

### Description of the Network and Uncertainty

In this chapter, we consider a pretrained $K$-layer ReLU neural network defined by

$$\begin{aligned} x^{[0]} &= x, \\ \hat{z}^{[k]} &= W^{[k-1]}x^{[k-1]} + b^{[k-1]}, \\ x^{[k]} &= \mathrm{ReLU}(\hat{z}^{[k]}), \\ z &= x^{[K]}, \end{aligned} \tag{2.1}$$

for all $k \in \{1, 2, \ldots, K\}$. Here, the neural network input is $x \in \mathbb{R}^{n_x}$, the output is $z \in \mathbb{R}^{n_z}$, and the $k$th layer's preactivation is $\hat{z}^{[k]} \in \mathbb{R}^{n_k}$. The parameters $W^{[k]} \in \mathbb{R}^{n_{k+1} \times n_k}$ and $b^{[k]} \in \mathbb{R}^{n_{k+1}}$ are the weight matrix and bias vector applied to the $k$th layer's activation $x^{[k]} \in \mathbb{R}^{n_k}$, respectively. Without loss of generality,[1] we assume that the bias terms are accounted for in the activations $x^{[k]}$, thereby setting $b^{[k]} = 0$ for all layers $k$. We remark that, since we make no assumptions on the linear transformations defined by $W^{[k]}$, our work also applies to networks with linear convolutional layers. We let the function $f \colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$ denote the map $x \mapsto z$ defined by (2.1). In the case that $f$ is a classification network, the output dimension $n_z$ equals the number of classes. The problem at hand is to certify the robustness in the neural network output $z$ when the input $x$ is subject to uncertainty.

To model the input uncertainty, we assume that the network inputs are unknown but contained in a compact set $\mathcal{X} \subseteq \mathbb{R}^{n_x}$, called the *input uncertainty set*. We assume that the set $\mathcal{X}$ is a convex polytope, so that the condition $x \in \mathcal{X}$ can be written as a finite number of affine inequalities and equalities. In the adversarial robustness literature, the input uncertainty set is commonly modeled as $\mathcal{X} = \{x \in \mathbb{R}^{n_x} : \|x - \bar{x}\|_\infty \leq \epsilon\}$, where $\bar{x} \in \mathbb{R}^{n_x}$ is a nominal input to the network and $\epsilon > 0$ is an uncertainty radius [150, 116]. We remark that our generalized polytopic model for $\mathcal{X}$ includes this common case. The primary theme of this chapter revolves around partitioning the input uncertainty set in order to strengthen convex robustness certification methods. Let us briefly recall the definition of a partition.

**Definition 1** (Partition)**.** The collection $\{\mathcal{X}^{(j)} \subseteq \mathcal{X} : j \in \{1, 2, \ldots, p\}\}$ is said to be a *partition* of the input uncertainty set $\mathcal{X}$ if $\mathcal{X} = \cup_{j=1}^{p} \mathcal{X}^{(j)}$ and $\mathcal{X}^{(j)} \cap \mathcal{X}^{(k)} = \emptyset$ for all $j \neq k$. The set $\mathcal{X}^{(j)}$ is called the $j$th *input part*.

We generally use the terminology *branching* to refer to partitioning with two parts, which is typically applied in a recursive fashion.

Now, in order to describe the robustness of the network (2.1), we need a notion of permissible outputs. For this, we consider a *safe set*, denoted $\mathcal{S} \subseteq \mathbb{R}^{n_z}$. If an output $z \in \mathbb{R}^{n_z}$

[1] Note that $Wx + b = \begin{bmatrix} W & b \end{bmatrix} \begin{bmatrix} x \\ 1 \end{bmatrix} =: \tilde{W}\tilde{x}$, so the bias term $b$ can be eliminated by appending a fixed value of 1 at the end of the activation $x$. This parameterization can be used throughout this chapter by using matching lower and upper activation bounds of 1 in the last coordinate of each layer.

is an element of $\mathcal{S}$, we say that $z$ is *safe*. It is common in the robustness certification literature to consider (possibly unbounded) polyhedral safe sets. We take this same perspective, and assume that $\mathcal{S}$ is defined by

$$\mathcal{S} = \{z \in \mathbb{R}^{n_z} : Cz \leq d\},$$

where $C \in \mathbb{R}^{n_\mathcal{S} \times n_z}$ and $d \in \mathbb{R}^{n_\mathcal{S}}$ are given. Note that, again, our model naturally includes the classification setting. In particular, suppose that $i^\star \in \arg\max_{i \in \{1,2,\ldots,n_z\}} f_i(\bar{x})$ is the true class of the nominal input $\bar{x}$. Then, define the $i$th row of $C \in \mathbb{R}^{n_z \times n_z}$ to be

$$c_i^\top = e_i^\top - e_{i^\star}^\top$$

and $d$ to be the zero vector. Then it is immediately clear that an input $x$ (which can be thought of as a perturbed version of $\bar{x}$) is safe if and only if $f_i(x) \leq f_{i^\star}(x)$ for all $i$, i.e., the network classifies $x$ into class $i^\star$. From this classification perspective, the safe set represents the set of outputs without any misclassification (with respect to the nominal input $\bar{x}$). From here on, we consider $f$, $\mathcal{X}$, and $\mathcal{S}$ in their general forms—we do not restrict ourselves to the classification setting.

## The Robustness Certification Problem

The fundamental goal of the robustness certification problem is to prove that all inputs in the input uncertainty set map to safe outputs, i.e., that $f(x) \in \mathcal{S}$ for all $x \in \mathcal{X}$. If this certificate holds, the network is said to be *certifiably robust*, which of course is a property that holds with respect to a particular input uncertainty set. The robustness condition can also be written as $f(\mathcal{X}) \subseteq \mathcal{S}$, or equivalently

$$\sup_{x \in \mathcal{X}} \left(c_i^\top f(x) - d_i\right) \leq 0 \text{ for all } i \in \{1, 2, \ldots, n_\mathcal{S}\},$$

where $c_i^\top$ is the $i$th row of $C$.[2] Under this formulation, the certification procedure amounts to solving $n_\mathcal{S}$ optimization problems. The methods we develop in this chapter can be applied to each of these optimizations individually, and therefore in the sequel we focus on a single optimization problem by assuming that $n_\mathcal{S} = 1$, namely $\sup_{x \in \mathcal{X}} c^\top f(x)$. We also assume without loss of generality that $d = 0$. If $d$ were nonzero, one may absorb $d$ into the cost vector $c$ and modify the network model by appending a fixed value of 1 at the end of the output vector $f(x)$. Under these formulations, we write the robustness certification problem as

$$\phi^\star(\mathcal{X}) = \sup\{c^\top z : z = f(x),\ x \in \mathcal{X}\}, \tag{2.2}$$

and recall that we seek to certify that $\phi^\star(\mathcal{X}) \leq 0$.

Since the function $f$ is in general nonconvex, the nonlinear equality constraint $z = f(x)$ makes the optimization (2.2) a nonconvex problem and the set $f(\mathcal{X})$ a nonconvex set. Furthermore, since the intermediate activations and preactivations of the network generally have

[2] In this chapter, we write the certification problem as a maximization. This is equivalent to the minimization form of the problem upon negation of the objective function.

a large dimension in practice, the problem (2.2) is typically a high-dimensional problem. Therefore, computing an exact robustness certificate, as formulated in (2.2), is computationally intractable. Instead of directly maximizing $c^\top z$ over the nonconvex output set $f(\mathcal{X})$, one can overcome these hurdles by optimizing over a convex outer approximation $\hat{f}(\mathcal{X}) \supseteq f(\mathcal{X})$. Indeed, this new problem is a convex relaxation of the original problem, so it is generally easier and more efficient to solve. If the convex outer approximation is shown to be safe, i.e., $\hat{f}(\mathcal{X}) \subseteq \mathcal{S}$, then the true nonconvex set $f(\mathcal{X})$ is also known to be safe, implying that the robustness of the network is certified. Figure 2.1 illustrates this idea.

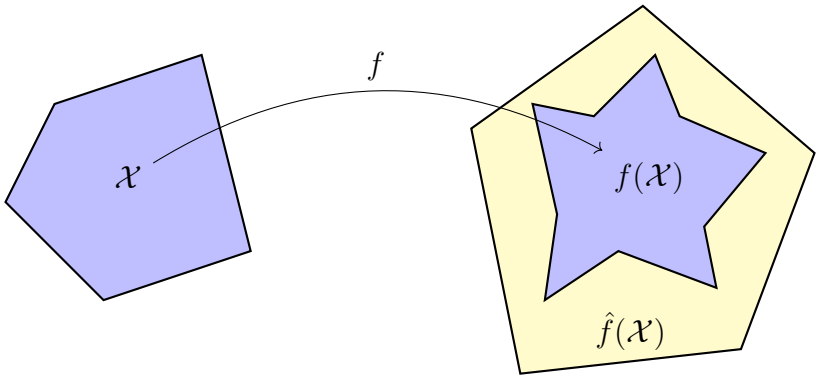



Figure 2.1: The set $\hat{f}(\mathcal{X})$ is a convex outer approximation of the nonconvex set $f(\mathcal{X})$. If the outer approximation is safe, i.e., $\hat{f}(\mathcal{X}) \subseteq \mathcal{S}$, then so is $f(\mathcal{X})$.

A fundamental drawback to the convex relaxation approach to robustness certification is as follows: if the outer approximation $\hat{f}(\mathcal{X})$ is too loose, then it may intersect with the unsafe region of the output space, meaning $\hat{f}(\mathcal{X}) \not\subseteq \mathcal{S}$, even in the case where the true output set is safe. In this scenario, the convex relaxation fails to issue a certificate of robustness, since the optimal value to the convex relaxation is positive, which incorrectly suggests the presence of unsafe network inputs within $\mathcal{X}$. This situation is illustrated in Figure 2.2.

The fundamental goal of this chapter is to develop convex relaxation methods for robustness certification such that the outer approximation tightly fits $f(\mathcal{X})$, in effect granting strong certificates while maintaining the computational and theoretical advantages of convex optimization. We focus on two popular types of convex relaxations, namely, LP [150] and SDP [116] relaxations. It has been shown that the SDP relaxation for ReLU networks is tighter than the LP relaxation, with the tradeoff of being computationally more demanding. Our general approach for both the LP and the SDP relaxations is based on partitioning the input uncertainty set and solving a convex relaxation on each input part separately. Throughout our theoretical development and numerical simulations, we will show that this approach presents a valid, efficient, and effective way to tighten the relaxations of both LP and SDP certifications, and we derive worst-case optimal branching strategies for both relaxation methods. We now turn to mathematically formulating these relaxations.

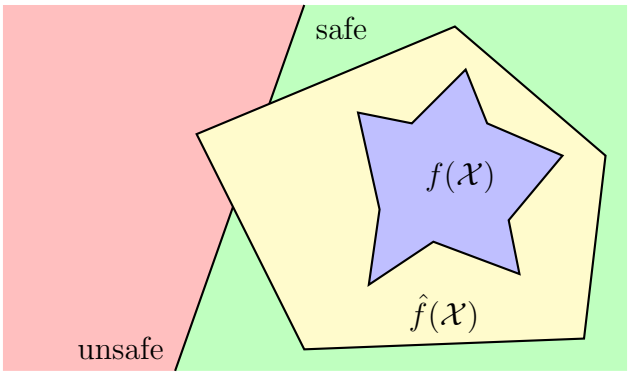



Figure 2.2: This scenario shows that if the convex outer approximation $\hat{f}(\mathcal{X})$ is too large, meaning the relaxation is too loose, then the convex approach fails to issue a certificate of robustness.

## LP Relaxation of the Network Constraints

We now introduce the LP relaxation. First, we remark that since $\mathcal{X}$ is bounded, the preactivations at each layer are bounded as well. That is, for every $k \in \{1, 2, \ldots, K\}$, there exist preactivation bounds $l^{[k]}, u^{[k]} \in \mathbb{R}^{n_k}$ such that $l^{[k]} \leq \hat{z}^{[k]} \leq u^{[k]}$, where $\hat{z}^{[k]}$ is the $k$th layer's preactivation in (2.1), for all $x \in \mathcal{X}$. We assume without loss of generality that $l^{[k]} \leq 0 \leq u^{[k]}$ for all $k$, since, if $l_i^{[k]} > 0$ for some $i$, the preactivation bound $l_i^{[k]} \leq \hat{z}_i^{[k]}$ can be replaced by $0 \leq \hat{z}_i^{[k]}$, and similarly for the case where $u_i^{[k]} < 0$. Although there exist various methods in the literature for efficiently approximating these preactivation bounds, we consider the bounds to be tight, i.e., $\hat{z}^{[k]} = l^{[k]}$ for some $x \in \mathcal{X}$, and similarly for the upper bound $u^{[k]}$. Tjeng, Xiao, and Tedrake [135] provides efficient methods for computing these bounds. Now, following the approach initially introduced in Wong and Kolter [150], we can relax the $k$th ReLU constraint in (2.1) to its convex upper envelope between the preactivation bounds. This is graphically shown in Figure 2.3.

We call the convex upper envelope associated with layer $k$ the *relaxed ReLU constraint set*, and its mathematical definition is given by three linear inequalities:

$$\begin{aligned}\mathcal{N}_{\text{LP}}^{[k]} = \{(x^{[k-1]}, x^{[k]}) \in \mathbb{R}^{n_{k-1}} \times \mathbb{R}^{n_k} : x^{[k]} \leq u^{[k]} \odot (\hat{z}^{[k]} - l^{[k]}) \oslash (u^{[k]} - l^{[k]}),\\ x^{[k]} \geq 0,\ x^{[k]} \geq \hat{z}^{[k]},\ \hat{z}^{[k]} = W^{[k-1]} x^{[k-1]}\}.\end{aligned} \tag{2.3}$$

Next, we define the *relaxed network constraint set* to be

$$\mathcal{N}_{\text{LP}} = \{(x, z) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} : (x, x^{[1]}) \in \mathcal{N}_{\text{LP}}^{[1]},\ (x^{[1]}, x^{[2]}) \in \mathcal{N}_{\text{LP}}^{[2]}, \ldots, (x^{[K-1]}, z) \in \mathcal{N}_{\text{LP}}^{[K]}\}. \tag{2.4}$$

In essence, $\mathcal{N}_{\text{LP}}$ is the set of all input-output pairs of the network that satisfy the relaxed ReLU constraints at every layer. Note that, since the bounds $l^{[k]}$ and $u^{[k]}$ are determined by the input uncertainty set $\mathcal{X}$, the set $\mathcal{N}_{\text{LP}}^{[k]}$ is also determined by $\mathcal{X}$ for all layers $k$.

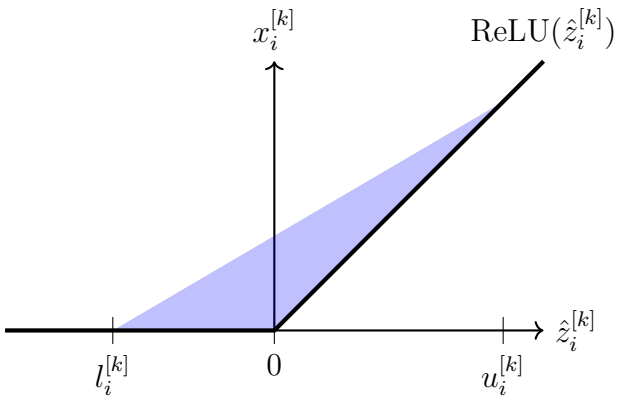



Figure 2.3: Relaxed ReLU constraint set $\mathcal{N}_{\mathrm{LP}}^{[k]}$ at a single neuron $i$ in layer $k$ of the network.

*Remark* 1. For networks with one hidden layer (i.e., $K = 1$), the single relaxed ReLU constraint set coincides with the relaxed network constraint set: $\mathcal{N}_{\mathrm{LP}}^{[1]} = \mathcal{N}_{\mathrm{LP}}$. Therefore, for $K = 1$ we drop the $k$-notation and simply write $z$, $\hat{z}$, $x$, $W$, $l$, $u$, and $\mathcal{N}_{\mathrm{LP}}$.

Finally, we introduce the LP relaxation of (2.2):

$$\hat{\phi}_{\mathrm{LP}}^{\star}(\mathcal{X}) = \sup\{c^{\top} z : (x, z) \in \mathcal{N}_{\mathrm{LP}},\ x \in \mathcal{X}\}. \tag{2.5}$$

Notice that, if $x \in \mathcal{X}$ and $z = f(x)$, then $(x, z) \in \mathcal{N}_{\mathrm{LP}}$ by the definition of $\mathcal{N}_{\mathrm{LP}}$. Furthermore, since $\mathcal{X}$ is a bounded convex polytope and $\mathcal{N}_{\mathrm{LP}}$ is defined by a system of linear constraints, we confirm that (2.5) is a linear program. Therefore, (2.5) is indeed an LP relaxation of (2.2), so it holds that

$$\phi^{\star}(\mathcal{X}) \leq \hat{\phi}_{\mathrm{LP}}^{\star}(\mathcal{X}). \tag{2.6}$$

This analytically shows what Figure 2.1 and Figure 2.2 illustrate: the condition that

$$\hat{\phi}_{\mathrm{LP}}^{\star}(\mathcal{X}) \leq 0$$

is sufficient to conclude that the network is certifiably robust, but if $\hat{\phi}_{\mathrm{LP}}^{\star}(\mathcal{X}) > 0$, the relaxation fails to certify whether or not the network is robust, since it may still hold that $\phi^{\star}(\mathcal{X}) \leq 0$.

## SDP Relaxation of the Network Constraints

An alternative convex relaxation of the robustness certification problem can be formulated as an SDP. This method was first introduced in Raghunathan, Steinhardt, and Liang [116]. Here, we will introduce the SDP relaxation for a network with a single hidden layer for notational convenience. The extension to multi-layer networks is straightforward and presented

in Raghunathan, Steinhardt, and Liang [116]. In this formulation, the optimization variable $(x, z) \in \mathbb{R}^{n_x+n_z}$ is lifted to a symmetric matrix

$$P = \begin{bmatrix} 1 \\ x \\ z \end{bmatrix} \begin{bmatrix} 1 & x^\top & z^\top \end{bmatrix} \in \mathbb{S}^{n_x+n_z+1}.$$

We use the following block-indexing for $P$:

$$P = \begin{bmatrix} P_1 & P_x^\top & P_z^\top \\ P_x & P_{xx} & P_{xz} \\ P_z & P_{zx} & P_{zz} \end{bmatrix},$$

where $P_1 \in \mathbb{R}$, $P_x \in \mathbb{R}^{n_x}$, $P_z \in \mathbb{R}^{n_z}$, $P_{xx} \in \mathbb{S}^{n_x}$, $P_{zz} \in \mathbb{S}^{n_z}$, $P_{xz} \in \mathbb{R}^{n_x \times n_z}$, and $P_{zx} = P_{xz}^\top$. This lifting procedure results in the optimization problem

$$\begin{aligned} \underset{P \in \mathbb{S}^{n_x+n_z+1}}{\text{maximize}} \quad & c^\top P_z \\ \text{subject to} \quad & P_z \geq 0, \\ & P_z \geq W P_x, \\ & \operatorname{diag}(P_{xx}) \leq (l+u) \odot P_x - l \odot u, \\ & \operatorname{diag}(P_{zz}) = \operatorname{diag}(W P_{xz}), \\ & P_1 = 1, \\ & P \succeq 0, \\ & \operatorname{rank}(P) = 1. \end{aligned}$$

Here, there are no preactivation bounds, unlike the LP relaxation. The vectors $l, u \in \mathbb{R}^{n_x}$ are lower and upper bounds on the input, which are determined by the input uncertainty set. For example, if $\mathcal{X} = \{x \in \mathbb{R}^{n_x} : \|x - \bar{x}\|_\infty \leq \epsilon\}$, then $l = \bar{x} - \epsilon \mathbf{1}_{n_x}$ and $u = \bar{x} + \epsilon \mathbf{1}_{n_x}$.

We remark that the above problem is equivalent to the original robustness certification problem; no relaxation has been made yet. The only nonconvex constraint in this formulation is the rank-1 constraint on $P$. Dropping this rank constraint, we obtain the SDP relaxed network constraint set:

$$\begin{aligned} \mathcal{N}_{\text{SDP}} = \{P \in \mathbb{S}^{n_x+n_z+1} : P_z \geq 0,\ P_z \geq W P_x,\ \operatorname{diag}(P_{zz}) = \operatorname{diag}(W P_{xz}), \\ \operatorname{diag}(P_{xx}) \leq (l+u) \odot P_x - l \odot u,\ P_1 = 1,\ P \succeq 0\}. \end{aligned} \tag{2.7}$$

Using this relaxed network constraint set as the feasible set for the optimization, we arrive at the SDP relaxation

$$\hat{\phi}^\star_{\text{SDP}}(\mathcal{X}) = \sup\{c^\top P_z : P \in \mathcal{N}_{\text{SDP}},\ P_x \in \mathcal{X}\}. \tag{2.8}$$

It is clear that by dropping the rank constraint, we have enlarged the feasible set, so again we obtain a viable relaxation of the original problem (2.2): $\phi^\star(\mathcal{X}) \leq \hat{\phi}^\star_{\text{SDP}}(\mathcal{X})$. In the

case the solution $P^\star$ to (2.8) is rank-1, we can factorize it as

$$P^\star = \begin{bmatrix} 1 \\ x^\star \\ z^\star \end{bmatrix} \begin{bmatrix} 1 & x^{\star\top} & z^{\star\top} \end{bmatrix}$$

and conclude that $(x^\star, z^\star)$ solves the original nonconvex problem. However, it is generally the case that the SDP solution will be of higher rank, leading to relaxation error and the possibility of a void robustness certificate, similar to the LP relaxation. We now turn to building upon the LP and SDP convex relaxations via input partitioning in order to tighten their relaxations.

## 2.3 Partitioned LP Relaxation

### Properties of Partitioned Relaxation

In this section, we investigate the properties and effectiveness of partitioning the input uncertainty set when solving the LP relaxation for robustness certification. We start by validating the approach, namely, by showing that solving the LP relaxation separately on each input part maintains a theoretically guaranteed upper bound on the optimal value of the unrelaxed problem (2.2). Afterwards, the approach is proven to yield a tighter upper bound than solving the LP relaxation without partitioning.

#### Partitioning Gives Valid Relaxation

**Proposition 1.** *Let $\{\mathcal{X}^{(j)} \subseteq \mathcal{X} : j \in \{1, 2, \ldots, p\}\}$ be a partition of $\mathcal{X}$. Then, it holds that*

$$\phi^\star(\mathcal{X}) \leq \max_{j \in \{1,2,\ldots,p\}} \hat{\phi}^\star_{\text{LP}}(\mathcal{X}^{(j)}). \tag{2.9}$$

*Proof.* Assume that $\phi^\star(\mathcal{X}) > \max_{j\in\{1,2,\ldots,p\}} \hat{\phi}^\star_{\text{LP}}(\mathcal{X}^{(j)})$. Then,

$$\phi^\star(\mathcal{X}) > \hat{\phi}^\star_{\text{LP}}(\mathcal{X}^{(j)}) \text{ for all } j \in \{1, 2, \ldots, p\}. \tag{2.10}$$

Let $(x^\star, z^\star)$ denote an optimal solution to the unrelaxed problem (2.2), i.e., $x^\star \in \mathcal{X}$, $z^\star = f(x^\star)$, and

$$c^\top z^\star = \phi^\star(\mathcal{X}). \tag{2.11}$$

Since $\cup_{j=1}^p \mathcal{X}^{(j)} = \mathcal{X}$, there exists $j^\star \in \{1, 2, \ldots, p\}$ such that $x^\star \in \mathcal{X}^{(j^\star)}$. Since $x^\star \in \mathcal{X}^{(j^\star)}$ and $z^\star = f(x^\star)$, it holds that $(x^\star, z^\star) \in \mathcal{N}^{(j^\star)}_{\text{LP}}$, where $\mathcal{N}^{(j^\star)}_{\text{LP}}$ is the relaxed network constraint set defined by $\mathcal{X}^{(j^\star)}$. Therefore,

$$c^\top z^\star \leq \sup\{c^\top z : x \in \mathcal{X}^{(j^\star)},\ (x, z) \in \mathcal{N}^{(j^\star)}_{\text{LP}}\} = \hat{\phi}^\star_{\text{LP}}(\mathcal{X}^{(j^\star)}) < \phi^\star(\mathcal{X}),$$

where the first inequality comes from the feasibility of $(x^\star, z^\star)$ over the $j^\star$th subproblem and the final inequality is due to (2.10). This contradicts the optimality of $(x^\star, z^\star)$ given in (2.11). Hence, (2.9) must hold. □

Despite the fact that Proposition 1 asserts an intuitively expected bound, we remark the importance for its formal statement and proof. In particular, the inequality (2.9) serves as the fundamental reason for why the partitioned LP relaxation can be used to certify that all inputs in $\mathcal{X}$ map to safe outputs in the safe set $\mathcal{S}$. Knowing that the partitioning approach is valid for robustness certification, we move on to studying the effectiveness of partitioning.

### Tightening of the Relaxation

We now show that the bound (2.6) can always be tightened by partitioning the input uncertainty set. The result is given for networks with one hidden layer for simplicity. However, the conclusion naturally generalizes to multi-layer ReLU networks.

**Proposition 2.** *Consider a feedforward ReLU neural network with one hidden layer. Let $\{\mathcal{X}^{(j)} \subseteq \mathcal{X} : j \in \{1, 2, \dots, p\}\}$ be a partition of $\mathcal{X}$. For the $j$th input part $\mathcal{X}^{(j)}$, denote the corresponding preactivation bounds by $l^{(j)}$ and $u^{(j)}$, where $l \leq l^{(j)} \leq Wx \leq u^{(j)} \leq u$ for all $x \in \mathcal{X}^{(j)}$. Then, it holds that*

$$\max_{j \in \{1,2,\dots,p\}} \hat{\phi}^{\star}_{\mathrm{LP}}(\mathcal{X}^{(j)}) \leq \hat{\phi}^{\star}_{\mathrm{LP}}(\mathcal{X}). \tag{2.12}$$

*Proof.* Let $j \in \{1, 2, \dots, p\}$. It will be shown that $\mathcal{N}^{(j)}_{\mathrm{LP}} \subseteq \mathcal{N}_{\mathrm{LP}}$. Let $(x, z) \in \mathcal{N}^{(j)}_{\mathrm{LP}}$. Define $u' = u^{(j)}$, $l' = l^{(j)}$, and

$$\begin{aligned} g(x) &= u \odot (Wx - l) \oslash (u - l), \\ g'(x) &= u' \odot (Wx - l') \oslash (u' - l'). \end{aligned}$$

Then, by letting $\Delta g(x) = g(x) - g'(x) = a \odot (Wx) + b$, where

$$\begin{aligned} a &= u \oslash (u - l) - u' \oslash (u' - l'), \\ b &= u' \odot l' \oslash (u' - l') - u \odot l \oslash (u - l), \end{aligned}$$

the following relations are derived for all $i \in \{1, 2, \dots, n_z\}$:

$$\begin{aligned} g^{\star}_i &:= \inf_{\{x : l' \leq Wx \leq u'\}} (\Delta g(x))_i \\ &\geq \inf_{\{\hat{z} : l' \leq \hat{z} \leq u'\}} (a \odot \hat{z} + b)_i \\ &= \inf_{\{\hat{z}_i : l'_i \leq \hat{z}_i \leq u'_i\}} (a_i \hat{z}_i + b_i) \\ &= \begin{cases} a_i l'_i + b_i & \text{if } a_i \geq 0, \\ a_i u'_i + b_i & \text{if } a_i < 0. \end{cases} \end{aligned}$$

In the case that $a_i \geq 0$, we have that

$$g^{\star}_i \geq a_i l'_i + b_i = \left( \frac{u_i}{u_i - l_i} - \frac{u'_i}{u'_i - l'_i} \right) l'_i + \left( \frac{u'_i l'_i}{u'_i - l'_i} - \frac{u_i l_i}{u_i - l_i} \right) = \frac{u_i}{u_i - l_i} (l'_i - l_i) \geq 0,$$

where the final inequality comes from the fact that $u \geq 0$, $l' \geq l$, and $u > l$. On the other hand, if $a_i < 0$, it holds that

$$\begin{aligned} g_i^\star &\geq a_i u_i' + b_i \\ &= \left(\frac{u_i}{u_i - l_i} - \frac{u_i'}{u_i' - l_i'}\right) u_i' + \left(\frac{u_i' l_i'}{u_i' - l_i'} - \frac{u_i l_i}{u_i - l_i}\right) \\ &= \frac{u_i}{u_i - l_i}(u_i' - l_i) - u_i' \\ &= \frac{u_i' - u_i}{u_i - l_i} l_i \\ &\geq 0, \end{aligned}$$

where the final inequality comes from the fact that $u' \leq u$, $l \leq 0$, and $u > l$. Therefore,

$$g^\star = (g_1^\star, g_2^\star, \ldots, g_{n_z}^\star) \geq 0,$$

which implies that $\Delta g(x) = g(x) - g'(x) \geq 0$ for all $x$ such that $l^{(j)} = l' \leq Wx \leq u' = u^{(j)}$. Hence, since $(x, z) \in \mathcal{N}_{\text{LP}}^{(j)}$, it holds that $z \geq 0$, $z \geq Wx$, and

$$z \leq g'(x) \leq g(x) = u \odot (Wx - l) \oslash (u - l).$$

Therefore, we have that $(x, z) \in \mathcal{N}_{\text{LP}}$.

Since $\mathcal{X}^{(j)} \subseteq \mathcal{X}$ (by definition) and $\mathcal{N}_{\text{LP}}^{(j)} \subseteq \mathcal{N}_{\text{LP}}$, it holds that the solution to the problem over the smaller feasible set gives a lower bound to the original solution: $\hat{\phi}_{\text{LP}}^\star(\mathcal{X}^{(j)}) \leq \hat{\phi}_{\text{LP}}^\star(\mathcal{X})$. Finally, since $j$ was chosen arbitrarily, this implies the desired inequality (2.12). □

Combining Proposition 1 and Proposition 2 shows that

$$\phi^\star(\mathcal{X}) \leq \max_{j \in \{1,2,\ldots,p\}} \hat{\phi}_{\text{LP}}^\star(\mathcal{X}^{(j)}) \leq \hat{\phi}_{\text{LP}}^\star(\mathcal{X}),$$

i.e., that the partitioned LP relaxation is theoretically guaranteed to perform at least as well as the unpartitioned LP when solving the robustness certification problem. The improvement in the partitioned LP relaxation is captured by the difference

$$\hat{\phi}_{\text{LP}}^\star(\mathcal{X}) - \max_{j \in \{1,2,\ldots,p\}} \hat{\phi}_{\text{LP}}^\star(\mathcal{X}^{(j)}),$$

which is always nonnegative. We remark that it is possible for the improvement to be null in the sense that $\max_{j \in \{1,2,\ldots,p\}} \hat{\phi}_{\text{LP}}^\star(\mathcal{X}^{(j)}) = \hat{\phi}_{\text{LP}}^\star(\mathcal{X})$. This may occur when the partition used is poorly chosen. An example of such a poor choice may be if one were to partition along a direction in the input space that, informally speaking, corresponds to directions near-orthogonal to the cost vector $c$ in the output space. In this case, one would expect all improvements to be nullified, and for the partitioned relaxation to give the same optimal value as the unpartitioned relaxation. Consequently, the following important question arises: *what constitutes a good partition so that the improvement* $\hat{\phi}_{\text{LP}}^\star(\mathcal{X}) - \max_{j \in \{1,2,\ldots,p\}} \hat{\phi}_{\text{LP}}^\star(\mathcal{X}^{(j)})$ *is strictly greater than zero and maximal?* We address this question below, and again in Section 2.4.

## Motivating Partition

In this section, we begin to answer our earlier inquiry, namely, how to choose a partition in order to maximize the improvement $\hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X}) - \max_{j\in\{1,2,\dots,p\}} \hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X}^{(j)})$ in the partitioned LP relaxation. Recall that this is equivalent to minimizing the relaxation error relative to the original unrelaxed problem, since $\phi^\star(\mathcal{X}) \leq \max_{j\in\{1,2,\dots,p\}} \hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X}^{(j)}) \leq \hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X})$. To this end, we construct a partition with finitely many parts, based on the parameters of the network, which is shown to exactly recover the optimal value of the original unrelaxed problem (2.2). For simplicity, we present the result for a single hidden layer, but the basic idea of partitioning at the "kinks" of the ReLUs in order to collapse the ReLU upper envelope onto the ReLU curve and eliminate relaxation error can be generalized to multi-layer settings. At this point, let us remark that in Proposition 3 below, we use a slight difference in notation for the partition. Namely, we use the set of all $n_z$-vectors with binary elements, $\mathcal{J} \coloneqq \{0,1\}^{n_z} = \{0,1\} \times \{0,1\} \times \cdots \times \{0,1\}$, to index the parts of the partition. Under this setting, the partition is composed of $p = 2^{n_z}$ parts, so that $\mathcal{X}^{(j)}$ is the part of the partition corresponding to the binary vector $j$, which is an element of the index set $\mathcal{J}$. This temporary change in notation is chosen to simplify the proof of Proposition 3.

**Proposition 3.** *Consider a feedforward ReLU neural network with one hidden layer and denote the ith row of $W$ by $w_i^\top \in \mathbb{R}^{1\times n_x}$ for all $i \in \{1,2,\dots,n_z\}$. Define $\mathcal{J} = \{0,1\}^{n_z}$ and take the partition of $\mathcal{X}$ to be indexed by $\mathcal{J}$, meaning that $\{\mathcal{X}^{(j)} \subseteq \mathcal{X} : j \in \mathcal{J}\}$, where for a given $j \in \mathcal{J}$ we define*

$$\mathcal{X}^{(j)} = \{x \in \mathcal{X} : w_i^\top x \geq 0 \text{ for all } i \text{ such that } j_i = 1,\ w_i^\top x < 0 \text{ for all } i \text{ such that } j_i = 0\}. \tag{2.13}$$

*Then, the partitioned relaxation is exact, i.e.,*

$$\phi^\star(\mathcal{X}) = \max_{j\in\mathcal{J}} \hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X}^{(j)}). \tag{2.14}$$

*Proof.* We first show that $\{\mathcal{X}^{(j)} \subseteq \mathcal{X} : j \in \mathcal{J}\}$ is a valid partition. Since $\mathcal{X}^{(j)} \subseteq \mathcal{X}$ for all $j \in \mathcal{J}$, the relation $\cup_{j\in\mathcal{J}} \mathcal{X}^{(j)} \subseteq \mathcal{X}$ is satisfied. Now, suppose that $x \in \mathcal{X}$. Then, for all $i \in \{1,2,\dots,n_z\}$, either $w_i^\top x \geq 0$ or $w_i^\top x < 0$ holds. Define $j \in \{0,1\}^{n_z}$ as follows:

$$j_i = \begin{cases} 1 & \text{if } w_i^\top x \geq 0, \\ 0 & \text{if } w_i^\top x < 0, \end{cases}$$

for all $i \in \{1,2,\dots,n_z\}$. Then, by the definition of $\mathcal{X}^{(j)}$ in (2.13), it holds that $x \in \mathcal{X}^{(j)}$. Therefore, the relation $x \in \mathcal{X}$ implies that $x \in \mathcal{X}^{(j)}$ for some $j \in \{0,1\}^{n_z} = \mathcal{J}$. Hence, $\mathcal{X} \subseteq \cup_{j\in\mathcal{J}} \mathcal{X}^{(j)}$, and therefore $\cup_{j\in\mathcal{J}} \mathcal{X}^{(j)} = \mathcal{X}$.

We now show that $\mathcal{X}^{(j)} \cap \mathcal{X}^{(k)} = \emptyset$ for all $j \neq k$. Let $j,k \in \mathcal{J}$ with the property that $j \neq k$. Then, there exists $i \in \{1,2,\dots,n_z\}$ such that $j_i \neq k_i$. Let $x \in \mathcal{X}^{(j)}$. In the case that $w_i^\top x \geq 0$, it holds that $j_i = 1$ and therefore $k_i = 0$. Hence, for all $y \in \mathcal{X}^{(k)}$, it holds that $w_i^\top y < 0$, and therefore $x \notin \mathcal{X}^{(k)}$. An analogous reasoning shows that $x \notin \mathcal{X}^{(k)}$ when

$w_i^\top x < 0$. Therefore, one concludes that $x \in \mathcal{X}^{(j)}$ and $j \neq k$ implies that $x \notin \mathcal{X}^{(k)}$, i.e., that $\mathcal{X}^{(j)} \cap \mathcal{X}^{(k)} = \emptyset$. Hence, $\{\mathcal{X}^{(j)} \subseteq \mathcal{X} : j \in \mathcal{J}\}$ is a valid partition.

We now prove (2.14). Let $j \in \mathcal{J}$. Since $w_i^\top x \geq 0$ for all $i$ such that $j_i = 1$, the preactivation lower bound becomes $l_i^{(j)} = 0$ for all such $i$. On the other hand, since $w_i^\top x < 0$ for all $i$ such that $j_i = 0$, the preactivation upper bound becomes $u_i^{(j)} = 0$ for all such $i$. Therefore, the relaxed network constraint set (2.4) for the $j$th input part reduces to

$$\begin{aligned}\mathcal{N}_{\text{LP}}^{(j)} = \{(x,z) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} : z_i = 0 \text{ for all } i \text{ such that } j_i = 0,\\ z_i = w_i^\top x = (Wx)_i \text{ for all } i \text{ such that } j_i = 1\}.\end{aligned}$$

That is, the relaxed ReLU constraint envelope collapses to the exact ReLU constraint through the prior knowledge of each preactivation coordinate's sign.

Therefore, we find that for all $x \in \mathcal{X}^{(j)}$ it holds that $(x,z) \in \mathcal{N}_{\text{LP}}^{(j)}$ if and only if $z = \text{ReLU}(Wx)$. Hence, the LP over the $j$th input part yields that

$$\begin{aligned}\hat{\phi}_{\text{LP}}^\star(\mathcal{X}^{(j)}) &= \sup\{c^\top z : (x,z) \in \mathcal{N}_{\text{LP}}^{(j)},\ x \in \mathcal{X}^{(j)}\}\\ &= \sup\{c^\top z : z = \text{ReLU}(Wx),\ x \in \mathcal{X}^{(j)}\}\\ &\leq \sup\{c^\top z : z = \text{ReLU}(Wx),\ x \in \mathcal{X}\}\\ &= \phi^\star(\mathcal{X}).\end{aligned}$$

Since $j$ was chosen arbitrarily, it holds that

$$\max_{j \in \mathcal{J}} \hat{\phi}_{\text{LP}}^\star(\mathcal{X}^{(j)}) \leq \phi^\star(\mathcal{X}).$$

Since $\phi^\star(\mathcal{X}) \leq \max_{j \in \mathcal{J}} \hat{\phi}_{\text{LP}}^\star(\mathcal{X}^{(j)})$ by the relaxation bound (2.9), the equality (2.14) holds, as desired. □

Although the partition proposed in Proposition 3 completely eliminates relaxation error of the LP, using it in practice may be computationally intractable, as it requires solving $2^{n_z}$ separate linear programs. Despite this limitation, the result provides two major theoretical implications. First, our input partitioning approach is fundamentally shown to be a simple, yet very powerful method, as the robustness certification problem can be solved exactly via a finite number of linear program subproblems. Second, the partition proposed in Proposition 3 shows us the structure of an optimal partition, namely that the parts of the partition are defined by the half-spaces generated by the rows of $W$ (see Figure 2.4). This result paves the way to develop a tractable branching scheme that incorporates the reduction in relaxation error endowed by the structure of this motivating partition. In the next section, we explore this idea further, and seek to answer the following question: *if we only partition along a single row of the weight matrix, which one is the best to choose?*

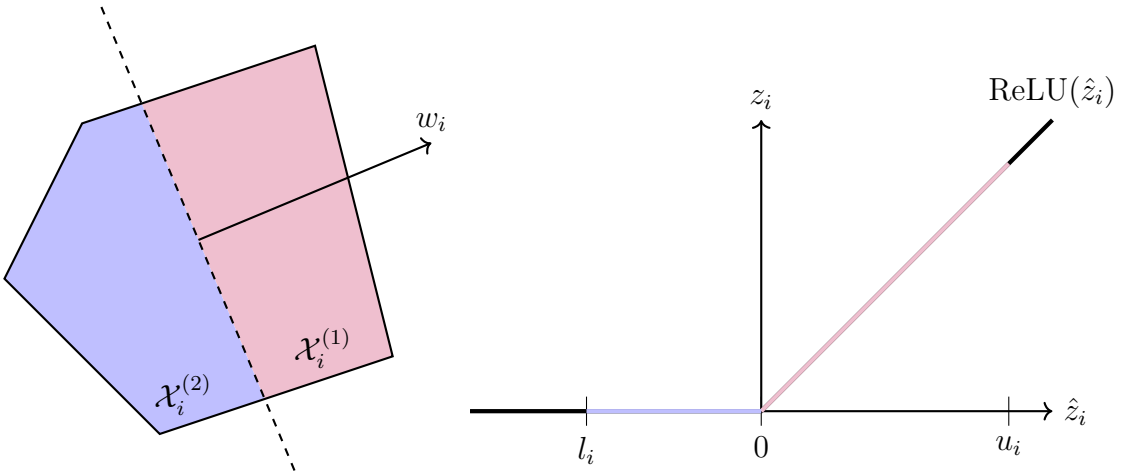



Figure 2.4: Partitioning based on row $w_i^\top$ of the weight matrix. This partition results in an exact ReLU constraint in coordinate $i$ over the two resulting input parts $\mathcal{X}_i^{(1)} = \{x \in \mathcal{X} : w_i^\top x \geq 0\}$ and $\mathcal{X}_i^{(2)} = \mathcal{X} \setminus \mathcal{X}_i^{(1)}$.

## LP Branching Scheme

In this section, we propose an explicit, computationally tractable, and effective LP branching scheme. The branching scheme is developed based on analyses for a single hidden layer. However, the resulting branching scheme is still applicable to multi-layer networks, and indeed will be shown to remain effective on two-layer networks in the numerical simulations of Section 2.6. The development of the partition boils down to two main ideas. First, we restrict our attention to two-part partitions defined by rows of the weight matrix $W$, specifically, $\mathcal{X}_i^{(1)} = \{x \in \mathcal{X} : w_i^\top x \geq 0\}$ and $\mathcal{X}_i^{(2)} = \mathcal{X} \setminus \mathcal{X}_i^{(1)}$, as motivated in the previous section. Second, we seek which index $i \in \{1, 2, \ldots, n_z\}$ gives the best partition, in the sense that the relaxation error of the resulting partitioned LP is minimized. As will be shown below, this second aspect is NP-hard to discern in general. Therefore, to find the optimal row to partition along, we instead seek to minimize the worst-case relaxation error.

### Worst-Case Relaxation Bound

We begin by bounding the relaxation error below.

**Theorem 1.** *Consider a feedforward ReLU neural network with one hidden layer, with the input uncertainty set $\mathcal{X}$ and preactivation bounds $l, u \in \mathbb{R}^{n_z}$. Consider also the relaxation error $\Delta\phi^\star(\mathcal{X}) \coloneqq \hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X}) - \phi^\star(\mathcal{X})$. Let $(\tilde{x}^\star, \tilde{z}^\star)$ and $(x^\star, z^\star)$ be optimal solutions for the relaxation $\hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X})$ and the unrelaxed problem $\phi^\star(\mathcal{X})$, respectively. Given an arbitrary norm*

*$\|\cdot\|$ on $\mathbb{R}^{n_x}$, it holds that*

$$\begin{aligned}\Delta\phi^\star(\mathcal{X}) \le \sum_{i=1}^{n_z} \Bigg( &\mathrm{ReLU}(c_i)\frac{u_i}{u_i - l_i}(\min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} - l_i) \\ &+ \mathrm{ReLU}(-c_i)\min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\}\Bigg),\end{aligned} \tag{2.15}$$

*where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$.*

*Proof.* First, recall that $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ is assumed to be compact, and is therefore bounded, and hence $d_{\|\cdot\|}(\mathcal{X}) < \infty$. The definitions of $(\tilde{x}^\star, \tilde{z}^\star)$ and $(x^\star, z^\star)$ give that

$$\Delta\phi^\star(\mathcal{X}) = \sum_{i=1}^{n_z} c_i(\tilde{z}_i^\star - z_i^\star) \le \sum_{i=1}^{n_z} \Delta\phi_i^\star, \tag{2.16}$$

where

$$\begin{aligned}\Delta\phi_i^\star = \sup\Bigg\{ &c_i(\tilde{z}_i - z_i) : z_i = \mathrm{ReLU}(w_i^\top x),\ \tilde{z}_i \ge 0,\ \tilde{z}_i \ge w_i^\top \tilde{x}, \\ &\tilde{z}_i \le \frac{u_i}{u_i - l_i}(w_i^\top \tilde{x} - l_i),\ x, \tilde{x} \in \mathcal{X}\Bigg\}\end{aligned}$$

for all $i \in \{1, 2, \ldots, n_z\}$. Note that

$$\begin{aligned}\Delta\phi_i^\star = \sup\Bigg\{ &c_i(\tilde{z}_i - z_i) : z_i = \mathrm{ReLU}(\hat{z}_i),\ \tilde{z}_i \ge 0,\ \tilde{z}_i \ge \hat{\tilde{z}}_i,\ \tilde{z}_i \le \frac{u_i}{u_i - l_i}(\hat{\tilde{z}}_i - l_i), \\ &\hat{z} = Wx,\ \hat{\tilde{z}} = W\tilde{x},\ x, \tilde{x} \in \mathcal{X}\Bigg\}.\end{aligned}$$

If $x, \tilde{x} \in \mathcal{X}$ and $\hat{z}, \hat{\tilde{z}}$ satisfy $\hat{z} = Wx$, $\hat{\tilde{z}} = W\tilde{x}$, then they satisfy $l \le \hat{z}, \hat{\tilde{z}} \le u$ and $|\hat{\tilde{z}}_k - \hat{z}_k| = |w_k^\top(\tilde{x} - x)| \le \|w_k\|_* \|\tilde{x} - x\| \le \|w_k\|_* d_{\|\cdot\|}(\mathcal{X})$ for all $k \in \{1, 2, \ldots, n_z\}$ by the Cauchy-Schwarz inequality for dual norms. Therefore,

$$\begin{aligned}\Delta\phi_i^\star \le\ &\sup\Bigg\{ c_i(\tilde{z}_i - z_i) : z_i = \mathrm{ReLU}(\hat{z}_i),\ \tilde{z}_i \ge 0,\ \tilde{z}_i \ge \hat{\tilde{z}}_i,\ \tilde{z}_i \le \frac{u_i}{u_i - l_i}(\hat{\tilde{z}}_i - l_i), \\ &\quad l \le \hat{z}, \hat{\tilde{z}} \le u,\ |\hat{\tilde{z}}_k - \hat{z}_k| \le \|w_k\|_* d_{\|\cdot\|}(\mathcal{X}) \text{ for all } k \in \{1, 2, \ldots, n_z\},\ \hat{z}, \hat{\tilde{z}} \in \mathbb{R}^{n_z}\Bigg\} \\ =\ &\sup\Bigg\{ c_i(\tilde{z}_i - z_i) : z_i = \mathrm{ReLU}(\hat{z}_i),\ \tilde{z}_i \ge 0,\ \tilde{z}_i \ge \hat{\tilde{z}}_i,\ \tilde{z}_i \le \frac{u_i}{u_i - l_i}(\hat{\tilde{z}}_i - l_i), \\ &\quad l_i \le \hat{z}_i, \hat{\tilde{z}}_i \le u_i,\ |\hat{\tilde{z}}_i - \hat{z}_i| \le \|w_i\|_* d_{\|\cdot\|}(\mathcal{X}),\ \hat{z}_i, \hat{\tilde{z}}_i \in \mathbb{R}\Bigg\}.\end{aligned}$$

For $c_i \geq 0$, the above inequality yields that

$$\Delta\phi_i^\star \leq c_i \sup\Bigg\{\tilde{z}_i - z_i : z_i = \mathrm{ReLU}(\hat{z}_i),\ \tilde{z}_i \geq 0,\ \tilde{z}_i \geq \hat{\tilde{z}}_i,\ \tilde{z}_i \leq \frac{u_i}{u_i - l_i}(\hat{\tilde{z}}_i - l_i),$$
$$l_i \leq \hat{z}_i, \hat{\tilde{z}}_i \leq u_i,\ |\hat{\tilde{z}}_i - \hat{z}_i| \leq \|w_i\|_* d_{\|\cdot\|}(\mathcal{X}),\ \hat{z}_i, \hat{\tilde{z}}_i \in \mathbb{R}\Bigg\}.$$

The optimal solution to the above supremum is readily found by comparing the line $\tilde{z}_i = \frac{u_i}{u_i - l_i}(\hat{\tilde{z}}_i - l_i)$ to the function $z_i = \mathrm{ReLU}(\hat{z}_i)$ over $\hat{\tilde{z}}_i, \hat{z}_i \in [l_i, u_i]$. In particular, the maximum distance between $\tilde{z}_i$ and $z_i$ on the above feasible set occurs when $z_i = \hat{z}_i = 0$, $\hat{\tilde{z}}_i = \|w_i\|_* d_{\|\cdot\|}(\mathcal{X})$, and $\tilde{z}_i = \frac{u_i}{u_i - l_i}(\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}) - l_i)$. Therefore, we find that

$$\Delta\phi_i^\star \leq c_i \frac{u_i}{u_i - l_i}(\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}) - l_i), \tag{2.17}$$

for all $i \in \{1, 2, \ldots, n_z\}$ such that $c_i \geq 0$. We also note the trivial bound that $\tilde{z}_i - z_i \leq u_i$ on the feasible set of the above supremum, so that

$$\Delta\phi_i^\star \leq c_i u_i = c_i \frac{u_i}{u_i - l_i}(u_i - l_i). \tag{2.18}$$

The inequalities (2.17) and (2.18) together imply that

$$\Delta\phi_i^\star \leq c_i \frac{u_i}{u_i - l_i}(\min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} - l_i) \tag{2.19}$$

for all $i \in \{1, 2, \ldots, n_z\}$ such that $c_i \geq 0$.

On the other hand, for all $i \in \{1, 2, \ldots, n_z\}$ such that $c_i < 0$, we have that

$$\Delta\phi_i^\star \leq c_i \inf\Bigg\{\tilde{z}_i - z_i : z_i = \mathrm{ReLU}(\hat{z}_i),\ \tilde{z}_i \geq 0,\ \tilde{z}_i \geq \hat{\tilde{z}}_i,\ \tilde{z}_i \leq \frac{u_i}{u_i - l_i}(\hat{\tilde{z}}_i - l_i),$$
$$l_i \leq \hat{z}_i, \hat{\tilde{z}}_i \leq u_i,\ |\hat{\tilde{z}}_i - \hat{z}_i| \leq \|w_i\|_* d_{\|\cdot\|}(\mathcal{X}),\ \hat{z}_i, \hat{\tilde{z}}_i \in \mathbb{R}\Bigg\}.$$

The optimal solution to the above infimum is readily found by comparing the line $\tilde{z}_i = 0$ to the function $z_i = \mathrm{ReLU}(\hat{z}_i)$ over $\hat{\tilde{z}}_i, \hat{z}_i \in [l_i, u_i]$. In particular, the minimum value of $\tilde{z}_i - z_i$ on the above feasible set occurs when $\tilde{z}_i = \hat{\tilde{z}}_i = 0$ and $z_i = \hat{z}_i = \|w_i\|_* d_{\|\cdot\|}(\mathcal{X})$. Therefore, we find that

$$\Delta\phi_i^\star \leq -c_i \|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), \tag{2.20}$$

for all $i \in \{1, 2, \ldots, n_z\}$ such that $c_i < 0$. We also note the trivial bound that $\tilde{z}_i - z_i \geq -u_i$ on the feasible set of the above infimum, so that

$$\Delta\phi_i^\star \leq -c_i u_i. \tag{2.21}$$

The inequalities (2.20) and (2.21) together imply that

$$\Delta\phi_i^\star \leq -c_i \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \tag{2.22}$$

for all $i \in \{1, 2, \ldots, n_z\}$ such that $c_i < 0$. Substituting (2.19) and (2.22) into (2.16) gives the desired bound (2.15). □

The value $\Delta\phi_i^\star$ in the proof of Theorem 1 can be interpreted as the worst-case relaxation error in coordinate $i$. From this perspective, Theorem 1 gives an upper bound on the worst-case relaxation error of the overall network. Notice that, since $\min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \leq u_i$, the bound (2.15) immediately gives rise to the simple bound that

$$\Delta\phi^\star(\mathcal{X}) \leq \sum_{i=1}^{n_z} \left(\mathrm{ReLU}(c_i) + \mathrm{ReLU}(-c_i)\right) u_i = \sum_{i=1}^{n_z} |c_i| u_i,$$

the right-hand side of which equals the relaxation error incurred when, at every neuron, the activations of the relaxation solution and the original nonconvex solution are at opposite corners of the relaxed ReLU constraint set, i.e., the convex upper envelope illustrated in Figure 2.3. On the other hand, when $d_{\|\cdot\|}(\mathcal{X})$ is small, i.e., the input uncertainty set is small, one would expect the number of "kinks" in the graph of $f$ over $\mathcal{X}$ to be small, and as a consequence the relaxation error to decrease. This intuition is captured by the bound (2.15), since, in this case, we find that

$$\begin{aligned}
\Delta\phi^\star(\mathcal{X}) &\leq \sum_{i=1}^{n_z} \left( \mathrm{ReLU}(c_i) \frac{u_i}{u_i - l_i} (\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}) - l_i) + \mathrm{ReLU}(-c_i) \|w_i\|_* d_{\|\cdot\|}(\mathcal{X}) \right) \\
&\approx -\sum_{i=1}^{n_z} \mathrm{ReLU}(c_i) \frac{u_i l_i}{u_i - l_i} \\
&\leq \sum_{i=1}^{n_z} |c_i| u_i.
\end{aligned}$$

To continue our development of a branching scheme that is optimal with respect to the worst-case relaxation error, we use (2.15) to bound the relaxation error of the partitioned LP in terms of the row $w_i^\top$ that is used to define the partition. This bound is given in the following lemma.

**Lemma 1.** *Let $i \in \{1, 2, \ldots, n_z\}$. Consider a two-part partition of $\mathcal{X}$ given by $\{\mathcal{X}_i^{(1)}, \mathcal{X}_i^{(2)}\}$, where $\mathcal{X}_i^{(1)} = \{x \in \mathcal{X} : w_i^\top x \geq 0\}$ and $\mathcal{X}_i^{(2)} = \mathcal{X} \setminus \mathcal{X}_i^{(1)}$. Consider also the partitioned relaxation error $\Delta\phi^\star(\{\mathcal{X}_i^{(1)}, \mathcal{X}_i^{(2)}\}) \coloneqq \max_{j\in\{1,2\}} \hat{\phi}_{\mathrm{LP}}^\star(\mathcal{X}_i^{(j)}) - \phi^\star(\mathcal{X})$. It holds that*

$$\begin{aligned}
\Delta\phi^\star(\{\mathcal{X}_i^{(1)}, \mathcal{X}_i^{(2)}\}) \leq\ & |c_i| \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \\
&+ \sum_{\substack{k=1 \\ k\neq i}}^{n_z} \Bigg( \mathrm{ReLU}(c_k) \frac{u_k}{u_k - l_k} (\min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} - l_k) \\
&\qquad + \mathrm{ReLU}(-c_k) \min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} \Bigg).
\end{aligned} \tag{2.23}$$

*Proof.* Consider the relaxation solved over the first input part, $\mathcal{X}_i^{(1)}$, and denote by $l^{(1)}, u^{(1)} \in \mathbb{R}^{n_z}$ the corresponding preactivation bounds. Since $w_i^\top x \geq 0$ on this input part, the preacti-

vation bounds for the first subproblem $\hat{\phi}^{\star}_{\mathrm{LP}}(\mathcal{X}_i^{(1)})$ can be taken as

$$l^{(1)} = (l_1, l_2, \ldots, l_{i-1}, 0, l_{i+1}, \ldots, l_{n_z})$$

and $u^{(1)} = u$. Thus, from (2.15) and the fact that $d_{\|\cdot\|}(\mathcal{X}_i^{(j)}) \leq d_{\|\cdot\|}(\mathcal{X})$ for $j \in \{1, 2\}$, it follows that

$$\begin{aligned}\Delta\phi^{\star}(\mathcal{X}_i^{(1)}) \leq\ & |c_i| \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \\ & + \sum_{\substack{k=1 \\ k\neq i}}^{n_z} \Bigg( \mathrm{ReLU}(c_k) \frac{u_k}{u_k - l_k} (\min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} - l_k) \\ & \qquad + \mathrm{ReLU}(-c_k) \min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} \Bigg).\end{aligned} \tag{2.24}$$

Similarly, over the second input part, $\mathcal{X}_i^{(2)}$, we have that $w_i^{\top} x < 0$, and so the preactivation bounds for the second subproblem $\hat{\phi}^{\star}_{\mathrm{LP}}(\mathcal{X}_i^{(2)})$ can be taken as $l^{(2)} = l$ and

$$u^{(2)} = (u_1, u_2, \ldots, u_{i-1}, 0, u_{i+1}, \ldots, u_{n_z}),$$

resulting in a similar bound to (2.24):

$$\begin{aligned}\Delta\phi^{\star}(\mathcal{X}_i^{(2)}) \leq & \sum_{\substack{k=1 \\ k\neq i}}^{n_z} \Bigg( \mathrm{ReLU}(c_k) \frac{u_k}{u_k - l_k} (\min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} - l_k) \\ & + \mathrm{ReLU}(-c_k) \min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} \Bigg).\end{aligned} \tag{2.25}$$

Putting the two bounds (2.24) and (2.25) together and using the fact that $\phi^{\star}(\mathcal{X}_i^{(j)}) \leq \phi^{\star}(\mathcal{X})$ for all $j \in \{1, 2\}$, we find that

$$\begin{aligned}\Delta\phi^{\star}(\{\mathcal{X}_i^{(1)}, \mathcal{X}_i^{(2)}\}) &= \max_{j\in\{1,2\}} \left( \hat{\phi}^{\star}_{\mathrm{LP}}(\mathcal{X}_i^{(j)}) - \phi^{\star}(\mathcal{X}) \right) \\ &\leq \max_{j\in\{1,2\}} \left( \hat{\phi}^{\star}_{\mathrm{LP}}(\mathcal{X}_i^{(j)}) - \phi^{\star}(\mathcal{X}_i^{(j)}) \right) \\ &= \max_{j\in\{1,2\}} \Delta\phi^{\star}(\mathcal{X}_i^{(j)}) \\ &\leq |c_i| \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \\ &\quad + \sum_{\substack{k=1 \\ k\neq i}}^{n_z} \Bigg( \mathrm{ReLU}(c_k) \frac{u_k}{u_k - l_k} (\min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} - l_k) \\ &\qquad + \mathrm{ReLU}(-c_k) \min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} \Bigg),\end{aligned}$$

as desired. □

**Proposed Branching Scheme**

Lemma 1 bounds the worst-case relaxation error for each possible row-based partition. Therefore, our final step in the development of our branching scheme is to find which row minimizes the upper bound (2.23). This worst-case optimal branching scheme is now presented.

**Theorem 2.** *Consider the two-part partitions defined by the rows of* $W$*:* $\{\mathcal{X}_i^{(1)}, \mathcal{X}_i^{(2)}\}$*, where* $\mathcal{X}_i^{(1)} = \{x \in \mathcal{X} : w_i^\top x \geq 0\}$ *and* $\mathcal{X}_i^{(2)} = \mathcal{X} \setminus \mathcal{X}_i^{(1)}$ *for all* $i \in \{1, 2, \ldots, n_z\} =: \mathcal{I}$*. The optimal partition that minimizes the worst-case relaxation error bound in* (2.23) *is given by*

$$i^\star \in \arg\min_{i \in \mathcal{I}} \mathrm{ReLU}(c_i) \frac{l_i}{u_i - l_i} \left(u_i - \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\}\right). \tag{2.26}$$

*Proof.* Denote the bound in (2.23) of Lemma 1 by

$$\begin{aligned} B(i) \coloneqq & |c_i| \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \\ & + \sum_{\substack{k=1 \\ k \neq i}}^{n_z} \Bigg( \mathrm{ReLU}(c_k) \frac{u_k}{u_k - l_k} (\min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} - l_k) \\ & \qquad + \mathrm{ReLU}(-c_k) \min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} \Bigg), \end{aligned}$$

which is the quantity to be minimized over $i \in \mathcal{I}$. We may rewrite this as

$$\begin{aligned} B(i) = & \sum_{k=1}^{n_z} \Bigg( \mathrm{ReLU}(c_k) \frac{u_k}{u_k - l_k} (\min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} - l_k) \\ & \qquad + \mathrm{ReLU}(-c_k) \min\{\|w_k\|_* d_{\|\cdot\|}(\mathcal{X}), u_k\} \Bigg) \\ & + |c_i| \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \\ & - \Bigg( \mathrm{ReLU}(c_i) \frac{u_i}{u_i - l_i} (\min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} - l_i) \\ & \qquad + \mathrm{ReLU}(-c_i) \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \Bigg). \end{aligned}$$

Hence, using the fact that $|c_i| = \mathrm{ReLU}(c_i) + \mathrm{ReLU}(-c_i)$, we find that

$$\begin{aligned}
\min_{i\in\mathcal{I}} B(i) = \min_{i\in\mathcal{I}} \Bigg( & |c_i| \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \\
& - \Bigg( \mathrm{ReLU}(c_i) \frac{u_i}{u_i - l_i} (\min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} - l_i) \\
& \quad + \mathrm{ReLU}(-c_i) \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \Bigg) \Bigg) \\
= \min_{i\in\mathcal{I}} \Bigg( & \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \Bigg( |c_i| - \mathrm{ReLU}(c_i) \frac{u_i}{u_i - l_i} - \mathrm{ReLU}(-c_i) \Bigg) \\
& + \mathrm{ReLU}(c_i) \frac{u_i l_i}{u_i - l_i} \Bigg) \\
= \min_{i\in\mathcal{I}} \Bigg( & \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \Bigg( \mathrm{ReLU}(c_i) - \mathrm{ReLU}(c_i) \frac{u_i}{u_i - l_i} \Bigg) \\
& + \mathrm{ReLU}(c_i) \frac{u_i l_i}{u_i - l_i} \Bigg) \\
= \min_{i\in\mathcal{I}} & \mathrm{ReLU}(c_i) \frac{l_i}{u_i - l_i} \left( u_i - \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\} \right),
\end{aligned}$$

as desired. □

Theorem 2 provides the branching scheme that optimally reduces the worst-case relaxation error that we seek. We remark its simplicity: to decide which row to partition along, it suffices to enumerate the values $\mathrm{ReLU}(c_i)\frac{l_i}{u_i - l_i}(u_i - \min\{\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}), u_i\})$ for $i \in \{1, 2, \ldots, n_z\}$, then choose the row corresponding to the minimum amongst these values. In practice (especially when using $\|\cdot\| = \|\cdot\|_\infty$), $d_{\|\cdot\|}(\mathcal{X})$ tends to be relatively small, making these values approximately equal to $\mathrm{ReLU}(c_i)\frac{u_i l_i}{u_i - l_i}$. In our numerical simulations that follow, we find that using these simplified values to select the partition does not degrade performance. Note that this optimization over $i$ scales linearly with the dimension $n_z$, and the resulting LP subproblems on each input part only require the addition of one extra linear constraint, meaning that this partitioning scheme is highly efficient.

We also note that Theorem 2 can be immediately extended to design multi-part partitions in two interesting ways. First, by ordering the values $\mathrm{ReLU}(c_i)\frac{u_i l_i}{u_i - l_i}$, we are ordering the optimality of the rows $w_i^\top$ to partition along. Therefore, by partitioning along the $n_p > 1$ rows corresponding to the smallest $n_p$ of these values, Theorem 2 provides a strategy to design an effective $2^{n_p}$-part partition, in the case one prefers to perform more than just a two-part partition. Second, Theorem 2 can be used directly in a branch-and-bound algorithm. See Section 2.5 for implementation details. In our simulations that follow, we find that

this technique works particularly well; our partition yields a branching method for branch-and-bound that outperforms the state-of-the-art per-neuron branching technique for ReLU networks.

### Optimal Partitioning is NP-Hard

In this section, we show that finding a row-based partition that minimizes the actual LP relaxation error is an NP-hard problem. Recall that this approach is in contrast to our previous approach in the sense that our optimal partition in Theorem 2 minimizes the worst-case relaxation error. Consequently, the results of this section show that the partition given by Theorem 2 is in essence an optimal *tractable* LP partitioning scheme.

To start, recall the robustness certification problem for a $K$-layer ReLU neural network:

$$\begin{aligned} \text{maximize} \quad & c^\top x^{[K]} \\ \text{subject to} \quad & x^{[0]} \in \mathcal{X}, \\ & x^{[k+1]} = \mathrm{ReLU}(W^{[k]} x^{[k]}), \quad k \in \{0, 1, \dots, K-1\}, \end{aligned} \tag{2.27}$$

where the optimal value of (2.27) is denoted by $\phi^\star(\mathcal{X})$. Moreover, recall the LP relaxation of (2.27):

$$\begin{aligned} \text{maximize} \quad & c^\top x^{[K]} \\ \text{subject to} \quad & x^{[0]} \in \mathcal{X}, \\ & x^{[k+1]} \geq W^{[k]} x^{[k]}, && k \in \{0, 1, \dots, K-1\}, \\ & x^{[k+1]} \geq 0, && k \in \{0, 1, \dots, K-1\}, \\ & x^{[k+1]} \leq u^{[k+1]} \odot (W^{[k]} x^{[k]} - l^{[k+1]}) \oslash (u^{[k+1]} - l^{[k+1]}), && k \in \{0, 1, \dots, K-1\}. \end{aligned} \tag{2.28}$$

As suggested by the motivating partition of Proposition 3, consider partitioning the input uncertainty set into $2^{n_p}$ parts based on $n_p$ preactivation decision boundaries corresponding to activation functions in the first layer. In particular, for each $j \in \mathcal{J}_p \coloneqq \{j_1, j_2, \dots, j_{n_p}\} \subseteq \{1, 2, \dots, n_1\}$ we partition the input uncertainty set along the hyperplane $w_j^{[0]\top} x^{[0]} = 0$, giving rise to the partition $\{\mathcal{X}^{(1)}, \mathcal{X}^{(2)}, \dots, \mathcal{X}^{(2^{n_p})}\}$. Note that, for all $j \in \mathcal{J}_p$, the partition implies that the $j$th coordinate of the first layer's ReLU equality constraint becomes linear and exact on each part of the partition. Therefore, for all $j' \in \{1, 2, \dots, 2^{n_p}\}$, we may write

the LP relaxation over part $\mathcal{X}^{(j')}$ as

$$\begin{aligned}
\text{maximize} \quad & c^\top x^{[K]} \\
\text{subject to} \quad & x^{[0]} \in \mathcal{X}^{(j')}, \\
& x^{[k+1]} \geq W^{[k]} x^{[k]}, && k \in \{0, 1, \dots, K-1\}, \\
& x^{[k+1]} \geq 0, && k \in \{0, 1, \dots, K-1\}, \\
& x^{[k+1]} \leq u^{[k+1]} \odot (W^{[k]} x^{[k]} - l^{[k+1]}) \oslash (u^{[k+1]} - l^{[k+1]}), && k \in \{0, 1, \dots, K-1\}, \\
& x_j^{[1]} = \mathrm{ReLU}(w_j^{[0]\top} x^{[0]}), && j \in \mathcal{J}_p,
\end{aligned} \tag{2.29}$$

with optimal objective value denoted by $\hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X}^{(j')})$, and thus the partitioned LP relaxation becomes

$$\phi^\star_{\mathcal{J}_p}(\mathcal{X}) \coloneqq \max_{j' \in \{1,2,\dots,2^{n_p}\}} \hat{\phi}^\star_{\mathrm{LP}}(\mathcal{X}^{(j')}). \tag{2.30}$$

To reiterate, the final equality constraint in (2.29) is linear over the restricted feasible set $\mathcal{X}^{(j')}$, which makes the problem (2.30) a partitioned linear program.

If we now allow the indices used to define the partition, namely $\mathcal{J}_p$, to act as a variable, we can search for the optimal $n_p$ rows of the first layer that result in the tightest partitioned LP relaxation. To this end, the problem of optimal partitioning in the first layer is formulated as

$$\begin{aligned}
\underset{\mathcal{J}_p \subseteq \{1,2,\dots,n_1\}}{\text{minimize}} \quad & f^\star_{\mathcal{J}_p}(\mathcal{X}) \\
\text{subject to} \quad & |\mathcal{J}_p| = n_p.
\end{aligned} \tag{2.31}$$

In what follows, we prove the NP-hardness of the optimal partitioning problem (2.31), thereby supporting the use of the worst-case sense optimal partition developed in Theorem 2. To show the hardness of (2.31), we reduce an arbitrary instance of an NP-hard problem, the Min-$\mathcal{K}$-Union problem, to an instance of (2.31). The reduction will show that the Min-$\mathcal{K}$-Union problem can be solved by solving an optimal partitioning problem. Before we proceed, we first recall the definition of the Min-$\mathcal{K}$-Union problem.

**Definition 2** (Min-$\mathcal{K}$-Union problem [66])**.** Consider a collection of $n$ sets $\{\mathcal{S}_1, \mathcal{S}_2, \dots, \mathcal{S}_n\}$, where $\mathcal{S}_j$ is finite for all $j \in \{1, 2, \dots, n\}$, and a positive integer $\mathcal{K} \leq n$. Find $\mathcal{K}$ sets in the collection whose union has minimum cardinality, i.e., find a solution $\mathcal{J}^\star$ of the following optimization problem:

$$\begin{aligned}
\underset{\mathcal{J} \subseteq \{1,2,\dots,n\}}{\text{minimize}} \quad & \left| \bigcup_{j \in \mathcal{J}} \mathcal{S}_j \right| \\
\text{subject to} \quad & |\mathcal{J}| = \mathcal{K}.
\end{aligned} \tag{2.32}$$

Remark the similarities between the optimal partitioning problem and the Min-$\mathcal{K}$-Union problem. In particular, if we think of the convex upper envelopes of the relaxed ReLU

constraints as a collection of sets, then the goal of finding the optimal $n_p$ input coordinates to partition along is intuitively equivalent to searching for the $\mathcal{K} = n_1 - n_p$ convex upper envelopes with minimum size, i.e., those with the least amount of relaxation. This perspective shows that the optimal partitioning problem is essentially a Min-$\mathcal{K}$-Union problem over the collection of relaxed ReLU constraint sets. Since the Min-$\mathcal{K}$-Union problem is NP-hard in general, it is not surprising that the optimal partitioning problem is also NP-hard. Indeed, this result is formalized in the following result.

**Theorem 3.** *Consider the partitioned LP relaxation* (2.30) *of the $K$-layer ReLU neural network certification problem. The optimal partitioning problem in the first-layer, as formulated in* (2.31)*, is NP-hard.*

*Proof.* See Appendix 2.A. □

This concludes our development and analysis for partitioning the LP relaxation. In the next section, we follow a similar line of reasoning to develop a branching scheme for the other popular convex robustness certification technique, i.e., the SDP relaxation. Despite approaching this relaxation from the same partitioning perspective as the LP, the vastly different geometries of the LP and SDP feasible sets make the branching procedures quite distinct.

## 2.4 Partitioned SDP Relaxation

### Tightening of the Relaxation

As with the LP relaxation, we begin by showing that the SDP relaxation error is decreased when the input uncertainty set is partitioned. This proposition is formalized below.

**Proposition 4.** *Consider a neural network with one hidden ReLU layer. Let $\{\mathcal{X}^{(j)} : j \in \{1, 2, \ldots, p\}\}$ be a partition of $\mathcal{X}$. For the $j$th input part $\mathcal{X}^{(j)}$, denote the corresponding input bounds by $l \leq l^{(j)} \leq x \leq u^{(j)} \leq u$, where $x \in \mathcal{X}^{(j)}$. Then, it holds that*

$$\max_{j \in \{1,2,\ldots,p\}} \hat{\phi}^{\star}_{\mathrm{SDP}}(\mathcal{X}^{(j)}) \leq \hat{\phi}^{\star}_{\mathrm{SDP}}(\mathcal{X}). \tag{2.33}$$

*Proof.* Let $j \in \{1, 2, \ldots, p\}$. From the definition of the partition, it holds that $\mathcal{X}^{(j)} \subseteq \mathcal{X}$. What remains to be shown is that $\mathcal{N}^{(j)}_{\mathrm{SDP}} \subseteq \mathcal{N}_{\mathrm{SDP}}$.

Let $P \in \mathcal{N}_{\text{SDP}}^{(j)}$. Define $u' = u^{(j)}$ and $l' = l^{(j)}$. Since $P \in \mathcal{N}_{\text{SDP}}^{(j)}$, it follows that

$$\begin{aligned} P_z &\geq 0, \\ P_z &\geq W P_x, \\ \operatorname{diag}(P_{zz}) &= \operatorname{diag}(W P_{xz}), \\ \operatorname{diag}(P_{xx}) &\leq (l' + u') \odot P_x - l' \odot u', \\ P_1 &= 1, \\ P &\succeq 0. \end{aligned}$$

To show that $P \in \mathcal{N}_{\text{SDP}}$, we should show that the above expressions imply that $\operatorname{diag}(P_{xx}) \leq (l + u) \odot P_x - l \odot u$. To do so, define $\Delta l_i \geq 0$ and $\Delta u_i \geq 0$ such that $l_i' = l_i + \Delta l_i$ and $u_i' = u_i - \Delta u_i$ for all $i \in \{1, 2, \ldots, n_x\}$. Then we find that

$$\begin{aligned} ((l' + u') \odot P_x - l' \odot u')_i &= (l_i' + u_i')(P_x)_i - l_i' u_i' \\ &= (l_i + u_i)(P_x)_i - l_i u_i + (\Delta l_i - \Delta u_i)(P_x)_i \\ &\quad - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i) \\ &= ((l + u) \odot P_x - l \odot u)_i + (\Delta l_i - \Delta u_i)(P_x)_i \\ &\quad - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i) \\ &= ((l + u) \odot P_x - l \odot u)_i + \Delta_i, \end{aligned}$$

where $\Delta_i \coloneqq (\Delta l_i - \Delta u_i)(P_x)_i - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i)$. Therefore, it suffices to prove that $\Delta_i \leq 0$ for all $i$. Since $-l_i u_i \geq -l_i' u_i'$ by definition, it holds that $\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i \geq 0$. Thus, when $(\Delta l_i - \Delta u_i)(P_x)_i \leq 0$, it holds that $\Delta_i \leq 0$, as desired. On the other hand, suppose that $(\Delta l_i - \Delta u_i)(P_x)_i \geq 0$. Then we find two cases:

1. $(\Delta l_i - \Delta u_i) \geq 0$ and $(P_x)_i \geq 0$. In this case, the maximum value of $(\Delta l_i - \Delta u_i)(P_x)_i$ is $(\Delta l_i - \Delta u_i) u_i'$. Therefore, the maximum value of $\Delta_i$ is

$$\begin{aligned} \Delta_i &= (\Delta l_i - \Delta u_i) u_i' - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i) \\ &= \Delta l_i (u_i' - u_i) - \Delta u_i u_i' + \Delta u_i l_i + \Delta u_i \Delta l_i \\ &= \Delta l_i (-\Delta u_i) + \Delta u_i \Delta l_i - \Delta u_i u_i' + \Delta u_i l_i \\ &= -\Delta u_i u_i' + \Delta u_i l_i. \end{aligned}$$

   Both of the two final terms are nonpositive, and therefore $\Delta_i \leq 0$.

2. $(\Delta l_i - \Delta u_i) \leq 0$ and $(P_x)_i \leq 0$. In this case, the maximum value of $(\Delta l_i - \Delta u_i)(P_x)_i$ is $(\Delta l_i - \Delta u_i) l_i'$. Therefore, the maximum value of $\Delta_i$ is

$$\begin{aligned} \Delta_i &= (\Delta l_i - \Delta u_i) l_i' - (\Delta l_i u_i - \Delta u_i l_i - \Delta u_i \Delta l_i) \\ &= -\Delta u_i \Delta l_i + \Delta u_i \Delta l_i + \Delta l_i l_i' - \Delta l_i u_i \\ &= \Delta l_i l_i' - \Delta l_i u_i. \end{aligned}$$

   Both of the two final terms are nonpositive, and therefore $\Delta_i \leq 0$.

Hence, we find that $(l' + u') \odot P_x - l' \odot u' \leq (l + u) \odot P_x - l \odot u$ for all $P \in \mathcal{N}_{\text{SDP}}^{(j)}$, proving that $P \in \mathcal{N}_{\text{SDP}}$, and therefore $\mathcal{N}_{\text{SDP}}^{(j)} \subseteq \mathcal{N}_{\text{SDP}}$.

Since $\mathcal{X}^{(j)} \subseteq \mathcal{X}$ and $\mathcal{N}_{\text{SDP}}^{(j)} \subseteq \mathcal{N}_{\text{SDP}}$, it holds that the solution to the problem over the smaller feasible set lower bounds the original solution: $\hat{\phi}_{\text{SDP}}^{\star}(\mathcal{X}^{(j)}) \leq \hat{\phi}_{\text{SDP}}^{\star}(\mathcal{X})$. Finally, since $j$ was chosen arbitrarily, this implies the desired inequality (2.33). □

Proposition 4 guarantees that partitioning yields a tighter SDP relaxation. However, it is not immediately clear how to design the partition in order to maximally reduce the relaxation error. Indeed, a poorly designed partition may even yield an equality in the bound (2.33). One notable challenge in designing the SDP partition relates to an inherent difference between the SDP relaxation and the LP relaxation. With the LP relaxation, the effect of partitioning can be visualized by how the geometry of the feasible set changes; see Figure 2.4. However, with the SDP, the relaxation comes from dropping the nonconvex rank constraint, the geometry of which is more abstract and harder to exploit.

In the next section, we develop a bound measuring how far the SDP solution is from being rank-1, which corresponds to an exact relaxation, where the improvement in (2.33) is as good as possible. By studying the geometry of the SDP feasible set through this more tractable bound, we find that the partition design for the SDP naturally reduces to a uniform partition along the coordinate axes of the input set.

## Motivating Partition

In this section, we seek the form of a partition that best reduces the SDP relaxation error. By restricting our focus to ReLU networks with one hidden layer, we develop a simple necessary condition for the SDP relaxation to be exact, i.e., for the matrix $P$ to be rank-1. We then work on the violation of this condition to define a measure of how close $P$ is to being rank-1 in the case it has higher rank. Next, we develop a tractable upper bound on this rank-1 gap. Finally, we formulate an optimization problem in which we search for a partition of the input uncertainty set that minimizes our upper bound. We show that such a worst-case optimal partition takes the form of a uniform division of the input set. The result motivates the use of uniform partitions of the input uncertainty set, and below, we answer the question of which coordinate is best to uniformly partition along. Note that, despite the motivating partition being derived for networks with one hidden layer, the relaxation tightening in Proposition 4 still holds for multi-layer networks. Indeed, the numerical simulations in Section 2.6 will show that the resulting SDP branching scheme design maintains a relatively constant efficacy as the number of layers increases.

**Proposition 5.** *Let $P^{\star} \in \mathbb{S}^{n_x+n_z+1}$ denote a solution to the semidefinite programming relaxation* (2.8)*. If the relaxation is exact, meaning that* $\text{rank}(P^{\star}) = 1$*, then the following conditions hold:*

$$\text{tr}(P_{xx}^{\star}) = \|P_x^{\star}\|_2^2, \quad \text{tr}(P_{zz}^{\star}) = \|P_z^{\star}\|_2^2. \tag{2.34}$$

*Proof.* Since the SDP relaxation is exact, it holds that $\operatorname{rank}(P^\star) = 1$. Therefore, $P^\star$ can be expressed as

$$P^\star = \begin{bmatrix} 1 \\ v \\ w \end{bmatrix} \begin{bmatrix} 1 & v^\top & w^\top \end{bmatrix} = \begin{bmatrix} 1 & v^\top & w^\top \\ v & vv^\top & vw^\top \\ w & wv^\top & ww^\top \end{bmatrix}$$

for some vectors $v \in \mathbb{R}^{n_x}$ and $w \in \mathbb{R}^{n_z}$. Recall the block decomposition of $P^\star$:

$$P^\star = \begin{bmatrix} P_1^\star & P_x^{\star\top} & P_z^{\star\top} \\ P_x^\star & P_{xx}^\star & P_{xz}^\star \\ P_z^\star & P_{zx}^\star & P_{zz}^\star \end{bmatrix}.$$

Equating coefficients, we find that $P_{xx}^\star = vv^\top = P_x^\star P_x^{\star\top}$ and $P_{zz}^\star = ww^\top = P_z^\star P_z^{\star\top}$. Therefore,

$$\operatorname{tr}(P_{xx}^\star) = \operatorname{tr}(P_x^\star P_x^{\star\top}) = \operatorname{tr}(P_x^{\star\top} P_x^\star) = \|P_x^\star\|_2^2,$$

proving the first condition in (2.34). The second condition follows in the same way. □

Enforcing the conditions (2.34) as constraints in the SDP relaxation may assist in pushing the optimization variable $P$ towards a rank-1 solution. However, because the conditions in (2.34) are nonlinear equality constraints in the variable $P$, we cannot impose them directly on the SDP without making the problem nonconvex. Instead, we will develop a convex method based on the rank-1 conditions (2.34) that can be used to motivate the SDP solution to have a lower rank.

In the general case that $\operatorname{rank}(P) = r \geq 1$, $P$ may be written as $P = VV^\top$, where

$$V = \begin{bmatrix} e^\top \\ X \\ Z \end{bmatrix}, \ e \in \mathbb{R}^r, \ X \in \mathbb{R}^{n_x \times r}, \ Z \in \mathbb{R}^{n_z \times r},$$

and where the vector $e$ satisfies the equation $e^\top e = \|e\|_2^2 = 1$. The $i$th row of $X$ (respectively, $Z$) is denoted by $X_i^\top \in \mathbb{R}^{1\times r}$ (respectively, $Z_i^\top \in \mathbb{R}^{1\times r}$). Under this expansion, we find that $P_x = Xe$, $P_z = Ze$, $P_{xx} = XX^\top$, and $P_{zz} = ZZ^\top$. Therefore, the conditions (2.34) can be written as

$$\operatorname{tr}(XX^\top) = \|Xe\|_2^2, \quad \operatorname{tr}(ZZ^\top) = \|Ze\|_2^2.$$

To simplify the subsequent analysis, we will restrict our attention to the first of these two necessary conditions for $P$ to be rank-1. As the simulation results in Section 2.6 show, this restriction still yields significant reduction in relaxation error. Now, note that

$$\operatorname{tr}(XX^\top) = \sum_{i=1}^{n_x} (XX^\top)_{ii} = \sum_{i=1}^{n_x} \|X_i\|_2^2,$$

where $(XX^\top)_{ii}$ is the $(i,i)$ element of the matrix $XX^\top$, and also that

$$\|Xe\|_2^2 = \sum_{i=1}^{n_x} (Xe)_i^2 = \sum_{i=1}^{n_x} (X_i^\top e)^2,$$

where $(Xe)_i$ is the $i$th element of the vector $Xe$. Therefore, the rank-1 necessary condition is equivalently written as

$$g(P) \coloneqq \sum_{i=1}^{n_x} (\|X_i\|_2^2 - (X_i^\top e)^2) = 0,$$

where $g(P)$ serves as a measure of the rank-1 gap. Note that $g(P)$ is solely determined by $P = VV^\top$, even though it is written in terms of $X$ and $e$, which are blocks of $V$. In general, $g(P) \geq 0$ when $\text{rank}(P) \geq 1$.

**Lemma 2.** *Let $P \in \mathbb{S}^{n_x+n_z+1}$ be an arbitrary feasible point for the SDP relaxation* (2.8)*. The rank-1 gap $g(P)$ is nonnegative, and is zero if $P$ is rank-1.*

*Proof.* By the Cauchy-Schwarz inequality, we have that $|X_i^\top e| \leq \|X_i\|_2 \|e\|_2$ for all $i \in \{1, 2, \ldots, n_x\}$. Since $P$ is feasible for (2.8) we also have that $\|e\|_2 = 1$, so squaring both sides of the inequality gives that $(X_i^\top e)^2 \leq \|X_i\|_2^2$. Summing these inequalities over $i$ gives

$$g(P) = \sum_{i=1}^{n_x} (\|X_i\|_2^2 - (X_i^\top e)^2) \geq 0.$$

If $P$ is rank-1, then the dimension $r$ of the vectors $e$ and $X_i$ is equal to 1. That is, $e, X_i \in \mathbb{R}$. Hence, $\|X_i\|_2 = |X_i|$ and $|e| = \|e\|_2 = 1$, yielding $\|X_i\|_2^2 - (X_i^\top e)^2 = X_i^2 - X_i^2 e^2 = 0$. Therefore, $g(P) = 0$ in the case that $\text{rank}(P) = 1$. □

Since $g(P) = 0$ is necessary for $P$ to be rank-1 and $g(P) \geq 0$, it is desirable to make $g(P^\star)$ as small as possible at the optimal solution $P^\star$ of the partitioned SDP relaxation. Indeed, this is our partitioning motivation: we seek to partition the input uncertainty set to minimize $g(P^\star)$, in order to influence $P^\star$ to be of low rank. However, there is a notable hurdle with this approach. In particular, the optimal solution $P^\star$ depends on the partition we choose, and finding a partition to minimize $g(P^\star)$ in turn depends on $P^\star$ itself. To overcome this cyclic dependence, we propose first bounding $g(P^\star)$ by a worst-case upper bound, and then choosing an optimal partition to minimize the upper bound. This will make the partition design tractable, resulting in a closed-form solution.

To derive the upper bound on the rank-1 gap at optimality, let $\{\mathcal{X}^{(j)} : j \in \{1, 2, \ldots, p\}\}$ denote the partition of $\mathcal{X}$. For the $j$th input part $\mathcal{X}^{(j)}$, denote the corresponding input bounds by $l^{(j)}, u^{(j)}$. The upper bound is derived below.

**Lemma 3.** *The rank-1 gap at the solution $P^\star$ of the partitioned SDP satisfies*

$$0 \leq g(P^\star) \leq \frac{1}{4} \sum_{i=1}^{n_x} \max_{j \in \{1,2,\ldots,p\}} (u_i^{(j)} - l_i^{(j)})^2. \tag{2.35}$$

*Proof.* The left inequality is a direct result of Lemma 2. For the right inequality, note that

$$g(P^\star) \leq \max_{j\in\{1,2,\ldots,p\}} \sup_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ P_x\in\mathcal{X}^{(j)}} g(P) \leq \sum_{i=1}^{n_x} \max_{j\in\{1,2,\ldots,p\}} \sup_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ P_x\in\mathcal{X}^{(j)}} (\|X_i\|_2^2 - (X_i^\top e)^2). \tag{2.36}$$

Let us focus on the optimization over the $j$th part of the partition, namely,

$$\sup_{P\in\mathcal{N}_{\text{SDP}}^{(j)},\ P_x\in\mathcal{X}^{(j)}} (\|X_i\|_2^2 - (X_i^\top e)^2).$$

To bound this quantity, we analyze the geometry of the SDP relaxation over part $j$, following the methodology of Raghunathan, Steinhardt, and Liang [116]; see Figure 2.5.

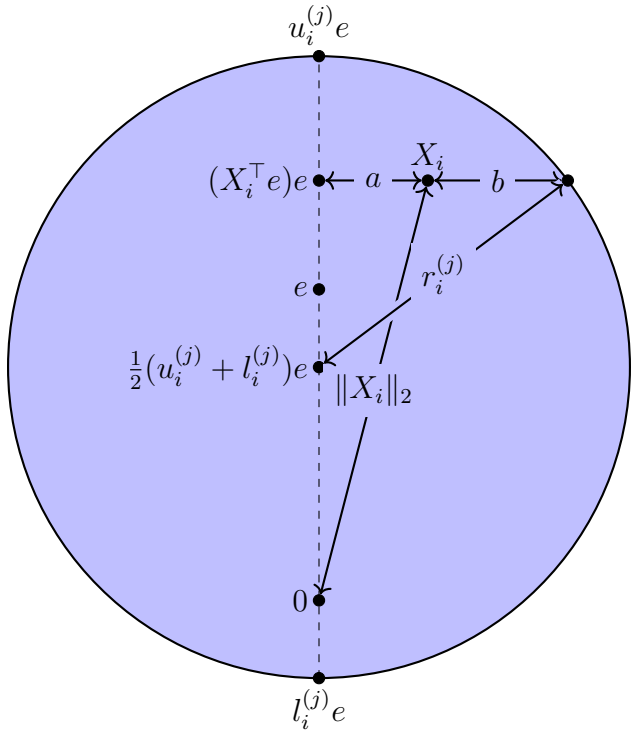



Figure 2.5: Geometry of the SDP relaxation in coordinate $i$ over part $j$ of the partition. The shaded region shows the feasible $X_i$ satisfying the input constraint [116].

The shaded circle represents the set of feasible $X_i$ over part $j$ of the partition, namely, those satisfying the $i$th coordinate of the constraint $\text{diag}(P_{xx}) \leq (l^{(j)} + u^{(j)}) \odot P_x - l^{(j)} \odot u^{(j)}$. To understand this, note that the constraint is equivalent to $\|X_i\|_2^2 \leq (l_i^{(j)} + u_i^{(j)})X_i^\top e - l_i^{(j)} u_i^{(j)}$, or, more geometrically written, that $\|X_i - \frac{1}{2}(u_i^{(j)} + l_i^{(j)})e\|_2^2 \leq \left(\frac{1}{2}(u_i^{(j)} - l_i^{(j)})\right)^2$. This shows

that $X_i$ is constrained to a 2-norm ball of radius $r_i^{(j)} = \frac{1}{2}(u_i^{(j)} - l_i^{(j)})$ centered at $\frac{1}{2}(u_i^{(j)} + l_i^{(j)})e$, as shown in Figure 2.5.

The geometry of Figure 2.5 immediately shows that $\|X_i\|_2^2 = a^2 + (X_i^\top e)^2$ and $r_i^{(j)2} = (a+b)^2 + (X_i^\top e - \frac{1}{2}(u_i^{(j)} + l_i^{(j)}))^2$, and therefore

$$\|X_i\|_2^2 - (X_i^\top e)^2 = a^2 = r_i^{(j)2} - (X_i^\top e - \tfrac{1}{2}(u_i^{(j)} + l_i^{(j)}))^2 - 2ab - b^2.$$

Since $a$ and $b$ are nonnegative,

$$\begin{aligned}
&\sup_{P \in \mathcal{N}_{\text{SDP}}^{(j)},\ P_x \in \mathcal{X}^{(j)}} \|X_i\|_2^2 - (X_i^\top e)^2 \\
&\quad = \sup_{P \in \mathcal{N}_{\text{SDP}}^{(j)},\ P_x \in \mathcal{X}^{(j)}} (r_i^{(j)2} - (X_i^\top e - \tfrac{1}{2}(u_i^{(j)} + l_i^{(j)}))^2 - 2ab - b^2) \\
&\quad \leq \sup_{P \in \mathcal{N}_{\text{SDP}}^{(j)},\ P_x \in \mathcal{X}^{(j)}} (r_i^{(j)2} - (X_i^\top e - \tfrac{1}{2}(u_i^{(j)} + l_i^{(j)}))^2) \\
&\quad \leq r_i^{(j)2} \\
&\quad = \frac{1}{4}(u_i^{(j)} - l_i^{(j)})^2.
\end{aligned}$$

Thus, (2.36) gives

$$g(P^\star) \leq \frac{1}{4} \sum_{i=1}^{n_x} \max_{j \in \{1,2,\ldots,p\}} (u_i^{(j)} - l_i^{(j)})^2,$$

as desired. □

With Lemma 3 in place, we now have an upper bound on the rank-1 gap at optimality, in terms of the input bounds $\{l^{(j)}, u^{(j)}\}_{j=1}^p$ associated with the partition. At this point, we turn to minimizing the upper bound over all valid choices of $p$-part partitions of the input uncertainty set along a given coordinate. Note that, in order for $\{l^{(j)}, u^{(j)}\}_{j=1}^p$ to define valid input bounds for a $p$-part partition, it must be that the union of the input parts cover the input uncertainty set. In terms of the input bounds, this leads to the constraint that $[l, u] = \cup_{j=1}^p [l^{(j)}, u^{(j)}]$, where $[l, u] \coloneqq [l_1, u_1] \times [l_2, u_2] \times \cdots \times [l_{n_x}, u_{n_x}]$, and similarly for $[l^{(j)}, u^{(j)}]$. Since we consider the partition along a single coordinate $k$, this constraint becomes equivalent to $\cup_{j=1}^p [l_k^{(j)}, u_k^{(j)}] = [l_k, u_k]$, because all other coordinates $i \neq k$ satisfy $l_i^{(j)} = l_i$ and $u_i^{(j)} = u_i$ for all $j$ by assumption. We now give the worst-case optimal partitioning scheme for the SDP that minimizes the upper bound in Lemma 3.

**Theorem 4.** *Let $\mathcal{I}_k = \{1, 2, \dots, n_x\} \setminus \{k\}$. Consider the optimization problem of finding the partition to minimize the upper bound* (2.35)*, namely*

$$\begin{aligned} \underset{\mathcal{P}=\{l^{(j)},u^{(j)}\}_{j=1}^{p} \subseteq \mathbb{R}^{n_x}}{\text{minimize}} \quad & h(\mathcal{P}) = \sum_{i=1}^{n_x} \max_{j \in \{1,2,\dots,p\}} (u_i^{(j)} - l_i^{(j)})^2 \\ \text{subject to} \quad & \bigcup_{j=1}^{p} [l_k^{(j)}, u_k^{(j)}] = [l_k, u_k], && i \in \mathcal{I}_k,\ j \in \{1, 2, \dots, p\}, \\ & l_i^{(j)} = l_i, && i \in \mathcal{I}_k,\ j \in \{1, 2, \dots, p\}, \\ & u_i^{(j)} = u_i, && i \in \mathcal{I}_k,\ j \in \{1, 2, \dots, p\}, \end{aligned} \tag{2.37}$$

*Consider also the uniform partition defined by $\bar{\mathcal{P}} = \{\bar{l}^{(j)}, \bar{u}^{(j)}\}_{j=1}^{p} \subseteq \mathbb{R}^{n_x}$, where*

$$\bar{l}_i^{(j)} = \begin{cases} \frac{j-1}{p}(u_i - l_i) + l_i & \text{if } i = k, \\ l_i & \text{otherwise}, \end{cases}$$

$$\bar{u}_i^{(j)} = \begin{cases} \frac{j}{p}(u_i - l_i) + l_i & \text{if } i = k, \\ u_i & \text{otherwise}, \end{cases}$$

*for all $j \in \{1, 2, \dots, p\}$. It holds that $\bar{\mathcal{P}}$ is a solution to* (2.37)*.*

*Proof.* To prove the result, we show that the proposed $\bar{\mathcal{P}}$ is feasible for the optimization, and that $h(\bar{\mathcal{P}}) \leq h(\mathcal{P})$ for all feasible $\mathcal{P}$. First, note that it is obvious by the definition of $\bar{\mathcal{P}}$ that $\bar{l}_i^{(j)} = l_i$ and $\bar{u}_i^{(j)} = u_i$ for all $i \in \{1, 2, \dots, n_x\} \setminus \{k\}$ and all $j \in \{1, 2, \dots, p\}$. Therefore, to prove that $\bar{\mathcal{P}}$ is feasible, it suffices to show that $\cup_{j=1}^{p} [\bar{l}_k^{(j)}, \bar{u}_k^{(j)}] = [l_k, u_k]$. Indeed, since

$$\bar{u}_k^{(j)} = \frac{j}{p}(u_k - l_k) + l_k = \frac{(j+1) - 1}{p}(u_k - l_k) + l_k = \bar{l}_k^{(j+1)}$$

for all $j \in \{1, 2, \dots, p-1\}$,

$$\bar{l}_k^{(1)} = \frac{1-1}{p}(u_k - l_k) + l_k = l_k,$$

and

$$\bar{u}_k^{(p)} = \frac{p}{p}(u_k - l_k) + l_k = u_k,$$

we have that

$$\bigcup_{j=1}^{p} [\bar{l}_k^{(j)}, \bar{u}_k^{(j)}] = [\bar{l}_k^{(1)}, \bar{u}_k^{(1)}] \cup [\bar{l}_k^{(2)}, \bar{u}_k^{(2)}] \cup \cdots \cup [\bar{l}_k^{(p)}, \bar{u}_k^{(p)}] = [\bar{l}_k^{(1)}, \bar{u}_k^{(p)}] = [l_k, u_k].$$

Hence, $\bar{\mathcal{P}} = \{\bar{l}^{(j)}, \bar{u}^{(j)}\}_{j=1}^{p}$ is feasible.

The objective at the proposed feasible point can be computed as

$$\begin{aligned}
h(\bar{\mathcal{P}}) &= \sum_{i=1}^{n_x} \max_{j\in\{1,2,\dots,p\}} (\bar{u}_i^{(j)} - \bar{l}_i^{(j)})^2 \\
&= \sum_{\substack{i=1\\ i\neq k}}^{n_x} \max_{j\in\{1,2,\dots,p\}} (\bar{u}_i^{(j)} - \bar{l}_i^{(j)})^2 + \max_{j\in\{1,2,\dots,p\}} (\bar{u}_k^{(j)} - \bar{l}_k^{(j)})^2 \\
&= \sum_{\substack{i=1\\ i\neq k}}^{n_x} \max_{j\in\{1,2,\dots,p\}} (u_i - l_i)^2 + \max_{j\in\{1,2,\dots,p\}} \left(\frac{j}{p}(u_k - l_k) + l_k - \frac{j-1}{p}(u_k - l_k) - l_k\right)^2 \\
&= \sum_{\substack{i=1\\ i\neq k}}^{n_x} (u_i - l_i)^2 + \max_{j\in\{1,2,\dots,p\}} \left(\frac{1}{p}(u_k - l_k)\right)^2 \\
&= C + \frac{1}{p^2}(u_k - l_k)^2,
\end{aligned}$$

where $C \coloneqq \sum_{\substack{i=1\\ i\neq k}}^{n_x} (u_i - l_i)^2$. Now, let $\mathcal{P} = \{l^{(j)}, u^{(j)}\}_{j=1}^{p}$ be an arbitrary feasible point for the optimization (2.37). Then by a similar analysis as above, the objective value at $\mathcal{P}$ satisfies

$$\begin{aligned}
h(\mathcal{P}) &= \sum_{i=1}^{n_x} \max_{j\in\{1,2,\dots,p\}} (u_i^{(j)} - l_i^{(j)})^2 \\
&= \sum_{\substack{i=1\\ i\neq k}}^{n_x} \max_{j\in\{1,2,\dots,p\}} (u_i^{(j)} - l_i^{(j)})^2 + \max_{j\in\{1,2,\dots,p\}} (u_k^{(j)} - l_k^{(j)})^2 \\
&= C + \max_{j\in\{1,2,\dots,p\}} (u_k^{(j)} - l_k^{(j)})^2 \\
&= C + \left(\max_{j\in\{1,2,\dots,p\}} (u_k^{(j)} - l_k^{(j)})\right)^2 \\
&\geq C + \left(\frac{1}{p}\sum_{j=1}^{p} (u_k^{(j)} - l_k^{(j)})\right)^2 \\
&= C + \frac{1}{p^2}\left(\sum_{j=1}^{p} (u_k^{(j)} - l_k^{(j)})\right)^2.
\end{aligned}$$

Since $\mathcal{P}$ is feasible, it holds that $[l_k, u_k] = \bigcup_{j=1}^{p} [l_k^{(j)}, u_k^{(j)}]$. Therefore, by subadditivity of Lebesgue measure $\mu$ on the Borel $\sigma$-algebra of $\mathbb{R}$, we have that

$$u_k - l_k = \mu([l_k, u_k]) = \mu\left(\bigcup_{j=1}^{p} [l_k^{(j)}, u_k^{(j)}]\right) \leq \sum_{j=1}^{p} \mu([l_k^{(j)}, u_k^{(j)}]) = \sum_{j=1}^{p} (u_k^{(j)} - l_k^{(j)}).$$

Substituting this into our above expressions, we conclude that

$$h(\bar{\mathcal{P}}) = C + \frac{1}{p^2}(u_k - l_k)^2 \leq C + \frac{1}{p^2}\left(\sum_{j=1}^{p}(u_k^{(j)} - l_k^{(j)})\right)^2 \leq h(\mathcal{P}).$$

Since $\mathcal{P}$ was an arbitrary feasible point for the optimization, this implies that $\bar{\mathcal{P}}$ is a solution to the optimization. □

Theorem 4 shows that by choosing the partition of the input set to be uniformly divided amongst the $p$ parts, we obtain a partition that minimizes the worst-case bound on the gap of the rank-1 necessary condition (2.34). This gives a well-motivated, yet simple way to design a partition of the input uncertainty set in order to push the SDP relaxation towards being rank-1, thereby reducing relaxation error.

## SDP Branching Scheme

With the motivating SDP partition now established, we turn our attention from the *form* of a worst-case optimal SDP partition to the *coordinate* of a worst-case optimal partition. In particular, we seek to find the best branching scheme to minimize relaxation error of the SDP. Our above results suggest using a uniform partition, and in this section we seek to find which coordinate to apply the partitioning to. Similar to the LP relaxation, we derive an optimal branching scheme by first bounding the relaxation error in the worst-case sense.

### Worst-Case Relaxation Bound

In the worst-case relaxation bound of Theorem 5 below, and the subsequent worst-case optimal SDP branching scheme proposed in Theorem 6, we restrict our attention to a single hidden ReLU layer and make the following assumption on the weight matrix.

**Assumption 1.** The rows of the weight matrix are assumed to be normalized with respect to the $\ell_1$-norm, i.e., that $\|w_i\|_1 = 1$ for all $i \in \{1, 2, \ldots, n_z\}$.

We briefly remark that Assumption 1 imposes no loss of generality, as it can be made to hold for any network by a simple rescaling. In particular, if the assumption does not hold,

the network architecture can be rescaled as follows:

$$z = \mathrm{ReLU}(Wx) = \mathrm{ReLU}\left(\begin{bmatrix} w_1^\top \\ w_2^\top \\ \vdots \\ w_{n_z}^\top \end{bmatrix} x\right)$$

$$= \mathrm{ReLU}\left(\mathrm{diag}(\|w_1\|_1, \|w_2\|_1, \ldots, \|w_{n_z}\|_1) \begin{bmatrix} \frac{w_1^\top}{\|w_1\|_1} \\ \frac{w_2^\top}{\|w_2\|_1} \\ \vdots \\ \frac{w_{n_z}^\top}{\|w_{n_z}\|_1} \end{bmatrix} x\right) = W_{\mathrm{scale}}\,\mathrm{ReLU}(W_{\mathrm{norm}}x),$$

where $W_{\mathrm{scale}} = \mathrm{diag}(\|w_1\|_1, \|w_2\|_1, \ldots, \|w_{n_z}\|_1) \in \mathbb{R}^{n_z \times n_z}$ and

$$W_{\mathrm{norm}} = \begin{bmatrix} \frac{w_1^\top}{\|w_1\|_1} \\ \frac{w_2^\top}{\|w_2\|_1} \\ \vdots \\ \frac{w_{n_z}^\top}{\|w_{n_z}\|_1} \end{bmatrix} \in \mathbb{R}^{n_z \times n_x}$$

are the scaling and normalized factors of the weight matrix $W$, respectively. The scaling factor can therefore be absorbed into the optimization cost vector $c$, yielding a problem with normalized rows as desired.

Before introducing the worst-case relaxation bound of Theorem 5, we state a short lemma that will be used in proving the relaxation bound.

**Lemma 4.** *Let $P \in \mathbb{S}^n$ be a positive semidefinite matrix. Then $|P_{ij}| \leq \frac{1}{2}(P_{ii} + P_{jj})$ for all $i, j \in \{1, 2, \ldots, n\}$.*

*Proof.* Let $i, j \in \{1, 2, \ldots, n\}$. Since $P$ is positive semidefinite, the 2nd-order principal minor $P_{ii}P_{jj} - P_{ij}^2$ is nonnegative, and therefore

$$|P_{ij}| \leq \sqrt{P_{ii}P_{jj}}. \tag{2.38}$$

Furthermore, by the basic inequality that $2ab \leq a^2 + b^2$ for all $a, b \in \mathbb{R}$, we have that $\sqrt{P_{ii}P_{jj}} \leq \frac{1}{2}(P_{ii} + P_{jj})$. Substituting this inequality into (2.38) gives the desired bound. □

**Theorem 5.** *Consider a feedforward ReLU neural network with one hidden layer, and with the input uncertainty set $\mathcal{X}$. Let the network have input bounds $l, u \in \mathbb{R}^{n_x}$ and preactivation bounds $\hat{l}, \hat{u} \in \mathbb{R}^{n_z}$. Consider also the relaxation error $\Delta\phi^\star_{\mathrm{SDP}}(\mathcal{X}) \coloneqq \hat{\phi}^\star_{\mathrm{SDP}}(\mathcal{X}) - \phi^\star(\mathcal{X})$. Let $P^\star$ and $(x^\star, z^\star)$ be optimal solutions for the relaxation $\hat{\phi}^\star_{\mathrm{SDP}}(\mathcal{X})$ and the unrelaxed problem $\phi^\star(\mathcal{X})$, respectively. Given an arbitrary norm $\|\cdot\|$ on $\mathbb{R}^{n_x}$, it holds that*

$$\Delta\phi^\star_{\mathrm{SDP}}(\mathcal{X}) \leq \sum_{i=1}^{n_z} \left(\mathrm{ReLU}(c_i)q(l,u) + \mathrm{ReLU}(-c_i)\min\{\hat{u}_i, \|w_i\|_* d_{\|\cdot\|}(\mathcal{X})\}\right), \tag{2.39}$$

*where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$, and where*

$$q(l,u) = \max_{k\in\{1,2,\ldots,n_x\}} \max\{|l_k|,|u_k|\}.$$

*Proof.* First, recall that $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ is assumed to be compact, and is therefore bounded, and hence $d_{\|\cdot\|}(\mathcal{X}) < \infty$. The definitions of $P_x^\star$ and $(x^\star, z^\star)$ give that

$$\Delta\phi^\star_{\text{SDP}}(\mathcal{X}) = \sum_{i=1}^{n_z} c_i((P_z^\star)_i - z_i^\star) \leq \sum_{i=1}^{n_z} \Delta\phi_i^\star, \tag{2.40}$$

where

$$\Delta\phi_i^\star = \sup\Big\{ c_i((P_z)_i - z_i) : z_i = \text{ReLU}(w_i^\top x),\ P_z \geq 0,\ P_z \geq WP_x,$$
$$\text{diag}(P_{zz}) = \text{diag}(WP_{xz}),\ P_1 = 1,\ P \succeq 0,\ x, P_x \in \mathcal{X}\Big\}$$

for all $i \in \{1,2,\ldots,n_z\}$. Defining the auxiliary variables $P_{\hat{z}} = WP_x$ and $\hat{z} = Wx$, this is equivalent to

$$\Delta\phi_i^\star = \sup\Big\{ c_i((P_z)_i - z_i) : z_i = \text{ReLU}(\hat{z}_i),\ P_z \geq 0,\ P_z \geq P_{\hat{z}},\ \text{diag}(P_{zz}) = \text{diag}(WP_{xz}),$$
$$P_1 = 1,\ P \succeq 0,\ P_{\hat{z}} = WP_x,\ \hat{z}_i = w_i^\top x,\ x, P_x \in \mathcal{X}\Big\}.$$

If $x, P_x \in \mathcal{X}$ and $\hat{z}, P_{\hat{z}}$ satisfy $\hat{z} = Wx$ and $P_{\hat{z}} = WP_x$, then $|(P_{\hat{z}})_i - \hat{z}_i| = |w_i^\top(P_x - x)| \leq \|w_i\|_*\|P_x - x\| \leq \|w_i\|_* d_{\|\cdot\|}(\mathcal{X})$ for all $i \in \{1,2,\ldots,n_z\}$ by the Cauchy-Schwarz inequality for dual norms. Therefore,

$$\Delta\phi_i^\star \leq \sup\Big\{ c_i((P_z)_i - z_i) : z_i = \text{ReLU}(\hat{z}_i),\ P_z \geq 0,\ P_z \geq P_{\hat{z}},$$
$$\text{diag}(P_{zz}) = \text{diag}(WP_{xz}),\ P_1 = 1,\ P \succeq 0,\ \hat{l} \leq \hat{z}, P_{\hat{z}} \leq \hat{u},$$
$$|(P_{\hat{z}})_k - \hat{z}_k| \leq \|w_k\|_* d_{\|\cdot\|}(\mathcal{X}) \text{ for all } k \in \{1,2,\ldots,n_z\},\ \hat{z}, P_{\hat{z}} \in \mathbb{R}^{n_z}\Big\}.$$

We now translate the optimization variables in the above problem from $\hat{z} \in \mathbb{R}^{n_z}$ and $P \in \mathbb{S}^{1+n_x+n_z}$ to the scalars $\hat{z}_i, (P_{\hat{z}})_i \in \mathbb{R}$. To this end, we note that if $P$ is feasible for the above supremum, then

$$\text{diag}(P_{zz})_i = \text{diag}(WP_{xz})_i = w_i^\top (P_{xz})_i \leq \|(P_{xz})_i\|_\infty \|w_i\|_1,$$

where $(P_{xz})_i$ is the $i$th column of the matrix $P_{xz}$, and the inequality again comes from Cauchy-Schwarz. By the weight matrix scaling assumption, this yields

$$\text{diag}(P_{zz})_i \leq \|(P_{xz})_i\|_\infty.$$

Now, since $P$ is positive semidefinite, Lemma 4 gives that

$$\begin{aligned}\|(P_{xz})_i\|_\infty &= \max_{k\in\{1,2,\dots,n_x\}} |(P_{xz})_i|_k = \max_{k\in\{1,2,\dots,n_x\}} |(P_{xz})_{ki}| \\ &\leq \max_{k\in\{1,2,\dots,n_z\}} \frac{1}{2}\Big((P_{xx})_{kk} + (P_{zz})_{ii}\Big) = \frac{1}{2}(P_{zz})_{ii} + \frac{1}{2}\max_{k\in\{1,2,\dots,n_x\}} (P_{xx})_{kk}.\end{aligned}$$

Noting that $(P_{zz})_{ii} = \operatorname{diag}(P_{zz})_i$, the bound of interest becomes

$$\operatorname{diag}(P_{zz})_i \leq \max_{k\in\{1,2,\dots,n_z\}} (P_{xx})_{kk}.$$

We now seek to bound $(P_{xx})_{kk}$. Recall that $(P_{xx})_{kk} = \operatorname{diag}(P_{xx})_k \leq (l_k + u_k)(P_x)_k - l_k u_k$. If $(l_k + u_k) \geq 0$, then $(P_x)_k \leq u_k$ implies that $(l_k + u_k)(P_x)_k \leq (l_k + u_k)u_k$, and therefore $(P_{xx})_{kk} \leq (l_k + u_k)u_k - l_k u_k = u_k^2$. On the other hand, if $(l_k + u_k) < 0$, then $(P_x)_k \geq l_k$ implies that $(l_k + u_k)(P_x)_k \leq (l_k + u_k)l_k$, and therefore $(P_{xx})_{kk} \leq (l_k + u_k)l_k - l_k u_k = l_k^2$. Hence, in all cases, it holds that

$$(P_{xx})_{kk} \leq \mathbb{I}(l_k + u_k \geq 0)u_k^2 + \mathbb{I}(l_k + u_k < 0)l_k^2.$$

We can further simplify this bound as follows. If $l_k + u_k \geq 0$, then $u_k \geq -l_k$ and $u_k \geq l_k$, implying $|l_k| \leq u_k$, so $l_k^2 \leq u_k^2$ and therefore $u_k^2 = \max\{l_k^2, u_k^2\}$. On the other hand, if $l_k + u_k < 0$, then an analogous argument shows that $l_k^2 = \max\{l_k^2, u_k^2\}$. Hence, we conclude that the above bound on $(P_{xx})_{kk}$ can be rewritten as

$$(P_{xx})_{kk} \leq \max\{l_k^2, u_k^2\}.$$

Therefore, returning to the bound on $(P_{zz})_i$, we find that

$$\operatorname{diag}(P_{zz})_i \leq \max_{k\in\{1,2,\dots,n_x\}} \max\{l_k^2, u_k^2\},$$

for all $i \in \{1, 2, \dots, n_z\}$. Now, note that since $P \succeq 0$, the Schur complement gives that

$$\begin{bmatrix} P_{xx} - P_x P_x^\top & P_{xz} - P_x P_z^\top \\ P_{xz}^\top - P_z P_x^\top & P_{zz} - P_z P_z^\top \end{bmatrix} \succeq 0,$$

which implies that

$$\operatorname{diag}(P_{zz}) \geq \operatorname{diag}(P_z P_z^\top) = P_z \odot P_z.$$

Therefore, our upper bound on the diagonal elements of $P_{zz}$ yields that

$$(P_z)_i \leq \max_{k\in\{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\} = q(l, u).$$

Hence, we have derived a condition on the component $(P_z)_i$ that all feasible $P$ must satisfy. The supremum of interest may now be further upper bounded giving rise to

$$\begin{aligned}\Delta\phi_i^\star \leq \sup\Big\{ & c_i((P_z)_i - z_i) : z_i = \operatorname{ReLU}(\hat{z}_i),\ (P_z)_i \geq 0,\ (P_z)_i \geq (P_{\hat{z}})_i,\ (P_z)_i \leq q(l,u), \\ & \hat{l}_i \leq \hat{z}_i, (P_{\hat{z}})_i \leq \hat{u}_i,\ |(P_{\hat{z}})_i - \hat{z}_i| \leq \|w_i\|_* d_{\|\cdot\|}(\mathcal{X}),\ \hat{z}_i, (P_{\hat{z}})_i \in \mathbb{R}\Big\},\end{aligned} \tag{2.41}$$

which is now in terms of the scalar optimization variables $\hat{z}_i$ and $(P_{\hat{z}})_i$, as we desired. This reformulation makes it tractable to compute the supremum in (2.41) in closed-form, which we now turn to do.

First, consider the case that $c_i \geq 0$. Then we seek to maximize the difference $(P_z)_i - z_i$ subject to the given constraints. Noting that $(P_z)_i \leq q(l,u)$ and $z_i \geq 0$ on the above feasible set, we remark that the objective is upper bounded as $c_i((P_z)_i - z_i) \leq c_i q(l,u)$. Indeed, this upper bound is attained at the feasible point defined by $z_i = \hat{z}_i = (P_{\hat{z}})_i = 0$ and $(P_z)_i = q(l,u)$. Hence, we conclude that for all $i \in \{1,2,\ldots,n_z\}$ such that $c_i \geq 0$, it holds that

$$\Delta\phi_i^\star \leq c_i q(l,u). \tag{2.42}$$

Now consider the case that $c_i < 0$. Then we seek to minimize the difference $(P_z)_i - z_i$ subject to the given constraints. In this case, the optimal objective value depends on the relative sizes of $\hat{u}_i$ and $\|w_i\|_* d_{\|\cdot\|}(\mathcal{X})$. In particular, when $\hat{u}_i \leq \|w_i\|_* d_{\|\cdot\|}(\mathcal{X})$, the constraint $\hat{z}_i \leq \hat{u}_i$ becomes active at optimum, yielding a supremum value of $-c_i u_i$. Alternatively, when $\|w_i\|_* d_{\|\cdot\|}(\mathcal{X}) \leq \hat{u}_i$, the constraint $|(P_{\hat{z}})_i - \hat{z}_i| \leq \|w_i\|_* d_{\|\cdot\|}(\mathcal{X})$ becomes active at optimum, yielding the supremum value of $-c_i\|w_i\|_* d_{\|\cdot\|}(\mathcal{X})$. Therefore, we conclude that for all $i \in \{1,2,\ldots,n_z\}$ such that $c_i < 0$, it holds that

$$\Delta\phi_i^\star \leq -c_i \min\{\hat{u}_i, \|w_i\|_* d_{\|\cdot\|}(\mathcal{X})\}. \tag{2.43}$$

Substituting (2.42) and (2.43) into (2.40) gives the desired bound. □

When the $x$-block $P_x^\star$ of the SDP relaxation stays close to the true solution $x^\star$, the bound (2.39) shows that the worst-case relaxation error scales with the loosest input bound, i.e., the maximum value amongst the limits $|l_k|$ and $|u_k|$. This fact allows us to choose which coordinate to partition along in order to maximally reduce the relaxation bound on the individual parts of the partition. We state our proposed SDP branching scheme next.

**Proposed Branching Scheme**

We now focus on developing a worst-case optimal branching scheme based on the relaxation bound of Theorem 5. Similar to the partitioned LP relaxation, the diameter $d_{\|\cdot\|}(\mathcal{X})$ tends to be small in practical settings, making the terms being summed in (2.39) approximately equal to $\mathrm{ReLU}(c_i)q(l,u)$. We restrict ourselves to this form in order to simplify the subsequent analysis. The objective to optimize therefore takes the form

$$q(l,u)\sum_{i=1}^{n_z}\mathrm{ReLU}(c_i), \tag{2.44}$$

where

$$q(l,u) = \max_{k\in\{1,2,\ldots,n_x\}} \max\{|l_k|, |u_k|\}.$$

Since the design of the partition amounts to choosing input bounds $l$ and $u$ for the input parts, the input bounds serve as our optimization variables in minimizing the above relaxation bound. By restricting the form of our partition to the uniform division motivated in Theorem 4, it follows from the form of $q$ that the best coordinate to partition along is that with the loosest input bound, i.e., along coordinate $i^\star \in \arg\max_{k\in\{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\}$. This observation is formalized below.

**Theorem 6.** *Consider the two-part partitions defined by dividing $\mathcal{X}$ uniformly along the coordinate axes: $\{\mathcal{X}_i^{(1)}, \mathcal{X}_i^{(2)}\}$, with $\mathcal{X}_i^{(1)} = \{x \in \mathcal{X} : l_i^{(1)} \le x \le u_i^{(1)}\}$ and $\mathcal{X}_i^{(2)} = \{x \in \mathcal{X} : l_i^{(2)} \le x \le u_i^{(2)}\}$, where $l_i^{(1)} = l$, $u_i^{(1)} = (u_1, u_2, \dots, u_{i-1}, \frac{1}{2}(l_i + u_i), u_{i+1}, \dots, u_{n_x})$, $l_i^{(2)} = (l_1, l_2, \dots, l_{i-1}, \frac{1}{2}(l_i + u_i), l_{i+1}, \dots, l_{n_x})$, and $u_i^{(2)} = u$, for all $i \in \{1, 2, \dots, n_x\} =: \mathcal{I}$. Let*

$$i^\star \in \underset{k\in\{1,2,\dots,n_x\}}{\arg\max} \max\{|l_k|, |u_k|\}, \tag{2.45}$$

*and assume that $|l_{i^\star}| \neq |u_{i^\star}|$. Then the partition $\{\mathcal{X}_{i^\star}^{(1)}, \mathcal{X}_{i^\star}^{(2)}\}$ is optimal in the sense that the upper bound factor $q(l_{i^\star}^{(j)}, u_{i^\star}^{(j)})$ in (2.44) equals the unpartitioned upper bound $q(l, u)$ on one part $j$ of the partition, is strictly less than $q(l, u)$ on the other part, and $q(l_i^{(j)}, u_i^{(j)}) = q(u, l)$ for both $j \in \{1, 2\}$ for all other $i \notin \arg\max_{k\in\{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\}$.*

*Proof.* First, consider partitioning along coordinate $i \notin \arg\max_{k\in\{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\}$. Then

$$\begin{aligned} q(l_i^{(1)}, u_i^{(1)}) &= \max_{k\in\{1,2,\dots,n_x\}} \max\{|(l_i^{(1)})_k|, |(u_i^{(1)})_k|\} \\ &= \max\left\{|l_1|, \dots, |l_{n_x}|, |u_1|, \dots, \left|\frac{l_i + u_i}{2}\right|, \dots, |u_{n_x}|\right\} = \max\{|l_{i^\star}|, |u_{i^\star}|\} = q(l, u), \end{aligned}$$

since $\left|\frac{l_i+u_i}{2}\right| \le \frac{|l_i|+|u_i|}{2} < \max\{|l_{i^\star}|, |u_{i^\star}|\}$ and $i \neq i^\star$ implies that

$$\max\{|l_{i^\star}|, |u_{i^\star}|\} \in \left\{|l_1|, \dots, |l_{n_x}|, |u_1|, \dots, \left|\frac{l_i + u_i}{2}\right|, \dots, |u_{n_x}|\right\}.$$

In an analogous fashion, it follows that

$$q(l_i^{(2)}, u_i^{(2)}) = q(l, u).$$

Now, consider partitioning along coordinate $i^\star$. Note that either

$$\max_{k\in\{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\} = |l_{i^\star}|, \text{ or } \max_{k\in\{1,2,\dots,n_x\}} \max\{|l_k|, |u_k|\} = |u_{i^\star}|.$$

Suppose that the first case holds true. Then

$$\begin{aligned} q(l_{i^\star}^{(1)}, u_{i^\star}^{(1)}) &= \max_{k\in\{1,2,\dots,n_x\}} \max\{|(l_{i^\star}^{(1)})_k|, |(u_{i^\star}^{(1)})_k|\} \\ &= \max\left\{|l_1|, \dots, |l_{n_x}|, |u_1|, \dots, \left|\frac{l_{i^\star} + u_{i^\star}}{2}\right|, \dots, |u_{n_x}|\right\} = |l_{i^\star}| = q(l, u), \end{aligned}$$

since $|(l_{i^\star} + u_{i^\star})/2| \leq (|l_i^\star| + |u_i^\star|)/2 < |l_i^\star|$ and

$$|l_{i^\star}| \in \max\left\{|l_1|, \ldots, |l_{n_x}|, |u_1|, \ldots, \left|\frac{l_{i^\star} + u_{i^\star}}{2}\right|, \ldots, |u_{n_x}|\right\}.$$

Over the second part of the partition,

$$\begin{aligned} q(l_{i^\star}^{(2)}, u_{i^\star}^{(2)}) &= \max_{k\in\{1,2,\ldots,n_x\}} \max\{|(l_{i^\star}^{(2)})_k|, |(u_{i^\star}^{(2)})_k|\} \\ &= \max\left\{|l_1|, \ldots, \left|\frac{l_{i^\star} + u_{i^\star}}{2}\right|, \ldots, |l_{n_x}|, |u_1|, \ldots, |u_{n_x}|\right\} < |l_{i^\star}| = q(l, u), \end{aligned}$$

since $|(l_{i^\star} + u_{i^\star})/2| < |l_{i^\star}|$ and

$$|l_i^\star| \notin \left\{|l_1|, \ldots, \left|\frac{l_{i^\star} + u_{i^\star}}{2}\right|, \ldots, |l_{n_x}|, |u_1|, \ldots, |u_{n_x}|\right\}$$

since $|l_{i^\star}| \neq |u_{i^\star}|$. In the other case that $\max_{k\in\{1,2,\ldots,n_x\}} \max\{|l_k|, |u_k|\} = |u_{i^\star}|$, it follows via the same argument that $q(l_{i^\star}^{(1)}, u_{i^\star}^{(1)}) < q(l, u)$ and $q(l_{i^\star}^{(2)}, u_{i^\star}^{(2)}) = q(l, u)$. Since partitioning along any other coordinate $i \notin \arg\max_{k\in\{1,2,\ldots,n_x\}} \max\{|l_k|, |u_k|\}$ was shown to yield $q(l_i^{(1)}, u_i^{(1)}) = q(l_i^{(2)}, u_i^{(2)}) = q(u, l)$, we conclude that the coordinate $i^\star$ is optimal in the sense proposed. □

Intuitively, the branching scheme defined in Theorem 6 is optimal because any other uniform partition along a coordinate axis cannot tighten the relaxation error bound (2.44). On the other hand, Theorem 6 guarantees that using the partition coordinate in (2.45) results in a strict tightening of the worst-case relaxation error on at least one part of the partition.

## 2.5 Implementing the Branching Schemes

For the reader's convenience, we give a pseudocode in Algorithm 1 to embed our branching schemes (Theorem 2 and Theorem 6) into an overall branch-and-bound algorithm, which is what we use to compute the simulation results in Section 2.6 below. The pseudocode for the SDP branch-and-bound procedure is the same as in Algorithm 1 albeit with lines 2 and 3 modified to use Theorem 6 and (2.8), respectively. Solving the two subproblems in line 3 can be done independently, and in a parallel fashion, to enhance computational efficiency and scalability.

## 2.6 Numerical Simulations

In this section, we experimentally corroborate the effectiveness of our proposed certification methods. We first perform LP branch-and-bound on benchmark datasets to show that our

**Algorithm 1** LP branch-and-bound procedure for certification

**Input:** $f$, $\mathcal{X}$, $n_{\text{branches}}$

**for** $n = 1, \ldots, n_{\text{branches}}$
  **compute** $\mathcal{X}_{i^\star}^{(1)}, \mathcal{X}_{i^\star}^{(2)}$ according to Theorem 2
  **compute** $\hat{\phi}_{\text{LP}}^\star(\mathcal{X}_{i^\star}^{(1)}), \hat{\phi}_{\text{LP}}^\star(\mathcal{X}_{i^\star}^{(2)})$ according to (2.5)
  **if** $\hat{\phi}_{\text{LP}}^\star(\mathcal{X}_{i^\star}^{(1)}) \geq \hat{\phi}_{\text{LP}}^\star(\mathcal{X}_{i^\star}^{(2)})$
    **assign** $\mathcal{X} \leftarrow \mathcal{X}_{i^\star}^{(1)}$
  **else**
    **assign** $\mathcal{X} \leftarrow \mathcal{X}_{i^\star}^{(2)}$
**if** $\max_{j \in \{1,2\}} \hat{\phi}_{\text{LP}}^\star(\mathcal{X}_{i^\star}^{(j)}) \leq 0$
  **assign** `is_certified` $\leftarrow$ `True`
**else**
  **assign** `is_certified` $\leftarrow$ `False`
**return** `is_certified`

proposed branching scheme beats the current state-of-the-art. We then compute the SDP and branched SDP, and compare to the LP results. Finally, we explore the effectiveness of branching on the LP and SDP as the network grows in size, namely, as the number of inputs and the number of layers independently increase. All simulations are performed on a standard laptop computer using Tensorflow 2.5 in Python 3.9. Training of networks is done using the Adam optimizer [77], and certifications are performed using MOSEK in CVXPY [41].

## LP Results

In this simulation, we consider classification networks trained on three datasets: the Wisconsin breast cancer diagnosis dataset with $(n_x, n_z) = (30, 2)$ [42], the MNIST handwritten digit dataset with $(n_x, n_z) = (784, 10)$ [84], and the CIFAR-10 image classification dataset with $(n_x, n_z) = (3072, 10)$ [80]. The Wisconsin breast cancer dataset is characteristic of a real-world machine learning setting in which robustness guarantees are crucial for safety; a misdiagnosis of breast cancer may result in grave consequences for the patient under concern. Each neural network is composed of an affine layer followed by a ReLU hidden layer followed by another affine layer. For each network, we consider 15 different nominal inputs $\bar{x}$ and corresponding uncertainty sets $\mathcal{X} = \{x \in \mathbb{R}^{n_x} : \|x - \bar{x}\|_\infty \leq \epsilon\}$, where we choose a range of attack radii $\epsilon$. We also perform a sweep over the number of branching steps to use in the branch-and-bound certification scheme. Recall that a negative optimal objective value of the robustness certification problem proves that no perturbation of $\bar{x}$ within $\mathcal{X}$ results in misclassification.

Figure 2.6a, Figure 2.6b, Figure 2.7, and Figure 2.8 display the percent of test inputs certified using the LP relaxation with our branching method and that with the state-of-

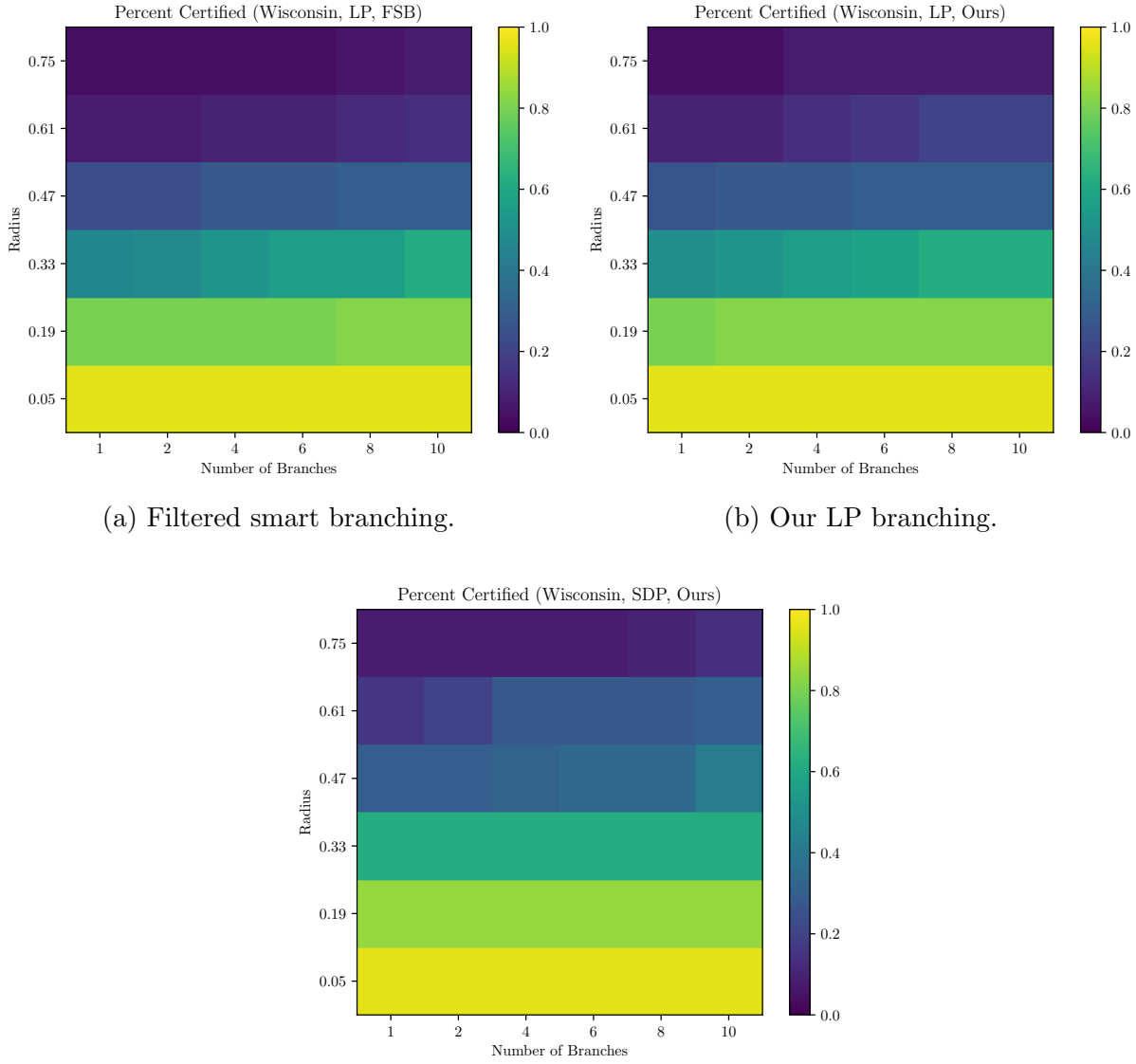
the-art branching method, filtered smart branching (FSB) [39], that was used in the $\alpha, \beta$-CROWN branch-and-bound scheme to win the 2021 and 2022 VNN-COMP certification competitions [143, 18]. We see that our LP branching method substantially outperforms FSB on all three benchmark datasets, attaining higher certification percentages in fewer branching steps and at larger radii than FSB, achieving around 8% (Wisconsin), 17% (MNIST), and 20% (CIFAR-10) increases in a variety of radius-number of branches settings. That is, for a fixed problem setting (attack radius and number of branches in a branch-and-bound scheme), our simulations show that simply replacing the filtered smart branching heuristic with our worst-case optimal branching scheme may result in large increases in certification percentages, and, according to our results, never performs any worse on average.

## SDP Results

In this simulation, we consider the same ReLU neural network trained on the Wisconsin breast cancer dataset as used in our LP results above. We solve the branched SDP using our proposed branching scheme from Theorem 6, where the same test data and simulation parameters are used as with the LP above. The results are shown in Figure 2.6c. In comparing the SDP results to the LP results in Figure 2.6a and Figure 2.6b, we see that our branched SDP achieves better certification percentages compared to the branched LP approaches, up to 12% in some radius-number of branches settings. We remark that Zhang [171] provides theoretical guarantees for the tightness of the SDP relaxation under some technical conditions. However, since we find a strict increase in the certified percentages upon branching on the SDP, we conclude that such conditions for exactness may not be satisfied in general by practical networks, indicating that branching on the SDP may in fact be necessary in settings where high accuracy is the primary concern at hand. As we will see in the next experiment, this certification enhancement via SDP branching becomes even more substantial as the network depth increases. We now move to this experiment, and propose a general rule of thumb for when LP, SDP, and their branch-and-bound variants are best applied based on depth and width of the network.

## Effectiveness as Network Grows

In this section, we perform two experiments to test the effectiveness of branching as the size and structure of the network changes. First, we consider two-layer networks of structure $n_x \times 100 \times 5$, where $n_x$ is the input dimension. For each input size $n_x \in \{5, 10, 20, 40, 80, 100\}$, we generate one network with standard normal random weights, and another network with uniformly distributed weights (where each element is distributed uniformly on the interval $[0, 1]$). The weights are normalized according to Assumption 1. For each network being tested, we compute the LP, branched LP, SDP, and branched SDP relaxations at a fixed nominal input $\bar{x}$ using the input uncertainty set $\mathcal{X} = \{x \in \mathbb{R}^{n_x} : \|x - \bar{x}\|_\infty \leq \epsilon\}$ with $\epsilon = 0.5$. The optimal values, corresponding computation times, and percentage improvements induced by branching are reported in Table 2.2. The effectiveness of branching for the LP remains



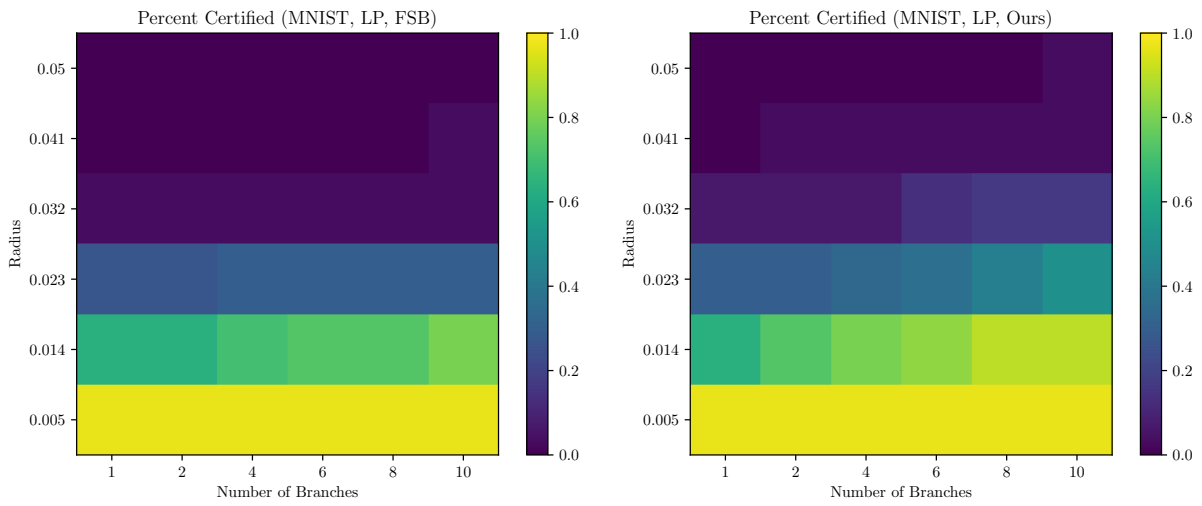

(a) Filtered smart branching.

(b) Our LP branching.

(c) Our SDP branching.

Figure 2.6: Percent certified on Wisconsin breast cancer diagnosis dataset using branching on LP and SDP relaxations.

relatively constant between 5 and 10 percent improvement, whereas the branching appears to lose its efficacy on the SDP as the input size grows. As expected, the two-part partitioned convex relaxations take twice as long to solve as their unpartitioned counterparts. Note that, despite the fact that branching works better for the LP with wide networks, the actual optimal value of the SDP-based certificates are always lower (tighter) than the LP-based ones. This matches what is known in the literature: the SDP is a tighter relaxation technique



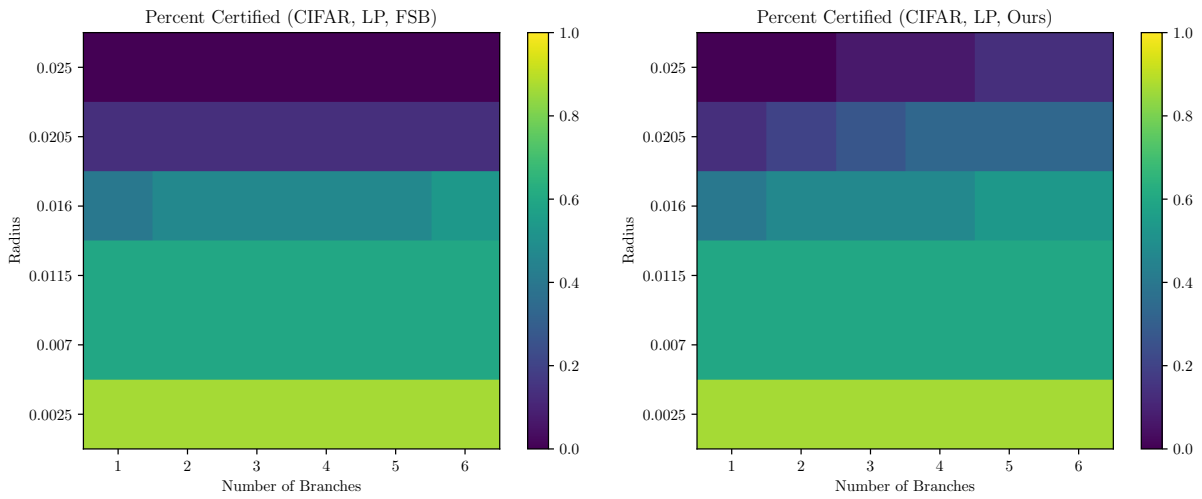

(a) Filtered smart branching.

(b) Our branching.

Figure 2.7: Percent certified on MNIST dataset using branching on LP relaxation.

than the LP [116]. However, the computation times of the SDP and branched SDP quickly increase as the network size increases, whereas the LP and branched LP computation times are seen to slowly increase. All of this suggests the following: in the regime of shallow (i.e., one or two hidden layers) but very wide networks, the branched LP should be used, since the branching remains effective in tightening the relaxation, yet the method is scalable to large networks where the SDP cannot be feasibly applied.

In the second simulation of this section, we analyze the effectiveness of branching as the depth of the network increases. In particular, we consider networks with normal random weights having 5 inputs and 5 outputs, and each intermediate layer having 10 neurons. We run the experiment on networks having 1 through 6 such intermediate layers. Note that when the network has more than one hidden layer, an extra step is needed in order to apply the worst-case optimal LP branching scheme from Theorem 2 since the number of rows $n_1$ of the first layer's weight matrix $W^{[0]}$ (i.e., the rows being partitioned along) may not equal the dimension of the output space $n_z$. The extra step is to generate a surrogate "$c$"-vector of size $n_1 \times 1$ so that Theorem 2 can be applied using this surrogate cost vector. There are a few ways of doing this. One such method is to treat the activation at the first hidden layer, $x^{[1]}$, as the output and determine which coordinate $i \in \mathbb{R}^{n_1}$ of the nominal activation $\bar{x}^{[1]} = \text{ReLU}(W^{[0]}\bar{x})$ is maximal. This means that $i$ would be the class assigned to $\bar{x}$ if the output were after the first hidden layer. Then, we find the second-best coordinate $j \neq i$ so that $\bar{x}_i^{[1]} \geq \bar{x}_j^{[1]} \geq \bar{x}_k^{[1]}$ for all other $k$. Afterwards, the surrogate vector $c$ can be taken as $c = e_j - e_i$, meaning that it serves as a measure of whether the classification after the first hidden layer changes from $i$ to $j$. In the case of our experiment, we choose the above $j \neq i$ randomly for simplicity. Of course, the surrogate vector $c$ is only used to compute the




(a) Filtered smart branching.

(b) Our branching.

Figure 2.8: Percent certified on CIFAR-10 dataset using branching on LP relaxation.

branching coordinate $i^\star$ in Theorem 2, and the full network and original cost vector $c$ are used in the resultant branched LP.

Unlike the branched LP, the SDP branching scheme given in Theorem 6 can directly be applied to deep networks, without the need to use the intermediate steps to compute the partition. We compute the LP, branched LP, SDP, and branched SDP on the networks at hand and report the objective values and computation times in Table 2.3. In this simulation, we see a stark contrast to the results in Table 2.2. Specifically, the percentage improvement induced by branching on the LP reduces quickly to nearly zero percent for networks with 3 or more intermediate 10-neuron hidden layers. Indeed, this is one fundamental drawback behind the LP relaxation: the convex upper envelope is used independently at every neuron, so the relaxation error quickly compounds as the network becomes deeper. On the other hand, the SDP relaxation takes into account the coupling between the layers of the network. This theoretical advantage is demonstrated empirically, as the percentage improvement gained by the branched SDP hovers around 10% even for the deep networks tested here. Moreover, note how the SDP computation time remains relatively close to that of the LP, unlike the rapid increase in computation time seen when increasing the input size. This behavior suggests the following: in the regime of deep but relatively narrow networks, the branched SDP should be used, since the branching is effective in tightening the relaxation, yet the computational cost grows relatively slowly as more layers are added (compared to the case where more inputs are added).

## 2.7 Conclusions

In this chapter, we propose intelligently designed closed-form branching schemes for linear programming (LP) and semidefinite programming (SDP) robustness certification methods of ReLU neural networks. The branching schemes are derived by minimizing the worst-case error induced by the corresponding convex relaxations, which is theoretically justified by showing that minimizing the true relaxation error is NP-hard. The proposed techniques are experimentally substantiated by demonstrating significant reduction in relaxation error on benchmark datasets. Our numerical simulations show that the LP and SDP branching schemes exhibit tradeoffs between different regimes, namely, as the input size and the number of layers are varied. The results conclude that both LP and SDP branching schemes yield a reduction in relaxation error on the order of 10%, with LP best applying to shallow but wide networks, and SDP best applying to deep but narrow networks.

Table 2.2: Varying input size $n_x$ for $n_x \times 100 \times 5$ ReLU network. Optimal values and corresponding computation times reported. B-LP and B-SDP correspond to branched LP and branched SDP, respectively. %-LP and %-SDP represent the percentage tightening of the optimal values obtained from branching.

(a) Normally distributed network weights.

| Input size | LP | B-LP | %-LP | SDP | B-SDP | %-SDP |
|---|---|---|---|---|---|---|
| 5 | 126.93 | 117.92 | **7.10**% | 16.82 | 14.83 | **11.85**% |
| | 0.71 s | 1.46 s | 104.18% | 1.66 s | 3.33 s | 101.33% |
| 10 | 187.57 | 176.19 | **6.07**% | 33.62 | 32.96 | **1.98**% |
| | 0.77 s | 1.36 s | 76.13% | 1.54 s | 3.16 s | 105.57% |
| 20 | 386.49 | 364.53 | **5.68**% | 54.02 | 54.01 | **0.02**% |
| | 0.71 s | 1.42 s | 100.49% | 1.85 s | 4.31 s | 132.94% |
| 40 | 874.70 | 864.56 | **1.16**% | 104.90 | 104.38 | **0.49**% |
| | 1.27 s | 2.68 s | 110.93% | 4.79 s | 9.33 s | 95.01% |
| 80 | 1591.41 | 1496.23 | **5.98**% | 310.37 | 310.31 | **0.02**% |
| | 1.76 s | 2.97 s | 69.00% | 9.81 s | 17.87 s | 82.11% |
| 100 | 2184.94 | 2175.87 | **0.42**% | 383.63 | 383.50 | **0.03**% |
| | 0.78 s | 1.84 s | 136.93% | 5.02 s | 10.52 s | 109.46% |

(b) Uniformly distributed network weights.

| Input size | LP | B-LP | %-LP | SDP | B-SDP | %-SDP |
|---|---|---|---|---|---|---|
| 5 | 11.65 | 10.69 | **8.31**% | 5.95 | 5.74 | **3.44**% |
| | 0.65 s | 1.36 s | 109.54% | 1.39 s | 2.20 s | 58.32% |
| 10 | 34.13 | 34.13 | **0.00**% | 12.61 | 11.92 | **5.47**% |
| | 0.68 s | 1.36 s | 101.35% | 1.32 s | 2.48 s | 87.45% |
| 20 | 83.74 | 83.02 | **0.86**% | 19.20 | 19.00 | **1.06**% |
| | 0.67 s | 1.40 s | 106.88% | 1.31 s | 2.85 s | 118.39% |
| 40 | 141.37 | 133.30 | **5.71**% | 25.89 | 25.69 | **0.74**% |
| | 0.69 s | 1.43 s | 106.67% | 1.63 s | 3.23 s | 97.62% |
| 80 | 260.80 | 242.19 | **7.14**% | 21.86 | 21.68 | **0.84**% |
| | 0.71 s | 1.42 s | 99.25% | 2.82 s | 5.44 s | 92.97% |
| 100 | 400.73 | 387.24 | **3.37**% | 102.87 | 102.35 | **0.51**% |
| | 0.74 s | 1.56 s | 111.10% | 3.33 s | 6.89 s | 106.64% |

Table 2.3: Varying number of hidden layers for a $5 \times 10 \times 10 \times \cdots \times 10 \times 5$ ReLU network with normal random weights. Optimal values and corresponding computation times reported. B-LP and B-SDP correspond to branched LP and branched SDP, respectively. %-LP and %-SDP represent the percentage tightening of the optimal values obtained from branching.

| Layers | LP | B-LP | %-LP | SDP | B-SDP | %-SDP |
|---|---|---|---|---|---|---|
| 1 | 10.16 | 7.03 | **30.79**% | 4.70 | 4.65 | **1.12**% |
| | 0.59 s | 1.21 s | 105.06% | 0.68 s | 1.29 s | 91.17% |
| 2 | 46.29 | 44.89 | **3.03**% | 2.42 | 1.94 | **19.94**% |
| | 0.62 s | 1.21 s | 93.07% | 0.71 s | 1.49 s | 108.81% |
| 3 | 626.96 | 626.96 | **0.00**% | 36.29 | 34.36 | **5.31**% |
| | 0.61 s | 1.29 s | 110.86% | 0.72 s | 1.47 s | 103.32% |
| 4 | 5229.32 | 5229.32 | **0.00**% | 179.79 | 167.34 | **6.93**% |
| | 0.65 s | 1.29 s | 97.47% | 0.99 s | 1.88 s | 89.81% |
| 5 | 37625.91 | 37625.86 | **0.00**% | 628.78 | 561.60 | **10.68**% |
| | 0.69 s | 1.34 s | 94.59% | 1.13 s | 2.04 s | 80.35% |
| 6 | 326743.55 | 326743.34 | **0.00**% | 3245.41 | 3050.69 | **6.00**% |
| | 0.75 s | 1.35 s | 79.44% | 1.19 s | 2.35 s | 98.01% |

# Appendices

## 2.A Proof of Theorem 3

**Theorem 3.** *Consider the partitioned LP relaxation* (2.30) *of the $K$-layer ReLU neural network certification problem. The optimal partitioning problem in the first-layer, as formulated in* (2.31)*, is NP-hard.*

*Proof.* We prove the result by reducing an arbitrary instance of the Min-$\mathcal{K}$-Union problem to an instance of the optimal partitioning problem (2.31). The proof is broken down into steps. In Step 1, we introduce the Min-$\mathcal{K}$-Union problem. We then construct a specific neural network based on the parameters of the Min-$\mathcal{K}$-Union problem in Step 2. In Step 3, we construct the solution to the partitioned LP relaxation for our neural network in the case that the partition is performed along all input coordinates. In Step 4, we construct the solution to the partitioned LP relaxation in the case that only a subset of the input coordinates are partitioned. Finally, in Step 5, we show that the solution to the Min-$\mathcal{K}$-Union problem can be constructed from the solution to the optimal partitioning problem, i.e., by finding the best subset of coordinates to partition along in the fourth step. As a consequence, we show that optimal partitioning is NP-hard.

**Step 1: Arbitrary Min-$\mathcal{K}$-Union Problem.** Suppose that we are given an arbitrary instance of the Min-$\mathcal{K}$-Union problem, i.e., a finite number of finite sets $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n$ and a positive integer $\mathcal{K} \leq n$. Since each set $\mathcal{S}_j$ is finite, the set $\bigcup_{j=1}^{n} \mathcal{S}_j$ is finite with cardinality $m \coloneqq \left|\bigcup_{i=j}^{n} \mathcal{S}_j\right| \in \mathbb{N}$. Therefore, there exists a bijection between the elements of $\bigcup_{j=1}^{n} \mathcal{S}_j$ and the set $\{1, 2, \ldots, m\}$. Hence, without loss of generality, we assume $\mathcal{S}_j \subseteq \mathbb{N}$ for all $j \in \{1, 2, \ldots, n\}$ such that $\bigcup_{j=1}^{n} \mathcal{S}_j = \{1, 2, \ldots, m\}$. In this Min-$\mathcal{K}$-Union problem, the objective is to find $\mathcal{K}$ sets $\mathcal{S}_{j_1}, \mathcal{S}_{j_2}, \ldots, \mathcal{S}_{j_\mathcal{K}}$ among the collection of $n$ given sets such that $\left|\bigcup_{i=1}^{\mathcal{K}} \mathcal{S}_{j_i}\right|$ is minimized over all choices of $\mathcal{K}$ sets. In what follows, we show that the solution to this problem can be computed by solving a particular instance of the optimal partitioning problem (2.31).

**Step 2: Neural Network Construction.** Consider a 3-layer ReLU network, where $x^{[0]}, x^{[1]} \in \mathbb{R}^n$ and $x^{[2]}, x^{[3]} \in \mathbb{R}^m$. Let the weight vector on the output be $c = \mathbf{1}_m$. Take

the input uncertainty set to be $\mathcal{X} = [-1, 1]^n$. Let $W^{[0]} = I_n$ and $W^{[2]} = I_m$. In addition, construct the weight matrix on the first layer to be $W^{[1]} \in \mathbb{R}^{m \times n}$ such that

$$W_{ij}^{[1]} = \begin{cases} 1 & \text{if } i \in \mathcal{S}_j, \\ 0 & \text{otherwise.} \end{cases}$$

We remark that, since all entries of $c = \mathbf{1}_m$, $W^{[0]} = I_n$, $W^{[1]}$, and $W^{[2]} = I_m$ are nonnegative, the optimal value of the unrelaxed certification problem (2.27) is $\phi^\star(\mathcal{X}) = \mathbf{1}_m^\top W^{[1]} \mathbf{1}_n$.

To finish defining the network and its associated LP relaxations, we must specify the preactivation bounds at each layer. Since all weights of the neural network are nonnegative, the largest preactivation at each layer is attained when the input is $x^{[0]} = \mathbf{1}_n$, the elementwise maximum vector in $\mathcal{X}$. The preactivations corresponding to this input are $\hat{z}^{[1]} = \mathbf{1}_n$, $\hat{z}^{[2]} = W^{[1]}\mathbf{1}_n$, and $\hat{z}^{[3]} = W^{[1]}\mathbf{1}_n$. Therefore, setting

$$\begin{aligned} u^{[1]} &= 2\mathbf{1}_n, \\ u^{[2]} &= \frac{3}{2} W^{[1]} \mathbf{1}_n, \\ u^{[3]} &= \frac{5}{4} W^{[1]} \mathbf{1}_n + \frac{1}{8} \mathbf{1}_m, \end{aligned}$$

we obtain valid preactivation upper bounds. Similarly, taking

$$l^{[k]} = -u^{[k]}$$

for all $k \in \{1, 2, 3\}$ defines valid preactivation lower bounds.

**Step 3: Densely Partitioned LP Relaxation.** With the network parameters defined, we now consider the first variant of our partitioned LP relaxation. In particular, we consider the relaxation (2.30) where all coordinates of the first layer are partitioned. We denote by $\bar{\phi}(\mathcal{X})$ the optimal objective value of this problem:

$$\bar{\phi}(\mathcal{X}) = \max_{j' \in \{1, 2, \ldots, 2^n\}} \hat{\phi}_{\mathrm{LP}}^\star(\mathcal{X}^{(j')}), \tag{2.46}$$

where $\hat{\phi}_{\mathrm{LP}}^\star(\mathcal{X}^{(j')})$ denotes the optimal objective value of

$$\begin{array}{lll} \text{maximize} & c^\top x^{[3]} & \\ \text{subject to} & x^{[0]} \in \mathcal{X}^{(j')}, & \\ & x^{[k+1]} \geq W^{[k]} x^{[k]}, & k \in \{0, 1, 2\}, \\ & x^{[k+1]} \geq 0, & k \in \{0, 1, 2\}, \\ & x^{[k+1]} \leq u^{[k+1]} \odot (W^{[k]} x^{[k]} - l^{[k+1]}) \oslash (u^{[k+1]} - l^{[k+1]}), & k \in \{0, 1, 2\}, \\ & x^{[1]} = \mathrm{ReLU}(W^{[0]} x^{[0]}). & \end{array} \tag{2.47}$$

This problem serves as a baseline; this is the tightest LP relaxation of the certification problem among all those with partitioning along the input coordinates. (Recall that the final equality constraint in (2.47) is linear over the restricted feasible set $\mathcal{X}^{(j')}$.)

We will now show that an optimal solution $\bar{x} = (\bar{x}^{[0]}, \bar{x}^{[1]}, \bar{x}^{[2]}, \bar{x}^{[3]})$ of the partitioned LP defined by (2.46) and (2.47) can be taken to satisfy

$$\bar{x}^{[3]} = \frac{5}{4} W^{[1]} \mathbf{1}_n + \frac{1}{16} \mathbf{1}_m.$$

To see this, note that since all weights of the network and optimization (2.46) are nonnegative, the optimal activations $\bar{x}$ will be as large as possible in all coordinates and at all layers. Therefore, since the input is constrained to $\mathcal{X} = [-1, 1]^n$, the optimal input for (2.46) is $\bar{x}^{[0]} = \mathbf{1}_n$. Since the ReLU constraint in (2.47) is exact, this implies that the optimal activation over this part at the first layer is

$$\bar{x}^{[1]} = \mathrm{ReLU}(W^{[0]} \bar{x}^{[0]}) = \mathrm{ReLU}(\mathbf{1}_n) = \mathbf{1}_n.$$

Now, for the second layer, the activation attains its upper bound. Since $u^{[2]} = -l^{[2]} = \frac{3}{2} W^{[1]} \mathbf{1}_n$, this implies that

$$\begin{aligned}
\bar{x}^{[2]} &= u^{[2]} \odot (W^{[1]} \bar{x}^{[1]} - l^{[2]}) \oslash (u^{[2]} - l^{[2]}) \\
&= u^{[2]} \odot (W^{[1]} \bar{x}^{[1]} + u^{[2]}) \oslash (2u^{[2]}) \\
&= \frac{1}{2} (W^{[1]} \bar{x}^{[1]} + u^{[2]}) \\
&= \frac{1}{2} \left( W^{[1]} \mathbf{1}_n + \frac{3}{2} W^{[1]} \mathbf{1}_n \right) \\
&= \frac{5}{4} W^{[1]} \mathbf{1}_n.
\end{aligned}$$

Similarly, for the third layer, we find that the optimal activation attains its upper bound as well. Since $u^{[3]} = -l^{[3]} = \frac{5}{4} W^{[1]} \mathbf{1}_n + \frac{1}{8} \mathbf{1}_m$ and $W^{[2]} = I_m$ this gives that

$$\begin{aligned}
\bar{x}^{[3]} &= u^{[3]} \odot (W^{[2]} \bar{x}^{[2]} - l^{[3]}) \oslash (u^{[3]} - l^{[3]}) \\
&= u^{[3]} \odot (\bar{x}^{[2]} + u^{[3]}) \oslash (2u^{[3]}) \\
&= \frac{1}{2} (\bar{x}^{[2]} + u^{[3]}) \\
&= \frac{1}{2} \left( \frac{5}{4} W^{[1]} \mathbf{1}_n + \frac{5}{4} W^{[1]} \mathbf{1}_n + \frac{1}{8} \mathbf{1}_m \right) \\
&= \frac{5}{4} W^{[1]} \mathbf{1}_n + \frac{1}{16} \mathbf{1}_m,
\end{aligned}$$

as claimed in (3). It is easily verified that $\bar{x}$ as computed above satisfies all constraints of the problem (2.47) over the part of the partition containing $\bar{x}^{[0]} = \mathbf{1}_n$.

**Step 4: Sparsely Partitioned LP Relaxation.** We now introduce the second variant of the partitioned LP relaxation. In particular, let $\mathcal{J}_p \subseteq \{1, 2, \ldots, n\}$ be an index set such

that $|\mathcal{J}_p| = n_p = n - \mathcal{K}$. Denote the complement of $\mathcal{J}_p$ by $\mathcal{J}_p^c = \{1, 2, \ldots, n\} \setminus \mathcal{J}_p$. We consider the partitioned LP defined in (2.30), which partitions along each coordinate in the index set $\mathcal{J}_p$. The optimal value of this problem is denoted by $\phi^\star_{\mathcal{J}_p}(\mathcal{X})$, and we denote an optimal solution by $\hat{x} = (\hat{x}^{[0]}, \hat{x}^{[1]}, \hat{x}^{[2]}, \hat{x}^{[3]})$. We will compute $\hat{x}$ in three steps.

**Step 4.1: Upper Bounding the Solution.** We start by upper bounding the final layer activation of the solution. In particular, we claim that the optimal solution $\hat{x}$ satisfies

$$\hat{x}^{[3]} \leq t \coloneqq u^{[3]} - \frac{1}{16}\mathbf{1}_{\mathcal{I}^c}. \tag{2.48}$$

where $\mathcal{I} = \bigcup_{j \in \mathcal{J}_p^c} S_j \subseteq \{1, 2, \ldots, m\}$ and $\mathcal{I}^c = \{1, 2, \ldots, m\} \setminus \mathcal{I}$. Since $\bar{x}^{[3]} = u^{[3]} - \frac{1}{16}\mathbf{1}_m$, the bound (2.48) is equivalent to

$$\hat{x}_i^{[3]} \leq t_i = \begin{cases} u_i^{[3]} & \text{if } i \in \mathcal{I}, \\ \bar{x}_i^{[3]} & \text{if } i \in \mathcal{I}^c, \end{cases} \tag{2.49}$$

for all $i \in \{1, 2, \ldots, m\}$. We now prove the element-wise representation of the bound, (2.49).

First, by the feasibility of $\hat{x}$ and the definitions of $u^{[3]}, l^{[3]}$, it must hold for all $i \in \{1, 2, \ldots, m\}$ that

$$\hat{x}_i^{[3]} \leq \frac{u_i^{[3]}}{u_i^{[3]} - l_i^{[3]}}(w_i^{[2]\top}\hat{x}^{[2]} - l_i^{[3]}) = \frac{1}{2}(w_i^{[2]\top}\hat{x}^{[2]} + u_i^{[3]}),$$

and also that

$$\hat{x}_i^{[3]} \geq w_i^{[2]\top}\hat{x}^{[2]}.$$

Combining these inequalities, we find that $\hat{x}_i^{[3]} \leq \frac{1}{2}(\hat{x}_i^{[3]} + u_i^{[3]})$, or, equivalently, that

$$\hat{x}_i^{[3]} \leq u_i^{[3]}.$$

This bound holds for all $i \in \{1, 2, \ldots, m\}$, and therefore it also holds for $i \in \mathcal{I}$. This proves the first case in the bound (2.49).

We now prove the second case of the claimed upper bound. For this case, suppose $i \notin \mathcal{I}$. Then $i \notin S_j$ for all $j \in \mathcal{J}_p^c$, which implies that

$$W_{ij}^{[1]} = 0 \text{ for all } j \in \mathcal{J}_p^c,$$

by the definition of $W^{[1]}$. Therefore,

$$w_i^{[1]\top}\hat{x}^{[1]} = \sum_{j=1}^{n} W_{ij}^{[1]}\hat{x}_j^{[1]} = \sum_{j \in \mathcal{J}_p^c} W_{ij}^{[1]}\hat{x}_j^{[1]} + \sum_{j \in \mathcal{J}_p} W_{ij}^{[1]}\hat{x}_j^{[1]} = \sum_{j \in \mathcal{J}_p} W_{ij}^{[1]}\hat{x}_j^{[1]}.$$

Now, note that for $j \in \mathcal{J}_p$, the $j$th coordinate of the input is being partitioned, and therefore the optimal solution must satisfy

$$\hat{x}_j^{[1]} = \mathrm{ReLU}(w_j^{[0]\top}\hat{x}^{[0]}) = \mathrm{ReLU}(e_j^\top \hat{x}^{[0]}) = \mathrm{ReLU}(\hat{x}_j^{[0]}) \leq 1,$$

since $\hat{x}_j^{[0]} \in [-1, 1]$. Therefore,

$$w_i^{[1]\top}\hat{x}^{[1]} \leq \sum_{j \in \mathcal{J}_p^c} W_{ij}^{[1]} \leq \sum_{j=1}^{n} W_{ij}^{[1]} = w_i^{[1]\top}\mathbf{1}_n.$$

It follows from the feasibility of $\hat{x}$ and the definitions of $u^{[2]}, l^{[2]}$ that

$$\begin{aligned}\hat{x}_i^{[2]} &\leq \frac{u_i^{[2]}}{u_i^{[2]} - l_i^{[2]}}(w_i^{[1]\top}\hat{x}^{[1]} - l_i^{[2]}) = \frac{1}{2}(w_i^{[1]\top}\hat{x}^{[1]} + u_i^{[2]}) \\ &\leq \frac{1}{2}\left(w_i^{[1]\top}\mathbf{1}_n + \frac{3}{2}w_i^{[1]\top}\mathbf{1}_n\right) = \frac{5}{4}w_i^{[1]\top}\mathbf{1}_n = \bar{x}_i^{[2]},\end{aligned}$$

where $\bar{x}$ is the solution computed for the densely partitioned LP relaxation in Step 3. Therefore, we conclude that for all $i \notin \mathcal{I}$, it holds that

$$\hat{x}_i^{[3]} \leq \frac{u_i^{[3]}}{u_i^{[3]} - l_i^{[3]}}(w_i^{[2]\top}\hat{x}^{[2]} - l_i^{[3]}) \leq \frac{u_i^{[3]}}{u_i^{[3]} - l_i^{[3]}}(w_i^{[2]\top}\bar{x}^{[2]} - l_i^{[3]}) = \bar{x}_i^{[3]},$$

by our previous construction of $\bar{x}^{[3]}$. Thus, we have proven the second case in (2.49) holds. Hence, the claimed bound (2.48) holds.

**Step 4.2: Feasibility of Upper Bound.** Let us define $x = (x^{[0]}, x^{[1]}, x^{[2]}, x^{[3]})$, a point in $\mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^m$, by

$$x^{[0]} = \mathbf{1}_n, \quad x^{[1]} = \mathbf{1}_{\mathcal{J}_p} + \frac{5}{4}\mathbf{1}_{\mathcal{J}_p^c}, \quad x^{[2]} = u^{[3]} - \frac{1}{8}\mathbf{1}_{\mathcal{I}^c}, \quad x^{[3]} = u^{[3]} - \frac{1}{16}\mathbf{1}_{\mathcal{I}^c}.$$

Note that $x^{[3]}$ equals the upper bound $t = (t_1, t_2, \ldots, t_m)$. We now show that $x$ is feasible for the partitioned LP defined by (2.29) and (2.30).

First, the input uncertainty constraint is satisfied, since $x^{[0]} = \mathbf{1}_n \in \mathcal{X}^{(j')} \subseteq \mathcal{X}$ for some part $\mathcal{X}^{(j')}$. Next, the relaxed ReLU constraints at the first layer are satisfied, since

$$x^{[1]} = \mathbf{1}_{\mathcal{J}_p} + \frac{5}{4}\mathbf{1}_{\mathcal{J}_p^c} \geq 0, \qquad \text{(Layer 1 lower bound.)}$$

$$x^{[1]} - W^{[0]}x^{[0]} = \frac{1}{4}\mathbf{1}_{\mathcal{J}_p^c} \geq 0, \qquad \text{(Layer 1 lower bound.)}$$

$$x^{[1]} - u^{[1]} \odot (W^{[0]}x^{[0]} - l^{[1]}) \oslash (u^{[1]} - l^{[1]}) = -\frac{1}{2}\mathbf{1}_{\mathcal{J}_p} - \frac{1}{4}\mathbf{1}_{\mathcal{J}_p^c} \leq 0. \qquad \text{(Layer 1 upper bound.)}$$

The relaxed ReLU constraints are also satisfied in the second layer, since

$$x^{[2]} = \frac{5}{4}W^{[1]}\mathbf{1}_n + \frac{1}{8}\mathbf{1}_{\mathcal{I}} \geq 0, \qquad \text{(Layer 2 lower bound.)}$$

$$x^{[2]} - W^{[1]}x^{[1]} = \frac{1}{4}W^{[1]}\mathbf{1}_{\mathcal{J}_p} + \frac{1}{8}\mathbf{1}_{\mathcal{I}} \geq 0, \qquad \text{(Layer 2 lower bound.)}$$

$$x^{[2]} - u^{[2]} \odot (W^{[1]}x^{[1]} - l^{[2]}) \oslash (u^{[2]} - l^{[2]}) = \frac{1}{8}(\mathbf{1}_{\mathcal{I}} - W^{[1]}\mathbf{1}_{\mathcal{J}_p^c}) \leq 0. \qquad \text{(Layer 2 upper bound.)}$$

The final inequality above follows from the fact that either $(\mathbf{1}_{\mathcal{I}})_i = 0$ or $(\mathbf{1}_{\mathcal{I}})_i = 1$. For coordinates $i$ such that $(\mathbf{1}_{\mathcal{I}})_i = 0$, the inequality obviously holds. For coordinates $i$ such that $(\mathbf{1}_{\mathcal{I}})_i = 1$, we know that $i \in \mathcal{I}$, implying that $i \in \mathcal{S}_j$ for some $j \in \mathcal{J}_p^c$. This in turn implies that $W_{ij}^{[1]} = 1$ for some $j \in \mathcal{J}_p^c$, and therefore $w_i^{[1]\top}\mathbf{1}_{\mathcal{J}_p^c} = \sum_{j \in \mathcal{J}_p^c} W_{ij}^{[1]} \geq 1 = (\mathbf{1}_{\mathcal{I}})_i$.

Continuing to check feasibility of $x$, the relaxed ReLU constraints in the final layer are also satisfied:

$$x^{[3]} = \frac{5}{4}W^{[1]}\mathbf{1}_n + \frac{1}{16}\mathbf{1}_m + \frac{1}{16}\mathbf{1}_{\mathcal{I}} \geq 0, \qquad \text{(Layer 3 lower bound.)}$$

$$x^{[3]} - W^{[2]}x^{[2]} = \frac{1}{16}\mathbf{1}_{\mathcal{I}^c} \geq 0, \qquad \text{(Layer 3 lower bound.)}$$

$$x^{[3]} - u^{[3]} \odot (W^{[2]}x^{[2]} - l^{[3]}) \oslash (u^{[3]} - l^{[3]}) = 0 \leq 0. \qquad \text{(Layer 3 upper bound.)}$$

Hence, the relaxed ReLU constraints are satisfied at all layers. The only remaining constraints to verify are the exact ReLU constraints for the partitioned input indices $\mathcal{J}_p$. Indeed, for all $j \in \mathcal{J}_p$, we have that

$$x_j^{[1]} - \mathrm{ReLU}(W^{[0]}x^{[0]})_j = (\mathbf{1}_{\mathcal{J}_p})_j + \frac{5}{4}(\mathbf{1}_{\mathcal{J}_p^c})_j - \mathrm{ReLU}(\mathbf{1}_n)_j = 1 + 0 - 1 = 0.$$

Hence, the ReLU equality constraint is satisfied for all input coordinates in $\mathcal{J}_p$. Therefore, our proposed point $x$ is feasible for (2.30).

**Step 4.3: Solution to Sparsely Partitioned LP.** As shown in the previous step, the proposed point $x = (x^{[0]}, x^{[1]}, x^{[2]}, x^{[3]})$ is feasible for the partitioned LP defined by (2.29) and (2.30). Recall from the upper bound (2.48) that our solution $\hat{x} = (\hat{x}^{[0]}, \hat{x}^{[1]}, \hat{x}^{[2]}, \hat{x}^{[3]})$ satisfies $\hat{x}^{[3]} \leq t$. The objective value of the feasible point $x$ gives that

$$c^\top x^{[3]} = \sum_{i=1}^m x^{[3]} = \sum_{i=1}^m t_i \geq \sum_{i=1}^m \hat{x}_i^{[3]} = \phi^\star_{\mathcal{J}_p}(\mathcal{X}).$$

Since $\phi^\star_{\mathcal{J}_p}(\mathcal{X})$ is the maximum value of the objective for all feasible points, it must be that $c^\top x^{[3]} = \phi^\star_{\mathcal{J}_p}(\mathcal{X})$. Hence, the point $x$ is an optimal solution to (2.30). Therefore, we can write the final activation of our optimal solution $\hat{x}$ to (2.30) as

$$\hat{x}^{[3]} = x^{[3]} = t = u^{[3]} - \frac{1}{16}\mathbf{1}_{\mathcal{I}^c}. \tag{2.50}$$

**Step 5: Min-$\mathcal{K}$-Union from Optimal Partition.** With the solutions constructed in Step 3 and Step 4, we compute the difference in the objective values between the two partitioned LP relaxations:

$$\begin{aligned}\phi^\star_{\mathcal{J}_p}(\mathcal{X}) - \bar{\phi}(\mathcal{X}) &= c^\top \hat{x}^{[3]} - c^\top \bar{x}^{[3]} = c^\top \left( u^{[3]} - \frac{1}{16}\mathbf{1}_{\mathcal{I}^c} - \frac{5}{4}W^{[1]}\mathbf{1}_n - \frac{1}{16}\mathbf{1}_m \right) \\ &= c^\top \left( \frac{5}{4}W^{[1]}\mathbf{1}_n + \frac{1}{8}\mathbf{1}_m - \frac{1}{16}\mathbf{1}_{\mathcal{I}^c} - \frac{5}{4}W^{[1]}\mathbf{1}_n - \frac{1}{16}\mathbf{1}_m \right) = \frac{1}{16}c^\top(\mathbf{1}_m - \mathbf{1}_{\mathcal{I}^c}) \\ &= \frac{1}{16}c^\top \mathbf{1}_{\mathcal{I}} = \frac{1}{16}\sum_{i\in\mathcal{I}} 1 = \frac{1}{16}\left|\mathcal{I}\right| = \frac{1}{16}\left| \bigcup_{j\in\mathcal{J}_p^c} S_j \right|.\end{aligned}$$

Therefore,

$$\left| \bigcup_{j\in\mathcal{J}_p^c} \mathcal{S}_j \right| = 16\left( \phi^\star_{\mathcal{J}_p}(\mathcal{X}) - \bar{\phi}(\mathcal{X}) \right), \tag{2.51}$$

which holds for all partition index sets $\mathcal{J}_p \subseteq \{1, 2, \ldots, n\}$ such that $|\mathcal{J}_p| = n_p = n - \mathcal{K}$.

Now, let $\mathcal{J}_p^\star$ be an optimal partition, i.e., a solution to (2.31) with our specified neural network parameters. Then, by (2.51), we have $\left|\bigcup_{j\in(\mathcal{J}_p^\star)^c} \mathcal{S}_j\right| = 16\left(\phi^\star_{\mathcal{J}_p^\star}(\mathcal{X}) - \bar{\phi}(\mathcal{X})\right) \leq 16\left(\phi^\star_{\mathcal{J}_p}(\mathcal{X}) - \bar{\phi}(\mathcal{X})\right) = \left|\bigcup_{j\in\mathcal{J}_p^c} \mathcal{S}_j\right|$ for all $\mathcal{J}_p$ with $|\mathcal{J}_p| = n_p$. Since this holds for all $\mathcal{J}_p^c$ with $|\mathcal{J}_p^c| = n - n_p = \mathcal{K}$, this shows that the set $(\mathcal{J}_p^\star)^c$ is an optimal solution to the Min-$\mathcal{K}$-Union problem specified at the beginning of the proof. Now, suppose the optimal partitioning problem in (2.31) could be solved for $\mathcal{J}_p^\star$ in polynomial time. Then the optimal solution $(\mathcal{J}_p^\star)^c$ to the Min-$\mathcal{K}$-Union problem is also computable in polynomial time. Since this holds for an arbitrary instance of the Min-$\mathcal{K}$-Union problem, this implies that the Min-$\mathcal{K}$-Union problem is polynomially solvable in general, which is a contradiction. Therefore, the problem (2.31) is NP-hard in general. □

# Chapter 3

# Globally Optimal Certification of Min-Max Affine Models

Although branch-and-bound approaches to solving the robustness certification problem are capable of finding a global solution, doing so may require solving an exponential number of subproblems in general. Instead, it is natural to wonder whether the certification problem may be solved to global optimality in polynomial time. In this chapter, we obtain *tight* robustness certificates over convex attack sets for min-max representations of ReLU neural networks by developing a convex reformulation of the nonconvex certification problem. This is done by "lifting" the problem to an infinite-dimensional optimization over probability measures, leveraging recent results in distributionally robust optimization to solve for an optimal discrete distribution, and proving that solutions of the original nonconvex problem are generated by the discrete distribution under mild boundedness, nonredundancy, and Slater conditions. As a consequence, optimal (worst-case) attacks against the model may be solved for *exactly*. This contrasts prior state-of-the-art that either requires expensive branch-and-bound schemes or loose relaxation techniques. Numerical simulations on robust control and MNIST image classification examples highlight the benefits of our approach.

This chapter is based on the following previously published work:

[6] Brendon G. Anderson, Samuel Pfrommer, and Somayeh Sojoudi, "Tight certified robustness via min-max representations of ReLU neural networks," *IEEE Conference on Decision and Control (CDC)*, 2023.

## 3.1 Introduction

As discussed in Chapter 2, certifying a neural network's robustness generally amounts to solving an intractable nonconvex optimization problem [73], and hence various techniques have been proposed to bound the problem as a tractable surrogate. In this chapter, we utilize

an alternative representation of ReLU neural networks as a means to efficiently compute tight robustness certificates using convex optimization (and hence in polynomial time). As a consequence, we are able to exactly compute optimal (worst-case) attacks, which is generally not possible using the popular local search-based attack methods such as projected gradient descent [95] and the Carlini-Wagner attack [28].

## Related Works

### Robustness Certification

As introduced previously, certifying the robustness of a model amounts to solving the non-convex optimization

$$\inf_{x \in X} g(x),$$

where $X$ is a set of possible inputs or attacks (i.e., the "threat model"), and $g(x)$ is either the model output at an input $x$, or some linear transformation of the model output (e.g., a classifier's margin between two classes).

Convex relaxations work by optimizing over a convex outer-approximation of the set $g(X)$ of possible outputs. Popular relaxations involve linear bounding and programming [150, 168], and semidefinite programming [116, 50], which constitutes a line of increasingly accurate yet computationally complex relaxations. Convex relaxation-based certificates remain loose in general, and their looseness has been shown to increase with model size [5].

The Lipschitz constant of a model provides a certified bound on how much the model output may change given some change in its input. Thus, bounds on the Lipschitz constant can yield efficient robustness certificates [51]. A number of works are devoted to computing Lipschitz bounds, but it has proven difficult to obtain tight enough bounds to grant meaningful certificates [51, 148, 139, 72].

Mixed-integer programming and branch-and-bound have also been applied to robustness certification for ReLU neural networks [135, 4, 143] (see also Chapter 2). In contrast to convex relaxations and Lipschitz bounding, these methods are capable of obtaining tight certificates if they are run to convergence, but this incurs exponential computational complexity, preventing them from scaling to practically-sized models [143]. Some methods allow for early termination of their optimizations to yield more efficient, yet loose certificates [143].

In this chapter, we take a different approach from the above works by changing our mathematical representation of ReLU neural networks.

### Representations of ReLU Neural Networks

ReLU neural networks are defined by compositions $g = \mathcal{A}_L \circ \sigma \circ \cdots \circ \sigma \circ \mathcal{A}_1$ with affine functions $\mathcal{A}_l$ and elementwise activation functions $\sigma = \mathrm{ReLU}\colon x \mapsto \max\{0, x\}$. The most prevalent alternative representation of such a model is as a piecewise linear function, i.e., a finite polyhedral partition of $\mathbb{R}^d$ with associated affine functions that agree with $g$ on each polyhedron [101, 11]. Another representation is as a rational function when working with

tropical algebra, where addition $\oplus$ and multiplication $\otimes$ are defined by $x \oplus y = \max\{x, y\}$ and $x \oplus y = x + y$ [170]. Finally, min-max representations—discussed in Section 3.2—have recently been introduced, where $g$ is expressed as the pointwise minimum of pointwise maxima of affine functions. These works restrict their focus to showcasing the impressive approximation capabilities of ReLU models and their alternative representations.

## Contributions

The primary contributions of this chapter are as follows:

1. We show that ReLU neural networks admit min-max representations and hence such representations are universal function approximators.
2. By lifting the certification to an infinite-dimensional problem over probability measures, we prove that, under mild boundedness, nonredundancy, and Slater conditions, *exact* solutions to the original nonconvex problem are efficiently obtained for min-max representations via reduction to a tractable finite-dimensional convex optimization problem.
3. Numerical simulations on robust control and MNIST image classification examples demonstrate the effectiveness of our approach.

To the best of our knowledge, our work is the first to grant *tight* robustness certificates in polynomial time amongst those considering general ReLU neural networks and their alternative representations.[1]

## Organization

The remainder of this chapter is organized as follows. In Section 3.2, we introduce and analyze the min-max representation of ReLU neural networks. We develop our tight robustness certificates in Section 3.3. Numerical simulations illustrating the effectiveness of our approach are given in Section 3.4, and concluding remarks are made in Section 3.5.

## Notations

The sets of natural, real, nonnegative real, and extended real numbers are denoted by $\mathbb{N}$, $\mathbb{R}$, $\mathbb{R}_+$, and $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ respectively. Throughout this chapter, we let $\mathcal{I}, \mathcal{J}, \mathcal{K} \subseteq \mathbb{N}$ denote index sets $\{1, \ldots, m\}$, $\{1, \ldots, n\}$, and $\{1, \ldots, p\}$, respectively. The cardinality, convex hull, conic hull, and relative interior of a subset $X$ of $\mathbb{R}^d$ are denoted by $|X|$, $\mathrm{conv}(X)$, $\mathrm{cone}(X)$, and $\mathrm{ri}(X)$, respectively. Furthermore, we define $\mathcal{B}(X)$ to be the Borel $\sigma$-algebra on $X$. We denote the set of probability measures on the measurable space $(X, \mathcal{B}(X))$ by $\mathcal{P}(X)$.

[1] See Awasthi, Dutta, and Vijayaraghavan [13] for a polynomial time solution to the special case of 2-layer ReLU models.

For $x \in \mathbb{R}^d$, the Dirac measure centered at $x$ is denoted by $\delta_x$, which we recall is the probability measure defined by $\delta_x(A) = 0$ if $x \notin A$ and $\delta_x(A) = 1$ if $x \in A$ for all $A \in \mathcal{B}(X)$. The set of all Dirac measures with center in $X$ is defined to be $\mathcal{D}(X) = \{\mu \in \mathcal{P}(X) : \mu = \delta_x \text{ for some } x \in X\}$. The set of continuous functions from $\mathbb{R}^d$ into $\mathbb{R}$ is denoted by $C(\mathbb{R}^d, \mathbb{R})$. The effective domain of a function $f\colon \mathbb{R}^d \to \overline{\mathbb{R}}$ is defined to be the set $\operatorname{dom}(f) = \{x \in \mathbb{R}^d : f(x) < \infty\}$. If $f$ is Borel measurable and $\mu$ is a probability measure on $(X, \mathcal{B}(X))$, then we denote the expected value of $f$ with respect to $\mu$ by $\mathbb{E}_{x\sim\mu} f(x) = \int_X f(x) d\mu(x)$. If $f$ is convex, the subdifferential of $f$ at $x$ is denoted by $\partial f(x)$. Throughout, we let $\|\cdot\|$ denote an arbitrary norm on $\mathbb{R}^d$, and we denote its dual norm by $\|\cdot\|_*$.

## 3.2 Min-Max Affine Functions

In this section, we formally define min-max affine functions, discuss works related to these functions, and show that every ReLU neural network admits such a representation.

**Definition 3.** A function $g\colon \mathbb{R}^d \to \mathbb{R}$ is a *min-max affine function* if there exist index sets $\mathcal{I}, \mathcal{J}_1, \ldots, \mathcal{J}_{|\mathcal{I}|} \subseteq \mathbb{N}$ and associated $a_{ij} \in \mathbb{R}^d$, $b_{ij} \in \mathbb{R}$ such that $g(x) = \min_{i\in\mathcal{I}} \max_{j\in\mathcal{J}_i} (a_{ij}^\top x + b_{ij})$ for all $x \in \mathbb{R}^d$. In this case, the function $x \mapsto \min_{i\in\mathcal{I}} \max_{j\in\mathcal{J}_i} (a_{ij}^\top x + b_{ij})$ is called the *min-max representation* of $g$.

The class of all min-max affine functions on $\mathbb{R}^d$ is denoted by $\mathcal{G}$. Notice that $g \in \mathcal{G}$ is the pointwise minimum of $m = |\mathcal{I}|$ convex functions $g_i\colon x \mapsto \max_{j\in\mathcal{J}_i} (a_{ij}^\top x + b_{ij})$, and is therefore nonconvex in general. Without loss of generality, we henceforth assume that, for every $g \in \mathcal{G}$, there exists some $\mathcal{J} \subseteq \mathbb{N}$ with $n = |\mathcal{J}|$ such that the min-max representation of $g$ satisfies $\mathcal{J}_i = \mathcal{J}$ for all $i \in \mathcal{I}$.[2]

**Related works on min-max affine functions.** In the mathematics literature, min-max affine functions are also termed lattice polynomials [99]. The work Velasco-Forero and Angulo [138] shows that piecewise linear activation functions can be written in min-max affine form, and that neural networks learned with such representations perform highly in image classification tasks. The works Bagirov [14] and Bagirov and Ugon [15] study the theoretical and algorithmic aspects of training min-max affine functions to separate data, and show that separating $\{x_1, \ldots, x_p\} \subseteq \mathbb{R}^d$ from $\{y_1, \ldots, y_q\} \subseteq \mathbb{R}^d$ requires no more that $pq$ affine components. The authors of Rister and Rubin [118] use min-max representations of neural networks to characterize the training optimization landscape. The conversion of ReLU neural networks into min-max affine form is characterized in Chen, Klivans, and Meka [31, Theorem 4.15]. An algorithm for nonlinear system identification using min-max affine functions is developed

[2]This is without loss of generality, since the value $g(x)$ does not change upon appending affine global underestimators of the convex function $g_i\colon x \mapsto \max_{j\in\mathcal{J}_i} (a_{ij}^\top x + b_{ij})$ to the set of affine components $x \mapsto a_{ij}^\top x + b_{ij}$ of $g$. In other words, $\min_{i\in\mathcal{I}} \max_{j\in\mathcal{J}_i} (a_{ij}^\top x + b_{ij}) = \min_{i\in\mathcal{I}} \max_{j\in\mathcal{J}} (a_{ij}^\top x + b_{ij})$ if one defines $\mathcal{J} = \{1, \ldots, n\}$ with $n = \max_{i\in\mathcal{I}} |\mathcal{J}_i|$ and $a_{ij} = v_i$, $b_{ij} = g_i(0)$ for $j \in \mathcal{J} \setminus \mathcal{J}_i$, for all $i \in \mathcal{I}$, where $v_i \in \mathbb{R}^d$ is a subgradient of $g_i$ at 0 (which exists by Rockafellar [119, Theorem 23.4]).

in Wang and Narendra [144]. Finally, min-max affine functions have been used as consistent statistical estimators, termed "Riesz estimators," in the mathematical economics literature [2]. To the best of our knowledge, our work is the first to exploit min-max representations for purposes of robustness certification.

We now proceed with analyzing the representation power min-max affine functions. Let $\mathcal{F}$ be the class of all ReLU neural networks on $\mathbb{R}^d$. The following theorem shows that every ReLU neural network can be represented as a min-max affine function, and therefore min-max affine functions are universal function approximators.

**Theorem 7.** *For every $f \in \mathcal{F}$, there exist $\mathcal{I}, \mathcal{J} \subseteq \mathbb{N}$ and $(a_{ij}, b_{ij}) \in \mathbb{R}^d \times \mathbb{R}$ for $i \in \mathcal{I},\ j \in \mathcal{J}$ such that*

$$f(x) = \min_{i \in \mathcal{I}} \max_{j \in \mathcal{J}} (a_{ij}^\top x + b_{ij}) \text{ for all } x \in \mathbb{R}^d. \tag{3.1}$$

*Hence, the class $\mathcal{G}$ of min-max affine functions is dense in $C(\mathbb{R}^d, \mathbb{R})$ with respect to the topology of uniform convergence on compact sets.*

*Proof.* Every $f \in \mathcal{F}$ is piecewise affine, i.e., there is a finite collection $\mathcal{Q}$ of closed subsets of $\mathbb{R}^d$ such that $\mathbb{R}^d = \bigcup_{Q \in \mathcal{Q}} Q$ and $f$ is affine on every $Q \in \mathcal{Q}$. Hence, by Ovchinnikov [109, Theorem 4.1], there exist $\mathcal{I}, \mathcal{J} \subseteq \mathbb{N}$ and $(a_{ij}, b_{ij}) \in \mathbb{R}^d \times \mathbb{R}$ for $i \in \mathcal{I},\ j \in \mathcal{J}$ such that (3.1) holds. Thus, since $\mathcal{F} \subseteq \mathcal{G}$ and $\mathcal{F}$ is dense in $C(\mathbb{R}^d, \mathbb{R})$ with respect to the topology of uniform convergence on compact sets [115, Theorem 3.1], it holds that $\mathcal{G}$ is dense in $C(\mathbb{R}^d, \mathbb{R})$ in the same sense. □

The next result shows that min-max affine functions admit tight closed-form Lipschitz bounds, unlike composition-based representations of ReLU models.

**Proposition 6.** *Let the function $g$ take the form $g(x) = \min_{i \in \mathcal{I}} \max_{j \in \mathcal{J}} (a_{ij}^\top x + b_{ij})$, and assume that for all $i \in \mathcal{I},\ j \in \mathcal{J}$ there exists an open set $U \subseteq \mathbb{R}^d$ such that $g(x) = a_{ij}^\top x + b_{ij}$ for all $x \in U$. Then $g$ is Lipschitz continuous with constant $\mathrm{Lip}_{\|\cdot\|}(g) = \max_{(i,j) \in \mathcal{I} \times \mathcal{J}} \|a_{ij}\|_*$.*

*Proof.* It is straightforward to see that, for $f_1, f_2 \colon \mathbb{R}^d \to \mathbb{R}$ with respective Lipschitz constants $L_1, L_2$, the functions $x \mapsto \min\{f_1(x), f_2(x)\}$ and $x \mapsto \max\{f_1(x), f_2(x)\}$ both have Lipschitz constants bounded above by $\max\{L_1, L_2\}$. Therefore, since $|a_{ij}^\top (x - x')| \leq \|a_{ij}\|_* \|x - x'\|$ for all $i, j, x, x'$, we have that $\mathrm{Lip}_{\|\cdot\|}(g) \leq \max_{(i,j) \in \mathcal{I} \times \mathcal{J}} \|a_{ij}\|_*$.

To show that the inequality holds with equality, let $(i, j) \in \arg\max_{(i',j') \in \mathcal{I} \times \mathcal{J}} \|a_{i'j'}\|_*$. By assumption, there exists some open $U \subseteq \mathbb{R}^d$ such that $g(x) = a_{ij}^\top x + b_{ij}$ for all $x \in U$. Fix such an $x \in U$, and let $x' \in \mathbb{R}^d$ be such that $\|x'\| = 1$ and $a_{ij}^\top x' = \|a_{ij}\|_*$ (this is possible since the dual norm supremum is attained). The set $U$ contains an open $\|\cdot\|$-norm ball of some radius $r > 0$ centered at $x$. Hence, there exists $C > 0$ such that $\|x'/C\| < r$, and thus $x + x'/C \in U$. Therefore, $g(x + x'/C) - g(x) = a_{ij}^\top x'/C = \|a_{ij}\|_*/C = \|a_{ij}\|_* \|(x + x'/C) - x\|$ since $\|x'/C\| = 1/C$. This shows that indeed $\mathrm{Lip}_{\|\cdot\|} = \|a_{ij}\|_* = \max_{(i',j') \in \mathcal{I} \times \mathcal{J}} \|a_{i'j'}\|_*$. □

Although global Lipschitz bounds may grant certificates of a model's robustness to input perturbations [51], the resulting robustness certificate is generally loose, even if the bound

on the global Lipschitz constant is tight. In the next section, we show how to *tightly* solve the robustness certification problem for min-max affine representations of models.

## 3.3 Theoretical Robustness Certificates

In this section, we develop our theoretical robustness certificates. Consider a model $g\colon \mathbb{R}^d \to \overline{\mathbb{R}}$, which may, for example, represent the output of a scalar-valued controller or the confidence of a binary classifier $f\colon \mathbb{R}^d \to \{1, 2\}$ defined by $f(x) = 1$ if $g(x) \geq 0$ and $f(x) = 2$ if $g(x) < 0$. We consider the asymmetric robustness setting introduced in Pfrommer* et al. [113], where nonnegative outputs $g(x) \geq 0$ are "sensitive" and we seek to certify that no input within some convex uncertainty set $X \subseteq \mathbb{R}^d$ causes the output to leave the sensitive operating regime. This asymmetric setting accurately models realistic adversarial situations. For example, an adversary may seek some imperceptible attack $x \in X = \{x' \in \mathbb{R}^d : \|x' - \overline{x}\| \leq \epsilon\}$ to cause a vehicle's image classifier to predict "no pedestrian" ($g(x) < 0$) when the nominal image $\overline{x}$ has a pedestrian in view (the sensitive regime; $g(x) \geq 0$), but not the other way around. We leave as future work the extension to vector-valued models.

Formally, the certification problem we seek to solve in this chapter is written

$$p^\star := \inf_{x \in X} g(x).$$

The model $g$ is robust if and only if $p^\star \geq 0$. On the other hand, if $x^\star$ solves $p^\star$, then $x^\star$ is an optimal (worst-case) attack in $X$, and it is successful if $p^\star < 0$.

The problem $p^\star$ is nonconvex due to the nonconvexity of $g$. When $g$ is a min-max affine function, a naive reformulation of $p^\star$ yields that

$$p^\star = \inf_{(x,i,t) \in X \times \mathcal{I} \times \mathbb{R}} \{t : a_{ij}^\top x + b_{ij} \leq t \text{ for all } j \in \mathcal{J}\},$$

which removes the nonconvexity in $x$ but is inefficient to solve in general due to the integer variable $i$. Alternatively, one may attempt to directly reformulate the problem into a convex one by minimizing the convex envelope of $g$. Although the resulting problem coincides with our convex reformulation $\underline{c}$ (introduced below) on the relative interior of the direct reformulation's feasible set, it is difficult to obtain regularity conditions under which the direct reformulation holds with respect to its entire feasible set.

We propose an alternative approach to solving $p^\star$ that consists of three steps: 1) lift the problem to an optimization over probability measures, 2) leverage results and regularity conditions in distributionally robust optimization to make a finite-dimensional reduction of the problem, and 3) reformulate and solve the finite-dimensional reduction.

## Lifting the Problem

We lift the problem to an optimization over probability measures by noting that $g(x) = \int_X g(x')d\delta_x(x') = \mathbb{E}_{x'\sim\delta_x} g(x')$ whenever $x \in X$:

$$p^\star = \inf_{\delta_x \in \mathcal{D}(X)} \mathbb{E}_{x'\sim\delta_x} g(x').$$

With this reformulation, the optimization objective is linear in the variable $\delta_x$, but the feasible set $\mathcal{D}(X)$ is nonconvex, making the problem intractable as written. Therefore, we consider relaxing the problem to an optimization over all probability measures:

$$p' \coloneqq \inf_{\mu \in \mathcal{P}(X)} \mathbb{E}_{x'\sim\mu} g(x').$$

The problem $p'$ is convex, but infinite-dimensional. We start by showing that the relaxation is exact:

**Proposition 7.** *It holds that $p' = p^\star$.*

*Proof.* Since $\mathcal{D}(X) \subseteq \mathcal{P}(X)$, it holds that $p' \leq p^\star$. Now, let $\mu \in \mathcal{P}(X)$. Then, since $p^\star \leq g(x')$ for all $x' \in X$, it holds that

$$p^\star = \int_X p^\star d\mu(x') \leq \int_X g(x')d\mu(x') = \mathbb{E}_{x'\sim\mu} g(x').$$

Since $\mu \in \mathcal{P}(X)$ is arbitrary, we conclude that $p^\star \leq \inf_{\mu\in\mathcal{P}(X)} \mathbb{E}_{x'\sim\mu} g(x') = p'$. Hence, $p' = p^\star$. □

Next, we show that solutions of the nonconvex problem $p^\star$ are generated by discrete solutions of the relaxation $p'$.

**Proposition 8.** *If $\mu^\star = \sum_{i\in\mathcal{I}} \lambda_i \delta_{x_i}$ is a discrete probability measure that solves $p'$, then $x^\star \coloneqq x_i$ solves $p^\star$ for all $i \in \mathcal{I}$ such that $\lambda_i > 0$.*

*Proof.* Let $i^\star \in \arg\min_{i\in\mathcal{I}} g(x_i)$. Since $\lambda_i \geq 0$ for all $i$, $\sum_{i\in\mathcal{I}} \lambda_i = 1$, and $g(x_{i^\star}) \leq g(x_i)$ for all $i$, it holds that

$$p^\star \leq g(x_{i^\star}) = \sum_{i\in\mathcal{I}} \lambda_i g(x_{i^\star}) \leq \sum_{i\in\mathcal{I}} \lambda_i g(x_i) = \mathbb{E}_{x'\sim\mu^\star} g(x') = p',$$

so $x_{i^\star}$ solves $p^\star$ by Proposition 7. If $x_{i'}$ does not solve $p^\star$ for some $i' \in \mathcal{I}$ such that $\lambda_{i'} > 0$, then $p^\star = g(x_{i^\star}) < g(x_{i'})$, implying that $\sum_{i\in\mathcal{I}} \lambda_i g(x_{i^\star}) < \sum_{i\in\mathcal{I}} \lambda_i g(x_i)$ and hence that $p^\star < p'$, which contradicts Proposition 7. □

The above results show that we may solve the problem $p^\star$ of interest by solving $p'$ for a discrete optimal distribution. The remainder of this section is dedicated to this approach.

## Finite-Dimensional Reduction

To make our finite-dimensional reduction, we recall the definitions of conjugate and perspective functions.

**Definition 4.** The *conjugate* of a function $f\colon \mathbb{R}^d \to \overline{\mathbb{R}}$ is the function $f^*\colon \mathbb{R}^d \to \overline{\mathbb{R}}$ defined by

$$f^*(y) = \sup_{x \in \operatorname{dom}(f)} (y^\top x - f(x)).$$

We write $f^{**}$ to denote the biconjugate $(f^*)^*$.

**Definition 5.** The *perspective* of a proper, closed, and convex function $f\colon \mathbb{R}^d \to \overline{\mathbb{R}}$ is the function $\mathscr{P}_f\colon \mathbb{R}^d \times \mathbb{R}_+ \to \overline{\mathbb{R}}$ defined by

$$\mathscr{P}_f(x,t) = \begin{cases} tf(x/t) & \text{if } t > 0, \\ \sup_{y \in \operatorname{dom}(f^*)} y^\top x & \text{if } t = 0. \end{cases}$$

Recall that the perspective $\mathscr{P}_f$ of a convex function $f$ is also convex, and that the conjugate $f^*$ is convex even when $f$ is nonconvex [24].

Throughout the remainder of the chapter, we fix $g$ and $X$ to be min-max affine and convex, respectively, via the following structural assumptions:

**Assumption 2.** It holds that $g \in \mathcal{G}$, taking the form $g(x) = \min_{i \in \mathcal{I}} g_i(x)$ with $g_i(x) = \max_{j \in \mathcal{J}} (a_{ij}^\top x + b_{ij})$.

**Assumption 3.** The set $X$ takes the form $X = \{x \in \mathbb{R}^d : c_k(x) \leq 0,\ k \in \mathcal{K}\}$ with $c_k\colon \mathbb{R}^d \to \overline{\mathbb{R}}$ a proper, closed, and convex function for all $k \in \mathcal{K}$.

We now make the reduction by introducing two finite-dimensional convex optimization problems:

$$\begin{aligned} \underline{c} \coloneqq \underset{\substack{\lambda_i, \eta_i \in \mathbb{R} \\ x_i \in \mathbb{R}^d}}{\text{minimize}} \quad & \sum_{i \in \mathcal{I}} \eta_i \\ \text{subject to} \quad & \mathscr{P}_{c_k}(x_i, \lambda_i) \leq 0,\ i \in \mathcal{I},\ k \in \mathcal{K}, \\ & \mathscr{P}_{g_i}(x_i, \lambda_i) \leq \eta_i,\ i \in \mathcal{I}, \\ & \sum_{i \in \mathcal{I}} \lambda_i = 1,\ \lambda \geq 0. \end{aligned}$$

$$\begin{aligned} \overline{c} \coloneqq \underset{\substack{\alpha, \beta_{ik} \in \mathbb{R} \\ y_i, z_{ik} \in \mathbb{R}^d}}{\text{maximize}} \quad & -\alpha \\ \text{subject to} \quad & g_i^*(y_i) + \sum_{k \in \mathcal{K}} \mathscr{P}_{c_k^*}(z_{ik}, \beta_{ik}) \leq \alpha,\ i \in \mathcal{I}, \\ & y_i + \sum_{k \in \mathcal{K}} z_{ik} = 0,\ i \in \mathcal{I}, \\ & \beta_{ik} \geq 0,\ i \in \mathcal{I},\ k \in \mathcal{K}. \end{aligned}$$

Intuitively, $\underline{c}$ is minimizing a sort of "average" of the components $g_i$ at a finite number of points $x_i$ with weights given by the probability vector $\lambda$, and $\overline{c}$ is its dual. We now leverage recent results in distributionally robust optimization to show that the finite reductions $\underline{c}, \overline{c}$ allow us to solve the infinite-dimensional problem $p'$ under mild assumptions.

**Definition 6.** Let $f_0, f_1, \ldots, f_m$ and $h_1, \ldots, h_n$ be extended real-valued functions defined on $\mathbb{R}^d$. The optimization problem $p = \inf\{f_0(x) : f_1(x) \leq 0, \ldots, f_m(x) \leq 0,\ h_1(x) = 0, \ldots, h_n(x) = 0,\ x \in \mathbb{R}^d\}$ *admits a Slater point* if there exists $x \in \bigcap_{i=0}^m \operatorname{ri}(\operatorname{dom}(f_i)) \cap \bigcap_{j=1}^n \operatorname{ri}(\operatorname{dom}(h_j))$ such that $f_i(x) \leq 0$ and $h_j(x) = 0$ for all $i$ and all $j$, and such that $f_i(x) < 0$ for all $i \neq 0$ such that $f_i$ is nonlinear.

**Assumption 4.** The set $X$ is bounded and the optimization problem $\overline{c}$ admits a Slater point.

The above boundedness assumption on $X$ is standard in the adversarial robustness literature. The Slater condition may be verified by solving $\overline{c}$ with a small number $\epsilon > 0$ added to all of the nonlinear inequality constraints; replace $f_i(x) \leq 0$ with $f_i(x) + \epsilon \leq 0$ for all nonlinear constraint functions $f_i$.

**Theorem 8.** *If Assumption 4 holds, then $\underline{c} = p' = \overline{c}$, and the discrete probability distribution $\sum_{i \in \mathcal{I}: \lambda_i^\star \neq 0} \lambda_i^\star \delta_{x_i^\star / \lambda_i^\star}$ solves $p'$ for all solutions $(\eta^\star, \lambda^\star, x^\star)$ to $\underline{c}$.*

*Proof.* Since $X$ is defined by a finite intersection of 0-sublevel sets of proper, closed, and convex functions (Assumption 3), and since every $g_i \colon x \mapsto \max_{j \in \mathcal{J}}(a_{ij}^\top x + b_{ij})$ is a proper, closed, and convex function, the result follows from Zhen, Kuhn, and Wiesemann [173, Theorem 12(ii)]. □

Theorem 8 together with our Proposition 7 and Proposition 8 show that we are able to *exactly* compute an optimal attack solving the nonconvex problem $p^\star$ by solving the convex optimizations $\underline{c}, \overline{c}$.

## Reformulating and Solving the Finite Reduction

In order to solve $\underline{c}, \overline{c}$, we must derive the appropriate conjugates and perspectives. In this subsection, we do so for the common cases where $X$ is defined in terms of norm balls or polyhedra. We will also see that computing the conjugate $g_i^*$ is highly nontrivial, and as a result we turn to tractably reformulating the constraint involving $g_i^*$ using duality theory.

**Proposition 9.** *The perspective of $g_i \colon x \mapsto \max_{j \in \mathcal{J}}(a_{ij}^\top x + b_{ij})$ is given by $\mathscr{P}_{g_i}(x, t) = \max_{j \in \mathcal{J}}(a_{ij}^\top x + b_{ij} t)$ for all $(x, t) \in \mathbb{R}^d \times \mathbb{R}_+$.*

*Proof.* Let $x \in \mathbb{R}^d$. If $t > 0$, then

$$\mathscr{P}_{g_i}(x, t) = t g_i(x/t) = t \max_{j \in \mathcal{J}}(a_{ij}^\top x/t + b_{ij}) = \max_{j \in \mathcal{J}}(a_{ij}^\top x + b_{ij} t).$$

If $t = 0$, then

$$\begin{aligned}\mathscr{P}_{g_i}(x,t) &= \liminf_{(x',t')\to(x,0)} \mathscr{P}_{g_i}(x',t') \\ &= \liminf_{(x',t')\to(x,0)} \max_{j\in\mathcal{J}}(a_{ij}^\top x' + b_{ij}t') \\ &= \max_{j\in\mathcal{J}} a_{ij}^\top x \\ &= \max_{j\in\mathcal{J}}(a_{ij}^\top x + b_{ij}t),\end{aligned}$$

where the first equality comes from Theorem 13.3 and Corollary 8.5.2 in Rockafellar [119] and the third equality comes from the continuity of $(x,t) \mapsto \max_{j\in\mathcal{J}}(a_{ij}^\top x + b_{ij}t)$. □

**Proposition 10.** *The perspective of* $c_k\colon x \mapsto \|x-\overline{x}\|-\epsilon$ *is given by* $\mathscr{P}_{c_k}(x,t) = \|x-t\overline{x}\|-\epsilon t$ *for all* $(x,t) \in \mathbb{R}^d \times \mathbb{R}_+$.

*Proof.* Following the same reasoning as in the proof of Proposition 9, we find that $\mathscr{P}_{c_k}(x,t) = t(\|x/t-\overline{x}\|-\epsilon) = \|x-t\overline{x}\|-\epsilon t$ for $t > 0$ and $\mathscr{P}_{c_k}(x,t) = \liminf_{(x',t')\to(x,0)}(\|x'-t'\overline{x}\|-\epsilon t') = \|x\| = \|x-t\overline{x}\|-\epsilon t$ for $t = 0$. □

**Proposition 11.** *The conjugate of* $c_k\colon x \mapsto \|x-\overline{x}\|-\epsilon$ *is given for all* $z \in \mathbb{R}^d$ *by*

$$c_k^*(z) = \begin{cases} z^\top\overline{x} + \epsilon & \text{if } \|z\|_* \le 1, \\ \infty & \text{if } \|z\|_* > 1. \end{cases}$$

*Proof.* Let $z \in \mathbb{R}^d$ be such that $\|z\|_* \le 1$. Then

$$\sup_{x\in\mathbb{R}^d : x\neq\overline{x}} \frac{z^\top(x-\overline{x})}{\|x-\overline{x}\|} = \sup_{x'\in\mathbb{R}^d : \|x'\|\le 1} z^\top x' = \|z\|_* \le 1,$$

so $z^\top(x-\overline{x}) - \|x-\overline{x}\| \le 0$ for all $x \neq \overline{x}$. Also, $z^\top(x-\overline{x}) - \|x-\overline{x}\| = 0$ for $x = \overline{x}$, and therefore $\sup_{x\in\mathbb{R}^d}(z^\top(x-\overline{x}) - \|x-\overline{x}\|) = 0$, indicating that

$$c_k^*(z) = \sup_{x\in\mathbb{R}^d}(z^\top(x-\overline{x}) - \|x-\overline{x}\|) + z^\top\overline{x} + \epsilon = z^\top\overline{x} + \epsilon.$$

On the other hand, let $z \in \mathbb{R}^d$ be such that $\|z\|_* > 1$. Then there exists $x' \in \mathbb{R}^d \setminus \{0\}$ such that $\frac{z^\top x'}{\|x'\|} > 1$, implying that $z^\top x' - \|x'\| > 0$, and hence

$$\begin{aligned}c_k^*(z) &\ge z^\top(\overline{x} + \alpha x') - \|\alpha x'\| + \epsilon \\ &= \alpha(z^\top x' - \|x'\|) + z^\top\overline{x} + \epsilon \\ &\to \infty\end{aligned}$$

as $\alpha \to \infty$. Thus, $c_k^*(z) = \infty$. □

**Proposition 12.** *The perspective of the conjugate of* $c_k \colon x \mapsto \|x - \overline{x}\| - \epsilon$ *is given for all* $(z,t) \in \mathbb{R}^d \times \mathbb{R}_+$ *by*

$$\mathscr{P}_{c_k^*}(z,t) = \begin{cases} z^\top \overline{x} + \epsilon t & \text{if } \|z\|_* \leq t, \\ \infty & \text{if } \|z\|_* > t. \end{cases}$$

*Proof.* Let $t > 0$. If $z \in \mathbb{R}^d$ is such that $\|z\|_* \leq t$, then $\|z/t\|_* \leq 1$, so $\mathscr{P}_{c_k^*}(z,t) = tc_k^*(z/t) = t((z/t)^\top \overline{x} + \epsilon) = z^\top \overline{x} + \epsilon t$. If $\|z\|_* > t$, then $\|z/t\|_* > 1$, so $\mathscr{P}_{c_k^*}(z,t) = tc_k^*(z/t) = \infty$.

On the other hand, let $t = 0$. Then

$$\mathscr{P}_{c_k^*}(z,t) = \sup_{x \in \mathrm{dom}(c_k^{**})} z^\top x = \sup_{x \in \mathbb{R}^d} z^\top x = \begin{cases} 0 & \text{if } z = 0, \\ \infty & \text{if } z \neq 0, \end{cases}$$

since $c_k^{**} = c_k$ which has domain $\mathbb{R}^d$, as $c_k$ is proper, closed, and convex [119, Theorem 12.2]. Since, when $t = 0$, the condition $z = 0$ is equivalent to $\|z\|_* \leq t$ and the condition $z \neq 0$ is equivalent to $\|z\|_* > t$, the proof is complete. □

We also provide the conjugates and perspectives for polyhedral $X$:

**Proposition 13.** *Let* $c_k \colon x \mapsto \psi_k^\top x + \omega_k$ *for some* $\psi_k \in \mathbb{R}^d$ *and some* $\omega_k \in \mathbb{R}$. *Then the following all hold:*

1. $\mathscr{P}_{c_k}(x,t) = \psi_k^\top x + \omega_k t$,
2. $c_k^*(z) = \begin{cases} -\omega_k & \text{if } z = \psi_k, \\ \infty & \text{if } z \neq \psi_k, \end{cases}$
3. *and* $\mathscr{P}_{c_k^*}(z,t) = \begin{cases} -\omega_k t & \text{if } z = t\psi_k, \\ \infty & \text{if } z \neq t\psi_k. \end{cases}$

The proof of Proposition 13 follows from a straightforward application of the definitions of conjugate and perspective, and is hence omitted for brevity.

The conjugate $g_i^*$ is all that remains to compute. However, although computing $g_i^*$ in closed form for the univariate ($d = 1$) function $g_i \colon x \mapsto \max_{j \in \mathcal{J}}(a_{ij}x + b_{ij})$ can be straightforward, generalizing the formula to higher-dimensional settings is nontrivial. In theory, it is possible to express $g_i^*$ for $d > 1$ in closed form via Rockafellar [119, Theorem 19.2]. However, this requires solving a vertex enumeration problem, i.e., determining finite sets $V, R \subseteq \mathbb{R}^d \times \mathbb{R}$ such that the polyhedron $\mathrm{epi}(g_i^*) \coloneqq \{(x,t) \in \mathbb{R}^d \times \mathbb{R} : a_{ij}^\top x + b_{ij} \leq t \text{ for all } j \in \mathcal{J}\}$ equals $\mathrm{conv}(P) + \mathrm{cone}(R)$. The vertex enumeration problem is NP-hard in general [75]. See the Minkowski-Weyl theorem [119, Theorem 19.1] for the theory on such representations of polyhedra. In Theorem 9 that follows, we instead take a duality-based robust optimization approach to tractably deal with the conjugate $g_i^*$ in a direct manner.

**Lemma 5.** *It holds that* $\mathrm{dom}(g_i^*) = \mathrm{conv}\{a_{ij} : j \in \mathcal{J}\}$.

*Proof.* Let $y \in \operatorname{conv}\{a_{ij} : j \in \mathcal{J}\}$. Then $y = \sum_{j\in\mathcal{J}} \theta_j a_{ij}$ for some $\theta \in \mathbb{R}^n$ such that $\theta \geq 0$ and $\sum_{j\in\mathcal{J}} \theta_j = 1$. Hence, for all $x \in \mathbb{R}^d$, we find that

$$\begin{aligned} y^\top x - \max_{j\in\mathcal{J}}(a_{ij}^\top x + b_{ij}) &= \sum_{j\in\mathcal{J}} \theta_j a_{ij}^\top x - \max_{j\in\mathcal{J}}(a_{ij}^\top x + b_{ij}) \\ &= \sum_{j\in\mathcal{J}} \theta_j (a_{ij}^\top x + b_{ij}) - \max_{j\in\mathcal{J}}(a_{ij}^\top x + b_{ij}) - \sum_{j\in\mathcal{J}} \theta_j b_{ij} \\ &\leq \sum_{j\in\mathcal{J}} \theta_j \max_{l\in\mathcal{J}}(a_{il}^\top x + b_{il}) - \max_{j\in\mathcal{J}}(a_{ij}^\top x + b_{ij}) - \sum_{j\in\mathcal{J}} \theta_j b_{ij} \\ &= -\sum_{j\in\mathcal{J}} \theta_j b_{ij}, \end{aligned}$$

and thus $g_i^*(y) \leq -\sum_{j\in\mathcal{J}} \theta_j b_{ij} < \infty$, so $y \in \operatorname{dom}(g_i^*)$.

On the other hand, let $y \in \operatorname{dom}(g_i^*)$, so that $g_i^*(y) < \infty$. An epigraphic reformulation of $g_i^*(y)$ yields that $\infty > g_i^*(y) = \sup_{x\in\mathbb{R}^d}(y^\top x - \max_{j\in\mathcal{J}}(a_{ij}^\top x + b_{ij})) = \sup_{(x,t)\in\mathbb{R}^d\times\mathbb{R}}\{y^\top x - t : a_{ij}^\top x + b_{ij} \leq t \text{ for all } j \in \mathcal{J}\}$. This reformulation is a linear program with a finite optimal value, and hence by Bertsekas [22, Proposition 3.1.3], the reformulation is attained by some $(x, t) \in \mathbb{R}^d \times \mathbb{R}$, and since it must be the case that $t = a_{ij}^\top x + b_{ij}$ for some $j \in \mathcal{J}$ at this point $(x, t)$, we conclude that this $x$ solves the supremum defining $g_i^*(y)$ in its original form (i.e., pre-epigraphic reformulation). Therefore, by the first-order optimality condition for unconstrained convex optimization [119, Theorem 23.2], it holds that $0 \in \partial h_i(x)$, where $h_i \colon \mathbb{R}^d \to \mathbb{R}$ is the convex function defined by $h_i(x) = \max_{j\in\mathcal{J}}(a_{ij}^\top x + b_{ij}) - y^\top x$. Using the rules for subdifferentials of pointwise maxima and sums of proper convex functions [22, Proposition B.22],[119, Theorem 23.8], we have that $\partial h_i(x) = \operatorname{conv}\left(\bigcup_{j\in\mathcal{A}(x)}\{a_{ij}\}\right) + \{-y\}$, where $\mathcal{A}(x)$ denotes the set of active indices at $x$: $\mathcal{A}(x) = \{j \in \mathcal{J} : a_{ij}^\top x + b_{ij} = \max_{l\in\mathcal{J}}(a_{il}^\top x + b_{il})\}$. Since $0 \in \partial h_i(x)$, this yields that $y \in \operatorname{conv}\left(\bigcup_{j\in\mathcal{A}(x)}\{a_{ij}\}\right) \subseteq \operatorname{conv}\{a_{ij} : j \in \mathcal{J}\}$. This completes the proof. □

**Assumption 5.** The functions $g_i$ are nonredundant in the sense that for all $j \in \mathcal{J}$ there exists $x \in \mathbb{R}^d$ such that $g_i(x) = a_{ij}^\top x + b_{ij}$.

It is easy to see that nonredundancy of $g_i$ is efficiently verified by solving the linear (feasibility) programs $\inf\{0 : (a_{il} - a_{ij})^\top x + (b_{il} - b_{ij}) \leq 0 \text{ for all } l \in \mathcal{J},\ x \in \mathbb{R}^d\}$ for all $j \in \mathcal{J}$. Removing the affine components of $g_i$ with infeasible programs ensures that Assumption 5 holds and does not change the model's predictions.

**Theorem 9.** *Suppose that Assumption 5 holds, and let $h \colon \Gamma \to \mathbb{R}$ be an arbitrary real-valued function defined on some nonempty set $\Gamma$. Then, for all $y \in \mathbb{R}^d$ and all $\gamma \in \Gamma$, it holds that $g_i^*(y) \leq h(\gamma)$ if and only if, for all $j \in \mathcal{J}$, there exists $\nu_{ij} \in \mathbb{R}^n$ such that the following all hold:*

1. *$y = \sum_{j\in\mathcal{J}} \theta_j a_{ij}$ for some $\theta \in \mathbb{R}^n$ such that $\theta \geq 0$ and $\sum_{j\in\mathcal{J}} \theta_j = 1$,*

2. $\nu_{ij} \geq 0$,

3. $y - a_{ij} + \sum_{l \in \mathcal{J}} (\nu_{ij})_l (a_{ij} - a_{il}) = 0$,

4. *and* $-b_{ij} + \sum_{l \in \mathcal{J}} (\nu_{ij})_l (b_{ij} - b_{il}) \leq h(\gamma)$.

*Proof.* Let $y \in \mathbb{R}^d$ and $\gamma \in \Gamma$. If $y \neq \sum_{j \in \mathcal{J}} \theta_j a_{ij}$ for all $\theta \in \mathbb{R}^n$ such that $\theta \geq 0$ and $\sum_{j \in \mathcal{J}} \theta_j = 1$, then $y \notin \operatorname{conv}\{a_{ij} : j \in \mathcal{J}\}$ and hence $y \notin \operatorname{dom}(g_i^*)$ by Lemma 5. In this case, $g_i^*(y) = \infty > h(\gamma)$ since $h$ is real-valued. Therefore, the first condition enumerated in the theorem is necessary for $g_i^*(y) \leq h(\gamma)$.

Going forward, assume that $y = \sum_{j \in \mathcal{J}} \theta_j a_{ij}$ for some $\theta \in \mathbb{R}^n$ such that $\theta \geq 0$ and $\sum_{j \in \mathcal{J}} \theta_j = 1$. Hence, $g_i^*(y) < \infty$. Breaking up the conjugate's supremum into $n$ suprema over the affine components of $g_i$ yields

$$\begin{aligned} g_i^*(y) &= \sup_{x \in \mathbb{R}^d} (y^\top x - \max_{j \in \mathcal{J}} (a_{ij}^\top x + b_{ij})) \\ &= \max_{j \in \mathcal{J}} \sup_{x \in \mathbb{R}^d} \{(y - a_{ij})^\top x - b_{ij} : (a_{il} - a_{ij})^\top x + (b_{il} - b_{ij}) \leq 0 \text{ for all } l \in \mathcal{J}\}. \end{aligned}$$

Denote the inner suprema by

$$\mathfrak{p}_{ij} \coloneqq \sup_{x \in \mathbb{R}^d} \{(y - a_{ij})^\top x - b_{ij} : (a_{il} - a_{ij})^\top x + (b_{il} - b_{ij}) \leq 0 \text{ for all } l \in \mathcal{J}\}.$$

Since, by Assumption 5, for all $j \in \mathcal{J}$ there exists $x \in \mathbb{R}^d$ such that $\max_{l \in \mathcal{J}} (a_{il}^\top x + b_{il}) = g_i(x) = a_{ij}^\top x + b_{ij}$, it holds that $\{x \in \mathbb{R}^d : a_{ij}^\top x + b_{ij} \geq a_{il}^\top x + b_{il} \text{ for all } l \in \mathcal{J}\} \neq \emptyset$ for all $j \in \mathcal{J}$, implying that every $\mathfrak{p}_{ij}$ is feasible, i.e., $\mathfrak{p}_{ij} > -\infty$. Furthermore, since $g_i^*(y) < \infty$, it must be the case that $\mathfrak{p}_{ij} < \infty$ for all $j \in \mathcal{J}$. Thus, every optimal value $\mathfrak{p}_{ij}$ is finite. Therefore, by Bertsekas [22, Proposition 3.1.3], every $\mathfrak{p}_{ij}$ is attained, and therefore by Bertsekas [22, Proposition 4.4.2] strong duality holds between $\mathfrak{p}_{ij}$ and its dual problem, which we denote by $\mathfrak{d}_{ij}$, and it also holds that $\mathfrak{d}_{ij}$ is attained. A routine derivation via Lagrangian duality therefore yields that

$$\begin{aligned} \mathfrak{p}_{ij} &= \mathfrak{d}_{ij} \\ &= \inf_{\nu_{ij} \in \mathbb{R}^n} \left\{ \sum_{l \in \mathcal{J}} (\nu_{ij})_l (b_{ij} - b_{il}) - b_{ij} : y - a_{ij} + \sum_{l \in \mathcal{J}} (\nu_{ij})_l (a_{ij} - a_{il}) = 0,\ \nu_{ij} \geq 0 \right\}. \end{aligned}$$

Hence, $g_i^*(y) \leq h(\gamma)$ if and only if $\max_{j \in \mathcal{J}} \mathfrak{p}_{ij} \leq h(\gamma)$ if and only if $\mathfrak{p}_{ij} \leq h(\gamma)$ for all $j \in \mathcal{J}$. Thus, since $\mathfrak{d}_{ij}$ is attained, it holds that $g_i^*(y) \leq h(\gamma)$ if and only if, for all $j \in \mathcal{J}$, there exists $\nu_{ij} \in \mathbb{R}^n$ such that $\nu_{ij} \geq 0$, $y - a_{ij} + \sum_{l \in \mathcal{J}} (\nu_{ij})_l (a_{ij} - a_{il}) = 0$, and $-b_{ij} + \sum_{l \in \mathcal{J}} (\nu_{ij})_l (b_{ij} - b_{il}) \leq h(\gamma)$. This completes the proof. □

With the above conjugate and perspective derivations, our reformulations of $\underline{c}, \overline{c}$ are complete; they may now be directly solved using off-the-shelf convex optimization solvers.

*Remark* 2. Our developments can be generalized, so long as one can compute the appropriate conjugates and perspectives. In particular, the mathematical machinery yielding a discrete distribution solution to $p'$ from a solution to an associated finite-dimensional convex optimization problem may be applied to general convex functions $g_i$ and other (non-norm-based and non-polyhedral) convex attack sets $X$ [173]. In fact, moment constraints on $\mu \in \mathcal{P}(X)$ may even be added to the semi-infinite program $p'$, which may allow for modeling alternative "distributional attacks" beyond the standard "Dirac attack" at a single point considered here.

## 3.4 Numerical Simulations

In this section, we illustrate the utility of our method in both robust control and image classification settings.[3]

### Robust Control Certification

We take an illustrative robust control example adapted from the well-known autonomous vehicle collision avoidance problem [117, 140]. Consider two planar vehicles approaching an intersection located at the origin $(0, 0) \in \mathbb{R}^2$. One vehicle travels east with state $\mathbf{x}(t) = (x(t), \dot{x}(t)) \in \mathbb{R}^2$ at time $t$. The other vehicle, which we control and hence term the "ego vehicle," travels north with state $\mathbf{y}(t) = (y(t), \dot{y}(t))$. The eastbound uncontrolled vehicle has a fixed velocity ($\ddot{x}(t) = 0$ for all $t$). The full state $(x(t), \dot{x}(t), y(t), \dot{y}(t))$ is randomly initialized at $t = 0$ within $[-3, -2] \times [1/2, 5/2] \times [-3, -2] \times [0, 2]$. The vehicles are each 1 unit long and $1/2$ unit wide, matching the width of the road. Thus, a vehicle is considered to be in the intersection if the absolute value of its position is less than $3/4$. If the vehicles collide, the simulation is stopped. We simulate standard double integrator dynamics with a time step $\Delta t = 0.05$ for 100 steps.

We control the northbound vehicle using a learned policy $u(t) = -\pi_\theta(\mathbf{x}(t), \mathbf{y}(t))$ that enters the dynamics as $\ddot{y}(t) = \Pi_{[-1,1]}(u(t))$, where $\pi_\theta \colon \mathbb{R}^4 \to \mathbb{R}$ is a min-max affine function with $m = n = 10$ and $\Pi_{[-1,1]}$ is the natural projection mapping of $\mathbb{R}$ onto $[-1, 1]$. Our robustness certificates apply for all training schemes, e.g., reinforcement learning and imitation learning. We train $\pi_\theta$ using imitation learning on 500 trajectories generated by a hand-programmed expert policy $\pi^\star$. We use the mean squared error loss function and train for 20 epochs using the Adam optimizer at a learning rate of 0.01. The expert policy $\pi^\star$ is designed to stop the ego vehicle $\delta = 0.1$ units before the intersection with a constant acceleration, then apply no acceleration until the tail of the uncontrolled vehicle is $\delta$ units past the intersection, and then accelerate with the maximum input of 1.

We now consider certifying the safety of our control system. Our goal is to guarantee that the ego vehicle always brakes when the uncontrolled vehicle is approaching or inside the

[3] All simulations are conducted on a Ubuntu 22.04 instance with an Intel i7-9700K CPU and NVIDIA RTX A6000 GPU.

intersection. This enforcement of braking corresponds to ensuring that the largest acceleration signal $u(t)$ is less than zero, which amounts to minimizing the output of $\pi_\theta$ over the set of states for which we desire braking. This is formalized by requiring braking for all states in the set

$$X = [-3 + \delta, \tfrac{3}{4}] \times [\tfrac{1}{2} + \delta, \tfrac{5}{2} - \delta] \times [-3 + \delta, -\tfrac{3}{4}] \times [\delta, 2 - \delta],$$

which consists of states where the uncontrolled vehicle is approaching or in the intersection and the ego vehicle is approaching the intersection. The small positive constant $\delta = 0.1$ accounts for boundary states where expert trajectories may not have been sampled.

Utilizing our robustness certificates from Section 3.3, we verify that indeed $u(t) = -\pi_\theta(\mathbf{x}(t), \mathbf{y}(t)) < 0$ for all states $(\mathbf{x}(t), \mathbf{y}(t)) \in X$. For visual purposes, we also consider fixing a particular $\mathbf{y}(t)$ and computing the largest possible acceleration $u(t)$ amongst all uncontrolled vehicle states $\mathbf{x}(t)$ captured by $X$. The solutions to this problem over a range of $\mathbf{y}(t)$ are plotted in Figure 3.1. As expected, for all $\mathbf{y}(t)$, the ego vehicle is braking. As the ego vehicle approaches the intersection (large $y(t)$) or becomes faster (large $\dot{y}(t)$), we certify that the controller brakes more heavily.

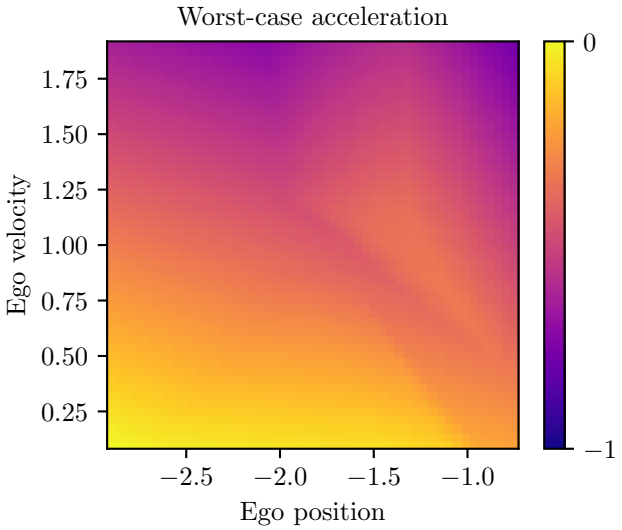



Figure 3.1: Largest possible acceleration over all uncontrolled vehicle states for particular values of the ego vehicle state. The output is always negative, ensuring some level of braking.

## Image Classification

We demonstrate the tightness and efficiency of our method on an image classification example adapted from Pfrommer* et al. [113]. The task is to distinguish between two visually similar MNIST classes: the digits 3 and 8 [84]. As we consider the asymmetric setting, we aim to certify predictions for one particular class, which we take to be the class of 3's, while

maintaining high clean accuracy for both classes. We consider the attack set $X = \{x \in \mathbb{R}^d : \|x - \overline{x}\|_\infty \leq \epsilon\}$ over a range of radii $\epsilon > 0$ around test images $\overline{x}$. In this setting, certificates ensure that pixelwise adversarial alterations of an image $\overline{x}$ of a 3 cannot fool the classifier into predicting an 8.

We compare two approaches: 1) directly learning our min-max representation with $n = m = 15$ and certifying via our convex optimization-based certificates, and 2) learning a standard composition-based ReLU model and certifying via the state-of-the-art verifier $\alpha, \beta$-CROWN [143]. Since $\alpha, \beta$-CROWN's worst-case runtime scales exponentially with model size, we instantiate the standard ReLU model with one hidden layer and 100 hidden units, which is the smallest hidden layer size that yields comparable clean accuracy to our min-max representation. We use adversarial training (see Madry et al. [95]) with $\ell_\infty$-attacks starting at a radius of 0.001 and linearly interpolate to a radius of $\epsilon_{\text{train}}$ over the first 20 epochs, where $\epsilon_{\text{train}} = 0.05$ for our model and $\epsilon_{\text{train}} = 0.3$ for the standard ReLU model. Both models are trained using the Adam optimizer with a learning rate of 0.001 for 60 epochs.

Figure 3.2 compares the certified accuracy (averaged over the test inputs) of our method against that of $\alpha, \beta$-CROWN. As certifying at a particular $\epsilon$ using our method is fast, for each test input, the largest certifiable $\ell_\infty$-radius is found using binary search in order to yield a smooth certified accuracy curve. On the other hand, due to the expensive runtime of $\alpha, \beta$-CROWN, we only certify at the select radii shown. Our min-max representation exceeds the state-of-the-art baseline certified radii at far faster runtimes: certifying a single input-radius pair $(\overline{x}, \epsilon)$ takes on average 3.67 seconds with $\alpha, \beta$-CROWN versus only 0.48 seconds with our method. We note that our runtime comparisons are solely based off of models with equivalent clean accuracy.

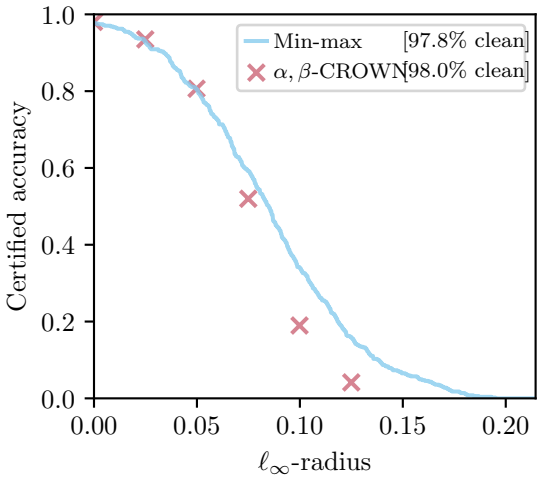



Figure 3.2: Certified accuracies of our min-max representation and of $\alpha, \beta$-CROWN on the MNIST 3-versus-8 dataset.

## 3.5 Conclusions

In this chapter, we *exactly* solve the nonconvex robustness certification problem over convex attack sets for min-max representations of ReLU neural networks by developing a tractable convex reformulation. An interesting line of future work may include developing more efficient min-max representations or estimations for arbitrary ReLU neural networks, so that the advantageous optimization properties derived in this chapter may be easily applied. Other interest lies in comparing the number of affine regions of a general min-max affine function versus that of a general ReLU neural network.

# Chapter 4

# Data-Driven Certification for Probabilistic Robustness

Most certification methods in the literature are designed for adversarial or worst-case inputs, but researchers have recently shown a need for methods that consider random input noise. In this chapter, we examine the setting where inputs are subject to random noise coming from an arbitrary probability distribution. We propose a robustness certification method that lower-bounds the probability that network outputs are safe. This bound is cast as a chance-constrained optimization problem, which is then reformulated using input-output samples to make the optimization constraints tractable. We develop sufficient conditions for the resulting optimization to be convex, as well as on the number of samples needed to make the robustness bound hold with overwhelming probability. We show for a special case that the proposed optimization reduces to an intuitive closed-form solution. Case studies on synthetic, MNIST, and CIFAR-10 networks experimentally demonstrate that this method is able to certify robustness against various input noise regimes over larger uncertainty regions than prior state-of-the-art techniques.

This chapter is based on the following previously published work:

[9] Brendon G. Anderson and Somayeh Sojoudi, "Data-driven certification of neural networks with random input noise," *IEEE Transactions on Control of Network Systems*, 2022.

## 4.1 Introduction

Much of the literature on robustness certification has revolved around adversarial inputs, i.e., inputs with small-magnitude perturbations that are designed to cause a worst-case prediction [150, 148, 116, 4]. However, as argued in Webb et al. [146] and Mangal, Nori, and Orso [97], random input uncertainty better models reality in many applications. Areas that commonly use a probabilistic model of uncertainty include stochastic control and

finance, where unpredictable measurement errors and state disturbances are assumed to be random [12, 56]. The stochastic framework also naturally encapsulates applications where unbounded uncertainties may exist, albeit with an extremely low probability. This is typical in real-world applications such as aviation [162]. In fact, the International Organization for Standardization (ISO) asserts in their guide on safety aspects that there is never absolute safety, and therefore the goal is to achieve what they define to be *tolerable risk* [68, 70]. Not only are random uncertainties pervasive and realistic, they have been shown to pose a legitimate threat—small uniform noise causes misclassification rates of well over 10% on MNIST and CIFAR-10 networks, and for Bernoulli noise the misclassification rates become drastically worse, sometimes reaching 100% [147].

The aforementioned motivations have led to an influx of recent works considering robustness against random inputs, which we review below. Many of them make stringent assumptions on the structure of the network or input distribution, or the formal certification guarantees are relaxed or eliminated in order to enhance computational tractability. Since neural networks are more sensitive to adversarial inputs than to random input noise [57], worst-case sensitivity analyses are too conservative for random input noise when the goal at hand is to achieve a tolerable risk level, whereas high-probability robustness certificates may completely fail in the presence of adversaries; the two settings are disjoint and should be studied using distinct methods. The work in this chapter is intended to certify robustness against random input noise with minimal conservatism, and is not intended to assess adversarial robustness.

## Related Works

In this section, we review the state-of-the-art methods for assessing robustness to random inputs, highlight their usages, and address their limitations. For instance, Mangal, Nori, and Orso [97] defines robustness as the network output being Lipschitz continuous with high probability when two inputs are chosen randomly. Their proposed method is limited to neural networks composed of conditional affine transformations, e.g., ReLU networks. On the other hand, Weng et al. [147] analytically bounds the probability that a classifier's margin function exceeds a given value. Although this probabilistic method applies to general neural network models, it assumes that the random input noise is constrained to an $\ell_p$-norm ball and is either Gaussian or has independent coordinates. Furthermore, their bounding technique relies on worst-case analysis methods, making their resulting certificates relatively loose (see Section 4.6).

In Webb et al. [146], robustness is measured by the probability that random input noise results in misclassification. The authors propose a sampling-based approximation of the robustness level. However, no theoretical guarantees are given to certify the network's robustness. Contrarily, Dvijotham et al. [43] formally bounds the probability that a random input maps to an unsafe output. However, the bounding function is nonconvex, and therefore obtaining tight bounds amounts to nonconvex optimization. Alternatively, Franceschi, Fawzi, and Fawzi [57] bounds the size of a random input perturbation that causes a classifier

prediction error with high probability. Their bounds provide elegant theoretical guarantees, but they depend on the network's worst-case robustness level, which is generally NP-hard to compute without approximation errors or additional assumptions [148, 73].

The works Fazlyab, Morari, and Pappas [49] and Couellan [36] provide methods to guarantee that network outputs do not significantly deviate from the nominal output when the input is subject to random uncertainty. This corresponds to the problem of localizing the network outputs within the output space. The work Fazlyab, Morari, and Pappas [49] uses the output localization to issue high-probability guarantees for the network's robustness. However, their method requires solving a semidefinite program, and their results are demonstrated on small, single-layer networks, so it is not clear whether their method scales to realistic applications. The concentration bounds presented in Couellan [36] can be applied to deep networks, but their localization results do not immediately translate into a meaningful certificate of robustness.

The authors of Devonport and Arcak [40] and Li et al. [89] use input-output samples to learn how input noise is propagated to the output space through a method called scenario optimization. This approach naturally embeds the stochastic nature of the input noise into the assessment procedure. The work Devonport and Arcak [40] provides a method to estimate a network's set of possible outputs, but this localization may fail to determine the safety of the output since the underlying optimization does not directly consider any safety specifications, e.g., classification boundaries. We demonstrate this phenomenon in our comparison to output set estimation in Section 4.6. Other scenario-based output set estimation techniques for general nonlinear maps, such as Yang et al. [159], Fravolini et al. [58], and Sartipizadeh et al. [125], suffer from the same limitation in the context of neural network certification. Contrarily, Li et al. [89] directly considers output safety in their scenario approach. However, their method makes use of an affine approximation to the network's nonlinear margin function, a worst-case analysis technique from the adversarial robustness literature. In our numerical simulations, we show that this worst-case technique yields loose robustness bounds as the size of the input noise increases.

Finally, Zakrzewski [162] also uses sampled data to bound the probability of failure. Their guarantees take the probably approximately correct (PAC) form in terms of probability levels $\epsilon, \delta \in [0, 1]$, and they improve the sample complexity from $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$ of the naive Chernoff bound to $O(\frac{1}{\epsilon} \log \frac{1}{\delta})$. This is achieved by imposing a Bayesian framework and assuming that the failure probability follows a uniform prior distribution. As the authors remark, this is a "very conservative choice." In contrast, the method to be proposed in this chapter solves for the optimal (least conservative) robustness bound of this form with the same $O(\frac{1}{\epsilon} \log \frac{1}{\delta})$ sample complexity.

## Contributions

In this chapter, we develop a data-driven framework for certifying neural network robustness against random input noise using scenario optimization. Our direct approach avoids worst-case analysis techniques, such as those found in the adversarial robustness literature,

e.g., Weng et al. [147], and also avoids the need for selecting a Bayesian prior distribution governing the network's failure probability, as in Zakrzewski [162]. The procedure is capable of localizing network outputs into a general class of sets, and we develop sufficient conditions on this class to ensure that the procedure amounts to a convex optimization problem. Furthermore, we develop formal guarantees that the resulting robustness certificate holds with overwhelming probability upon using sufficiently many samples in the scenario optimization. Although the method is applicable to all neural networks and all input noise distributions, we show how to exploit the structure of networks with affinely bounded activation functions in order to reduce sample complexity.

Our numerical simulations demonstrate that the proposed optimization is capable of issuing robustness certificates in cases where the two-step process of optimally localizing the outputs (e.g., using Devonport and Arcak [40]) and then certifying them cannot, providing a novel perspective that output set estimation techniques do not necessarily work well for certification. Furthermore, we show on both synthetic networks and large MNIST and CIFAR-10 networks that our robustness bounds are much tighter than those obtained by the state-of-the-art method Weng et al. [147], particularly for large noise levels.

## Outline

We begin by formulating the certification problem in Section 4.2. In Section 4.3 and Section 4.4, we propose an optimization to solve for the high-probability robustness certificate, and then show that it suffices to solve a data-driven convex optimization problem with sufficiently many samples. Next, we demonstrate how to exploit the neural network structure to reduce the sample complexity of the proposed method in Section 4.5. We numerically illustrate the results and compare to state-of-the-art methods in Section 4.6. We conclude in Section 4.7. Supporting lemmas and various supplementary materials are presented in the appendices.

## Notations

In this section, we define the notations used throughout this chapter. The ceiling of $x \in \mathbb{R}$ is written $\lceil x \rceil$. For $x, y \in \mathbb{R}^n$, we define $[x, y] = \{z \in \mathbb{R}^n : x \leq z \leq y\}$, where the inequalities are interpreted element-wise. Given a set $\mathcal{X}$, we denote its power set by $\mathcal{P}(\mathcal{X})$. The Minkowski sum of sets $\mathcal{X}$ and $\mathcal{Y}$ is defined as $\mathcal{X} + \mathcal{Y} = \{x + y : x \in \mathcal{X},\ y \in \mathcal{Y}\}$. We define $\mathbb{R}_{++} = \{x \in \mathbb{R} : x > 0\}$. For a function $f \colon \mathbb{R}^m \to \mathbb{R}^n$, we write the image of $\mathcal{X} \subseteq \mathbb{R}^m$ under $f$ as $f(\mathcal{X}) = \{f(x) \in \mathbb{R}^n : x \in \mathcal{X}\}$. If $g \colon \mathbb{R}^n \to \mathbb{R}^p$ is another function, we define the composition $g \circ f \colon \mathbb{R}^m \to \mathbb{R}^p$ by $g \circ f(x) = g(f(x))$. If $X \colon \Omega \to \mathbb{R}^n$ is a random variable on a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, $g \colon \mathbb{R}^n \to \mathbb{R}$ is a Borel measurable function, and $c \in \mathbb{R}$, we use the notation $\mathbb{P}(g(X) \geq c)$ to mean $\mathbb{P}(\{\omega \in \Omega : g(X(\omega)) \geq c\})$. Similarly, if $\mathcal{S} \subseteq \mathbb{R}^m$ is a Borel set and $h \colon \mathbb{R}^n \to \mathbb{R}^m$ is a Borel measurable function, we write $\mathbb{P}(h(X) \in \mathcal{S})$ to mean $\mathbb{P}(\{\omega \in \Omega : h(X(\omega)) \in \mathcal{S}\})$. For a norm $\|\cdot\|$ on $\mathbb{R}^n$ we denote its dual norm by $\|\cdot\|_*$,

where $\|y\|_* = \sup\{x^\top y : \|x\| \leq 1\}$. We assume throughout that optimization problems are attained by a solution.

## 4.2 Problem Statement

### Network Description, Safe Set, and Safety Level

In this chapter, we consider a Borel measurable neural network $f\colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ with arbitrary structure and parameters.[1] We assume that the input to the network is a random variable $X\colon \Omega_X \to \mathbb{R}^{n_x}$ on a fixed probability space $(\Omega_X, \mathcal{F}_X, \mathbb{P}_X)$.[2] We do not assume that the distribution $\mathbb{P}_X$ is exactly known—we only assume that we are able to sample from $\mathbb{P}_X$. The support of the probability measure $\mathbb{P}_X$ is called the *input set*, which is denoted by $\mathcal{X} \subseteq \mathbb{R}^{n_x}$. The *output set* of the network is defined to be $\mathcal{Y} = f(\mathcal{X}) \subseteq \mathbb{R}^{n_y}$.

Next, consider a given convex polyhedral *safe set* $\mathcal{S} = \{y \in \mathbb{R}^{n_y} : Ay + b \geq 0\}$, where $A \in \mathbb{R}^{n_s \times n_y}$ and $b \in \mathbb{R}^{n_s}$. Without loss of generality, we assume that $n_s = 1$, henceforth setting $A = a^\top \in \mathbb{R}^{1 \times n_y}$ and $b \in \mathbb{R}$.[3] The results of this chapter can be immediately generalized to the case where $n_s > 1$. See Appendix 4.B for a detailed explanation.

The elements of the set $\mathcal{S}$ are considered to be *safe*. For a point $y$ in the output space $\mathbb{R}^{n_y}$, the value $s(y) = a^\top y + b$ is called the *safety level* of $y$. The point $y$ is safe if and only if its safety level is nonnegative. Broadly speaking, the overall goal of this chapter is to certify that the random output $Y = f(X)$ is safe. When this holds for all or many of the possible outputs in $\mathcal{Y}$, we obtain a natural certificate for the robustness of the network against the random noise.

**Example 1.** When $f$ is an $n_y$-class classifier and $\bar{x} \in \mathbb{R}^{n_x}$ is a deterministic nominal input with class $i^\star \in \arg\max_{i \in \{1,2,\ldots,n_y\}} f_i(\bar{x})$, a common goal is to certify that additive random noise $\delta$ on $\bar{x}$ does not cause misclassification [147]. This problem falls within our framework by defining the safety level of $f(X)$ to be the *margin function* value $g_i(X) \coloneqq f_{i^\star}(X) - f_i(X)$ for $X = \bar{x} + \delta$.

[1] Borel measurability of the network is almost always satisfied in practice. Indeed, every continuous function is Borel measurable.

[2] It is easily verified that all probabilistic expressions in this chapter are well-defined from a measure-theoretic perspective. We leave out these measurability verifications for the sake of exposition.

[3] The polyhedral safe set assumption is without loss of generality. Suppose that the safe set is $\mathcal{S} = \{y \in \mathbb{R}^{n_y} : a^\top g(y) + b \geq 0\}$ for some nonlinear Borel measurable $g\colon \mathbb{R}^{n_y} \to \mathbb{R}^{n_z}$ and some $a \in \mathbb{R}^{n_z}, b \in \mathbb{R}$. Then we can reduce the problem to our assumed form by considering $f' \coloneqq g \circ f$ to be the (Borel measurable) neural network and $\mathcal{S}' \coloneqq \{z \in \mathbb{R}^{n_z} : a^\top z + b \geq 0\}$ to be the (polyhedral) safe set, since then $f(x) \in \mathcal{S}$ if and only if $f'(x) \in \mathcal{S}'$, for all $x \in \mathcal{X}$. The architecture-dependent results of Section 4.5 must be applied to $f'$ with care, since $g$ must now be considered as another layer in the network, making Assumption 6 and Assumption 7 to follow slightly more stringent.

## Various Notions of Robustness

We now use the safety level of outputs to introduce three meaningful notions of robustness against random input noise, and discuss how they are related to one another.

### Deterministic Robustness Level

The *deterministic robustness level* of the network is defined as

$$r^\star = \inf_{y \in \mathcal{Y}} a^\top y + b. \tag{4.1}$$

If $r^\star \geq 0$, then $\mathcal{Y} \subseteq \mathcal{S}$, implying that the random output $Y = f(X)$ is safe with probability one. This notion of robustness coincides with that used when considering adversarial inputs [150, 116, 4], but the resulting worst-case safety level is often much lower than the safety levels of random outputs in practice [146]. Consequently, using $r^\star$ to assess robustness may falsely indicate that the network is sensitive to the input noise.

### Approximate Robustness Level

Although $r^\star$ can issue strong guarantees about the safety of the network output, (4.1) amounts to an intractable nonconvex optimization problem, since $\mathcal{Y}$ is generally a nonconvex set. Instead of computing $r^\star$, we can consider approximating it by

$$\hat{r}(\hat{\mathcal{Y}}) = \inf_{y \in \hat{\mathcal{Y}}} a^\top y + b, \tag{4.2}$$

where $\hat{\mathcal{Y}} \subseteq \mathbb{R}^{n_y}$, termed the *surrogate output set*, is more tractable than $\mathcal{Y}$, and preferably convex. We call (4.2) the *approximate robustness level* of the network. If $\mathcal{Y} \subseteq \hat{\mathcal{Y}}$, then $\hat{r}(\hat{\mathcal{Y}}) \leq r^\star$. In this case, if $\hat{r}(\hat{\mathcal{Y}}) \geq 0$, then the random output $Y = f(X)$ is safe with probability one. In general, choosing $\hat{\mathcal{Y}}$ to cover $\mathcal{Y}$ makes $\hat{r}(\hat{\mathcal{Y}})$ an over-conservative measure of robustness for the same reasons $r^\star$ is.

### Probabilistic Robustness Level

The notion of deterministic robustness is too strong for applications involving random input noise, as many input distributions have unbounded support or have their worst-case inputs in regions of low probability measure [146]. Furthermore, (4.1) and (4.2) neglect the distributional information given for $X$. This conservatism means that the robustness levels (4.1) and (4.2), with $\mathcal{Y} \subseteq \hat{\mathcal{Y}}$, are generally unable to certify that $Y$ is safe, even when $Y$ concentrates around safe outputs. Consequently, for a prescribed probability level $\epsilon \in [0, 1]$, we define the more natural *probabilistic robustness level* of the network to be

$$\bar{r}(\epsilon) = \sup\{r \in \mathbb{R} : \mathbb{P}_X(a^\top f(X) + b \geq r) \geq 1 - \epsilon\}. \tag{4.3}$$

Intuitively, the random output $f(X)$ has a safety level at least $\bar{r}(\epsilon)$ with high probability. In the case that $\bar{r}(\epsilon) \geq 0$, we certify that the random output $Y = f(X)$ is safe with probability $1 - \epsilon$, and we say that the network is *probabilistically robust*.[4] Another interpretation of probabilistic robustness is that the majority of possible outputs (with respect to the distribution $\mathbb{P}_X$) are safe. The probabilistic robustness level is catered towards our setting of random input noise, and, compared to the worst-case alternatives, reduces conservatism by considering the actual likelihood of the possible inputs. This is precisely the notion of robustness we adopt in this chapter. We remark that this definition of probabilistic robustness coincides with that used in Webb et al. [146] and Weng et al. [147], albeit our subsequent analysis and guarantees vastly differ from these works.

**Example 2.** Consider again the classification network in Example 1. In this setting, if $r^\star \geq 0$, then the probability of misclassification is zero, in which case the noise $\delta$ is not important. On the other hand, if there exists a perturbed input $\bar{x}+\delta$ in the input set $\mathcal{X}$ that is misclassified, then $r^\star < 0$. It may be the case, however, that this misclassification occurs with a sufficiently low probability with respect to the tolerance $\epsilon$. This is precisely what we seek to certify: that the probability of misclassification is less than $\epsilon$, which mathematically amounts to showing that $\bar{r}(\epsilon) \geq 0$.

In this chapter, we aim to certify the safety of $Y = f(X)$ by lower-bounding the probabilistic robustness level $\bar{r}(\epsilon)$. A trivial lower bound is easily verified: $r^\star \leq \bar{r}(\epsilon)$ for all $\epsilon \in [0, 1]$, and $r^\star = \bar{r}(0)$. However, as we will see in Section 4.6, this approach of considering *worst-case* inputs instead of *likely* inputs, often results in a very loose lower bound, sometimes failing to issue a robustness guarantee at all. In the next section, we show that by using a special type of surrogate output set in $\hat{r}(\hat{\mathcal{Y}})$, we can optimize a lower bound on $\bar{r}(\epsilon)$ and obtain an estimate of the output set $\mathcal{Y}$ as a natural byproduct.

## 4.3 Formulating the Certificate

### Bounding the Probabilistic Robustness Level

As we have seen, $\hat{r}(\hat{\mathcal{Y}}) \approx r^\star \leq \bar{r}(\epsilon)$. Two natural questions arise. 1) Can one use $\hat{r}(\hat{\mathcal{Y}})$ to certifiably lower-bound the quantity $\bar{r}(\epsilon)$ of interest? 2) If so, how can $\hat{\mathcal{Y}}$ be chosen to optimize the bound? In this section, we study the first question. As it turns out, such a lower bound holds so long as $\hat{\mathcal{Y}}$ has high enough coverage over $\mathcal{Y}$. Before proving this claim, we formally define this notion of coverage.

**Definition 7.** Let $\hat{\mathcal{Y}}$ be a subset of $\mathbb{R}^{n_y}$. For $\epsilon \in [0, 1]$, the set $\hat{\mathcal{Y}}$ is said to be an *$\epsilon$-cover* of $\mathcal{Y} = f(\mathcal{X})$ if $\hat{\mathcal{Y}}$ is Borel measurable and $\mathbb{P}_X(f(X) \in \hat{\mathcal{Y}}) \geq 1 - \epsilon$.

[4]Note the distinction: "safety" is a property of outputs, whereas "robustness" is a property of the neural network (with respect to the noisy input distribution $\mathbb{P}_X$). We are always careful when using these terminologies.

Intuitively, an $\epsilon$-cover of $\mathcal{Y}$ is a set that contains $Y = f(X)$ with high probability. If we can compute an $\epsilon$-cover of $\mathcal{Y}$, then we will have localized the output with high confidence. By restricting $\hat{\mathcal{Y}}$ in (4.2) to be an $\epsilon$-cover of $\mathcal{Y}$, we ensure that the approximate robustness level takes into account the *likely* inputs $X$, but not necessarily the worst-case inputs. Consequently, this permits $\hat{r}(\hat{\mathcal{Y}})$ to be greater than $r^\star$, reducing the conservatism in our measure of robustness caused by unlikely worst-case inputs. We now show that this special type of surrogate output set is a good enough estimate of $\mathcal{Y}$ to maintain the lower bound on $\bar{r}(\epsilon)$ that we seek.

**Proposition 14.** *Let $\hat{\mathcal{Y}}$ be an arbitrary subset of $\mathbb{R}^{n_y}$. If $\hat{\mathcal{Y}}$ is an $\epsilon$-cover of $\mathcal{Y} = f(\mathcal{X})$, then*

$$\hat{r}(\hat{\mathcal{Y}}) \leq \bar{r}(\epsilon). \tag{4.4}$$

*Proof.* Note that $y \in \hat{\mathcal{Y}}$ implies that $a^\top y + b \geq \hat{r}(\hat{\mathcal{Y}})$ by (4.2). Therefore, it holds that $\mathbb{P}_X(f(X) \in \hat{\mathcal{Y}}) \leq \mathbb{P}_X(a^\top f(X) + b \geq \hat{r}(\hat{\mathcal{Y}}))$. Since $\hat{\mathcal{Y}}$ is an $\epsilon$-cover of $\mathcal{Y}$, we have that $\mathbb{P}_X(f(X) \in \hat{\mathcal{Y}}) \geq 1 - \epsilon$. Hence,

$$1 - \epsilon \leq \mathbb{P}_X(f(X) \in \hat{\mathcal{Y}}) \leq \mathbb{P}_X(a^\top f(X) + b \geq \hat{r}(\hat{\mathcal{Y}})).$$

This shows that $\hat{r}(\hat{\mathcal{Y}})$ is feasible for the optimization (4.3). Therefore, $\hat{r}(\hat{\mathcal{Y}}) \leq \bar{r}(\epsilon)$, as desired. □

## Optimizing the Bound

From Proposition 14, we know that $\epsilon$-covers constitute good choices of the surrogate output set $\hat{\mathcal{Y}}$ used to compute the approximate robustness level. This is because the random output $Y = f(X)$ is guaranteed to have safety level at least $\hat{r}(\hat{\mathcal{Y}})$ with high probability. However, it is entirely possible that the choice of $\epsilon$-cover results in $\hat{r}(\hat{\mathcal{Y}}) < 0$, even when the network is probabilistically robust. In this case, $\hat{r}(\hat{\mathcal{Y}})$ fails to issue a high-probability certificate for the safety of the random output $Y = f(X)$, despite $\hat{\mathcal{Y}}$ being able to localize it.

To overcome the above problem, we turn to studying our second inquiry from earlier, namely, how to optimize the lower bound (4.4). This amounts to finding an $\epsilon$-cover of $\mathcal{Y}$ that maximizes $\hat{r}(\hat{\mathcal{Y}})$. Since optimizing over all subsets of $\mathbb{R}^{n_y}$ is intractable, we restrict our search to sets within a class $\mathcal{H} = \{h(\theta) : \theta \in \Theta\}$ parameterized by a parameter set $\Theta \subseteq \mathbb{R}^p$ and a set-valued function $h \colon \mathbb{R}^p \to \mathcal{P}(\mathbb{R}^{n_y})$. We assume throughout that the class $\mathcal{H}$ is chosen such that $h(\theta)$ is a Borel set for all parameters $\theta \in \Theta$. A concrete example of one such class is given below.

**Example 3.** Let $\|\cdot\|$ be a fixed norm on $\mathbb{R}^{n_y}$ and $\Theta = \mathbb{R}^{n_y} \times \mathbb{R}_{++}$. Defining $p = n_y + 1$, let $h \colon \mathbb{R}^p \to \mathcal{P}(\mathbb{R}^{n_y})$ be defined by $h(\bar{y}, r) = \{y \in \mathbb{R}^{n_y} : \|y - \bar{y}\| \leq r\}$. Then, $\Theta$ and $h(\cdot)$ define the class of $\|\cdot\|$-norm balls:

$$\mathcal{H} = \big\{\{y \in \mathbb{R}^{n_y} : \|y - \bar{y}\| \leq r\} : r > 0,\ \bar{y} \in \mathbb{R}^{n_y}\big\}.$$

The problem of choosing $h(\cdot)$ and $\Theta$ (and therefore also $\mathcal{H}$) is discussed in detail in Section 4.4. By restricting our search for $\epsilon$-covers to within the class $\mathcal{H}$, our search reduces to maximizing the approximate robustness level over the parameter set $\Theta$. By slightly abusing notation, we denote the dependence of the approximate robustness level on the parameter $\theta$ explicitly as $\hat{r}(\theta) = \inf\{a^\top y + b : y \in h(\theta)\}$, and we formulate the following optimization problem:

$$\begin{aligned} \underset{\theta \in \Theta}{\text{maximize}} \quad & \hat{r}(\theta) - \lambda v(\theta) \\ \text{subject to} \quad & \mathbb{P}_X(f(X) \in h(\theta)) \geq 1 - \epsilon, \end{aligned} \tag{4.5}$$

where $\lambda \geq 0$ and $v \colon \mathbb{R}^p \to \mathbb{R}$ is taken to be a nonnegative convex function on $\Theta$ that increases with the volume of $h(\theta)$. The objective $\hat{r}(\theta)$ in (4.5) is the approximate robustness level computed using the set $h(\theta)$ as the surrogate output set. The constraint $\mathbb{P}_X(f(X) \in h(\theta)) \geq 1 - \epsilon$ enforces that we only consider parameters $\theta$ such that $h(\theta)$ is an $\epsilon$-cover of $\mathcal{Y}$. The regularization term $-\lambda v(\theta)$ penalizes the size of $h(\theta)$. This makes the set $h(\theta)$ as small as possible while maintaining its $\epsilon$-coverage, thereby yielding the tightest localization of the output $Y$. The regularization is done at the expense of a slightly suboptimal bound (4.4), and can be eliminated by setting $\lambda = 0$, if no localization of the output $Y$ is desired. On the other hand, increasing $\lambda$ amounts to putting more assessment effort into localizing $Y$, making the $\epsilon$-cover $h(\theta)$ a better estimate of $\mathcal{Y}$. This certification-localization tradeoff is experimentally explored in the illustrative example of Section 4.6.

## 4.4 Data-Driven Reformulation

Even when the set $h(\theta)$ is convex for all $\theta \in \Theta$, the probabilistic constraint in (4.5) is in general nonconvex [107]. Constraints of this form are referred to as *chance constraints*, and there exist various approaches to reformulating and relaxing them into convex constraints. Since the problem at hand considers neural networks whose models are usually complicated to analyze, but whose input-output samples are easily obtained, we seek a data-driven approach to approximately enforcing the chance constraint in (4.5), without losing the robustness certificate provided by the solution. The *scenario approach* is a popular method within the stochastic optimization and robust control communities that replaces the chance constraint with hard constraints on a number of random samples [107, 132, 26, 93]. The scenario approach has been studied for general problems, even those with nonconvex objectives and those whose resulting hard constraints are nonconvex [27]. However, the most powerful use of scenario optimization arises when the resultant scenario problem is convex, as then *a priori* probabilistic guarantees can be made about the solution's feasibility for the original chance constraint. As we will soon see, this sampling-based method fits nicely into the framework of our problem, and maintains a lower bound on $\bar{r}(\epsilon)$ with high probability, provided that a sufficiently large number of samples is used and the scenario problem is convex.

To implement the scenario approach, suppose that $\{x_j : j \in \{1, 2, \ldots, N\}\} \subseteq \mathcal{X}$ is a set of $N$ independent and identically distributed samples drawn from $\mathbb{P}_X$. For each input $x_j$,

we compute its corresponding output $y_j = f(x_j)$. Then, replacing the chance constraint in (4.5) with $N$ hard constraints on the samples $y_j$ yields the following scenario optimization:

$$\begin{aligned} \underset{\theta \in \Theta}{\text{maximize}} \quad & \hat{r}(\theta) - \lambda v(\theta) \\ \text{subject to} \quad & y_j \in h(\theta) \text{ for all } j \in \{1, 2, \ldots, N\}. \end{aligned} \tag{4.6}$$

Note that, because the data $y_j$ is random, solutions $\theta^\star$ to (4.6) are random. We assume throughout the chapter that (4.6) is attained by a solution $\theta^\star$, and we denote the probability space on which it is defined by $(\Omega_{\theta^\star}, \mathcal{F}_{\theta^\star}, \mathbb{P}_{\theta^\star})$.

*Remark* 3. The above assumption of independent and identically distributed samples is critical for relating the solution $\theta^\star$ back to the original chance-constrained problem (4.5). In particular, it is a key assumption on which the forthcoming high-probability robustness certificate in Theorem 11 rests. Despite these assumptions holding in many practical models, the independence may be violated in certain applications with inherent time-correlation between samples, and the assumption prevents the use of selective sampling to improve the efficiency of the scenario approach.

The identical distribution assumption is also critical, and it may be violated in two main ways. First, the underlying distribution of the data used in (4.6) may change from sample to sample, and second, the underlying noise distribution of the actual input may be different in practice from the samples used in the robustness certification procedure. Despite these sources of modeling error, our scenario-based approach can be modified into a distributionally robust variant to still give high-probability robustness certificates in the case that the distribution of the input is contained in a finite set of possible distributions.

As mentioned in Section 4.1, the scenario approach was used recently in reachable set estimation for dynamical systems [40]. We remark that (4.6) recovers the scenario optimization of Devonport and Arcak [40] in the special case that the objective is re-scaled to $\frac{1}{\lambda}\hat{r}(\theta) - v(\theta)$ and $\lambda \to \infty$, the regularizer $v(\theta)$ equals the volume of the set $h(\theta)$, and $\mathcal{H}$ is the norm ball class. This reduction amounts to finding the tightest norm ball $\epsilon$-cover of $\mathcal{Y}$, without regard to optimizing the lower bound (4.4) of interest. In our comparison to output set estimation in Section 4.6, we demonstrate the necessity for the more general formulation (4.6) by giving an example where reducing to the special case of Devonport and Arcak [40] causes robustness certification to fail, despite finding the tightest $\epsilon$-cover of $\mathcal{Y}$.

Although the scenario approach has eliminated the chance constraint from (4.5), there remain two problems to consider. First, it is not immediately clear whether (4.6) is convex or computationally tractable, as it has an inherent max-min optimization structure. However, it is important to ensure the problem's convexity, since no *a priori* guarantees can be made regarding the feasibility of $\theta^\star$ for the original chance constraint in the general case of non-convex scenario optimization [27]. In the next section, we leverage results from parametric optimization to develop conditions on our choice of $\Theta$ and $h(\cdot)$ to ensure that (4.6) is convex. Second, the solution of (4.6) gives a random approximation to the solution of (4.5), which optimizes the bound (4.4) on $\bar{r}(\epsilon)$. In the section after next, we develop formal guarantees

showing that the solution of (4.6) maintains a lower bound on $\bar{r}(\epsilon)$ with high probability, provided that $N$ is sufficiently large.

## Conditions for Convex Optimization

In this section, we consider the choices of the parameter set $\Theta$ and the set-valued function $h(\cdot)$ on lower-bounding $\bar{r}(\epsilon)$, and on the tractability of the resulting scenario problem (4.6). A key insight is this: an $\epsilon$-cover of $\mathcal{Y}$ may in general be much larger than $\mathcal{Y}$ itself. This is because regions of an $\epsilon$-cover that do not intersect with $\mathcal{Y}$ also do not count towards the coverage proportion $1-\epsilon$. Therefore, if the class $\mathcal{H}$ from which we choose an $\epsilon$-cover does not have high enough complexity, then the $\epsilon$-covers within $\mathcal{H}$ may need to be exceedingly large in order to achieve $\epsilon$-coverage.

The problem with unnecessarily large $\epsilon$-covers is that the feasible set in the optimization defining $\hat{r}(\theta)$ includes many vectors $y$ that may not be actual outputs in $\mathcal{Y}$. In this case, $\hat{r}(\theta)$ is small, even though $\bar{r}(\epsilon)$ may be large. To avoid this problem, our choice of $\Theta$ and $h(\cdot)$ should ensure that the class $\mathcal{H}$ has high enough complexity. However, our choices should also yield a scenario problem (4.6) that is convex. Indeed, Theorem 10 gives sufficient conditions for the convexity of the scenario problem. Before presenting these conditions, let us recall a fundamental definition for set-valued functions.

**Definition 8.** A set-valued function $h\colon \mathbb{R}^p \to \mathcal{P}(\mathbb{R}^{n_y})$ is said to be *convex* on a convex set $\Theta \subseteq \mathbb{R}^p$ if

$$\big(\lambda h(\theta_1) + (1-\lambda)h(\theta_2)\big) \subseteq h(\lambda\theta_1 + (1-\lambda)\theta_2)$$

for all $\theta_1, \theta_2 \in \Theta$ and all $\lambda \in [0,1]$. The function $h(\cdot)$ is said to be *concave* on $\Theta$ if

$$h(\lambda\theta_1 + (1-\lambda)\theta_2) \subseteq \big(\lambda h(\theta_1) + (1-\lambda)h(\theta_2)\big)$$

for all $\theta_1, \theta_2 \in \Theta$ and all $\lambda \in [0,1]$. Finally, the function $h(\cdot)$ on $\Theta$ is said to be *affine* if it is both convex and concave.

**Example 4.** Consider the norm ball class $\mathcal{H}$ given in Example 3. It is easily verified by Definition 8 that the set-valued function $h(\cdot)$ defining the class $\mathcal{H}$ is affine on $\Theta = \mathbb{R}^{n_y} \times \mathbb{R}_{++}$.

With tools for defining and proving convexity of set-valued functions now in place, we can present conditions under which the scenario optimization (4.6) is convex, and therefore easily solvable. In Theorem 11, we will also rely on this convexity to guarantee with high probability that $h(\theta^\star)$ is an $\epsilon$-cover of $\mathcal{Y}$ and that our desired lower bound $\hat{r}(\theta^\star) \leq \bar{r}(\epsilon)$ holds. Generating such guarantees is in general not possible for nonconvex scenario optimization [27], further illustrating the importance of Theorem 10 below.

**Theorem 10.** *Consider the scenario problem* (4.6)*. Suppose that* $\Theta$ *takes the form*

$$\Theta = \{\theta \in \mathbb{R}^p : g_i(\theta) \leq 0 \text{ for all } i \in \{1, 2, \ldots, m\}\},$$

*where every* $g_i \colon \mathbb{R}^p \to \mathbb{R}$ *is convex. Furthermore, suppose that* $h(\cdot)$ *is a concave set-valued function that takes the form*

$$h(\theta) = \{y \in \mathbb{R}^{n_y} : h_i(y, \theta) \leq 0 \text{ for all } i \in \{1, 2, \ldots, n\}\},$$

*where* $h_i \colon \mathbb{R}^{n_y} \times \mathbb{R}^p \to \mathbb{R}$ *and* $h_i(y, \cdot)$ *is convex for all* $y \in \mathbb{R}^{n_y}$*. Then,* (4.6) *is a convex optimization problem.*

*Proof.* Since (4.6) is a maximization problem, we must show that under the assumptions on $\Theta$ and $h(\cdot)$, the objective is concave on $\Theta$ and the constraints are convex.

Let us first consider the objective $\hat{r}(\theta) - \lambda v(\theta)$, where $\hat{r}(\theta) = \inf\{a^\top y + b : y \in h(\theta)\}$. Since

1. $g(y, \theta) \coloneqq a^\top y + b$ is jointly concave on $\mathbb{R}^{n_y} \times \Theta$;
2. $h(\cdot)$ is a concave set-valued function on $\Theta$;
3. and $\Theta$ is a convex set;

Proposition 3.1 of Fiacco and Kyparisis [52] gives that $\hat{r}(\cdot)$ is a concave function on $\Theta$. Since $v(\cdot)$ is assumed to be convex on $\Theta$ and $\lambda \geq 0$, we conclude that the objective is concave.

Now, let us consider the constraints. The constraints $g_i(\theta) \leq 0$ are convex, so $\theta \in \Theta$ is a convex constraint. Next, the random constraint $y_j \in h(\theta)$ is equivalent to the constraint on $\theta$ that $h_i(y_j, \theta) \leq 0$ for all $i$. Since $h_i(y_j, \cdot)$ is a convex function, the constraint is convex. Since this holds for all $i \in \{1, 2, \ldots, n\}$ and all $j \in \{1, 2, \ldots, N\}$, we conclude that all of the constraints in (4.6) are convex. □

*Remark* 4. Theorem 10 is easily extended to include affine equality constraints in the forms taken by $\Theta$ and $h(\theta)$. Additionally, if $h_i(y, \theta)$ in Theorem 10 is *jointly* convex in $(y, \theta)$ for all $i$, one can show that $h(\cdot)$ is an affine set-valued function, and therefore $\hat{r}(\cdot)$ in (4.6) is affine (see, e.g., Proposition 4.2 of Fiacco and Kyparisis [52]). Therefore, if $v(\cdot)$ is also affine, the scenario problem (4.6) has an affine objective.

Theorem 10 precisely answers our earlier inquiry: the class $\mathcal{H}$ should be complex enough to contain tight $\epsilon$-covers of the output set $\mathcal{Y}$, but at the same time $\Theta$ should be defined by convex constraints and $h(\cdot)$ should be taken as a concave set-valued function also defined by convex constraints. Note that these conditions on $h(\cdot)$ are not as restrictive as they may seem. In particular, Example 4 shows for the norm ball class that $h(\cdot)$ is affine (and therefore concave) and defined by convex constraints, and that this holds *for all norms on* $\mathbb{R}^{n_y}$, even though norm functions themselves are not affine. Therefore, Theorem 10 guarantees that the scenario optimization (4.6) using the norm ball class is a convex problem. We verify this fact in the following example.

**Example 5.** Recall the norm ball class of Example 3 and Example 4. We show that (4.6) using this class is convex. Indeed, the approximate robustness level is

$$\hat{r}(\bar{y}, r) = \inf_{\|y-\bar{y}\| \leq r} a^\top y + b = a^\top \bar{y} - r\|a\|_* + b,$$

which is affine in the optimization variable $\theta = (\bar{y}, r)$. Hence, the scenario problem reduces to

$$\begin{array}{ll} \underset{(\bar{y},r)\in\mathbb{R}^{n_y}\times\mathbb{R}_{++}}{\text{maximize}} & b + a^\top \bar{y} - r\|a\|_* - \lambda v(\bar{y}, r) \\ \text{subject to} & \|y_j - \bar{y}\| \leq r \text{ for all } j \in \{1, 2, \ldots, N\}, \end{array} \tag{4.7}$$

which is a convex problem since $v(\cdot)$ is convex.

## High-Probability Guarantees

We now turn to consider the randomness of the scenario problem's optimal value. In particular, we ask the following question: Does the random solution to (4.6) maintain a certified lower bound on $\bar{r}(\epsilon)$? In Theorem 11, we show that the answer is affirmative with high probability, provided that the problem is convex and a large enough number of samples is used.

**Theorem 11.** *Let $\epsilon, \delta \in [0, 1]$. Assume that the scenario optimization* (4.6) *is convex and is attained by a solution $\theta^\star \in \mathbb{R}^p$. If $N \geq \frac{2}{\epsilon}\left(\log\frac{1}{\delta} + p\right)$, then the following inequalities hold:*

1. $\mathbb{P}_{\theta^\star}(\mathbb{P}_X(f(X) \in h(\theta^\star)) \geq 1 - \epsilon) \geq 1 - \delta$;
2. $\mathbb{P}_{\theta^\star}(\hat{r}(\theta^\star) \leq \bar{r}(\epsilon)) \geq 1 - \delta$.

*Proof.* Since the scenario problem is convex and $N \geq \frac{2}{\epsilon}\left(\log\frac{1}{\delta} + p\right)$, Theorem 1 of Campi, Garatti, and Prandini [26] gives that, with probability at least $1-\delta$, the solution $\theta^\star$ is feasible for the chance-constrained problem (4.5). Therefore, $\mathbb{P}_{\theta^\star}(\mathbb{P}_X(f(X) \in h(\theta^\star)) \geq 1-\epsilon) \geq 1-\delta$, which proves the first conclusion.

To prove the second conclusion, recall the law of total probability: for an arbitrary event $A \in \mathcal{F}_{\theta^\star}$ and an arbitrary partition $\{B_1, B_2\} \subseteq \mathcal{F}_{\theta^\star}$ of $\Omega_{\theta^\star}$ such that $\Omega_{\theta^\star} = B_1 \cup B_2$, $B_1 \cap B_2 = \emptyset$, and $\mathbb{P}_{\theta^\star}(B_i) > 0$ for $i \in \{1, 2\}$, we have that

$$\mathbb{P}_{\theta^\star}(A) = \mathbb{P}_{\theta^\star}(A|B_1)\mathbb{P}_{\theta^\star}(B_1) + \mathbb{P}_{\theta^\star}(A|B_2)\mathbb{P}_{\theta^\star}(B_2),$$

where $\mathbb{P}_{\theta^\star}(A|B_1) = \frac{\mathbb{P}_{\theta^\star}(A\cap B_1)}{\mathbb{P}_{\theta^\star}(B_1)}$ denotes the probability of event $A$ conditioned on event $B_1$, and similarly for $B_2$. Choose the particular events $A = \{\omega \in \Omega_{\theta^\star} : \hat{r}(\theta^\star(\omega)) \leq \bar{r}(\epsilon)\}$, $B_1 = \{\omega \in \Omega_{\theta^\star} : h(\theta^\star(\omega))$ is an $\epsilon$-cover of $\mathcal{Y}\}$, and $B_2 = \Omega_{\theta^\star} \setminus B_1$. Then, Proposition 14 shows that $B_1 \subseteq A$, so $\mathbb{P}_{\theta^\star}(A|B_1) = \frac{\mathbb{P}_{\theta^\star}(B_1)}{\mathbb{P}_{\theta^\star}(B_1)} = 1$. Furthermore, by the first conclusion proved above, $\mathbb{P}_{\theta^\star}(B_1) = \mathbb{P}_{\theta^\star}(\mathbb{P}_X(f(X) \in h(\theta^\star)) \geq 1-\epsilon) \geq 1-\delta$. Hence, the law of total probability gives that

$$\mathbb{P}_{\theta^\star}(A) \geq 1 - \delta + \mathbb{P}_{\theta^\star}(A|B_2)\mathbb{P}_{\theta^\star}(B_2) \geq 1 - \delta,$$

which proves the second conclusion. □

The conclusions of Theorem 11 assert that, with overwhelming probability, $h(\theta^\star)$ is an $\epsilon$-cover of $\mathcal{Y}$ and that the probabilistic robustness level is lower-bounded as $\hat{r}(\theta^\star) \leq \bar{r}(\epsilon)$. This gives high-probability guarantees for the simultaneous localization and safety certification of the output $Y = f(X)$.

In Theorem 11, randomness of a solution $\theta^\star$ to (4.6) is taken care of by the $1-\delta$ probability bound. In particular, $h(\theta^\star)$ may not actually be an $\epsilon$-cover, albeit with probability at most $\delta$. For this reason, we slightly abuse terminology and call $h(\theta^\star)$ the optimal $\epsilon$-cover. The additional layer of uncertainty embedded into the parameter $\delta$ is precisely the price paid for replacing the intractable chance-constrained problem (4.5) with the tractable scenario problem (4.6). However, Theorem 11 shows that the additional randomness is not an issue, since the requirement on $N$ scales as $\log \frac{1}{\delta}$. Therefore, we can select a small value for $\delta$ while maintaining a reasonable sample size $N$.

*Remark* 5. The guarantees in Theorem 11 are of the probably approximately correct (PAC) form. In the language of PAC learning, the surrogate output set $h(\theta^\star)$ is the hypothesis of the learner, which is selected from the concept class $\mathcal{H} = \{h(\theta) : \theta \in \Theta\}$. Theorem 11 asserts that the hypothesis is probably approximately correct, where *approximately correct* means the hypothesis (which is a set) contains the random output $Y = f(X)$ with probability at least $1-\epsilon$, and where *probably* means the hypothesis (which is selected based on the specific instances $x_1, x_2, \ldots, x_N$) is approximately correct (for general $X$) with probability at least $1-\delta$. Since this PAC guarantee holds whenever the scenario problem is convex, Theorem 10 gives sufficient conditions for the concept class $\mathcal{H}$ to be PAC learnable, and our proposed method can be viewed as learning robustness using the framework of PAC learning.

## 4.5 Exploiting Network Structure

In this section, we show how to exploit the structure of deep neural networks to reduce the time complexity of our method. The basic idea is to utilize adversarial bounds on the deep layers to replace $f$ with a shallower neural network, in effect developing a hybrid adversarial-probabilistic certification scheme. We assume that the network takes the form

$$f = \sigma^{(K)} \circ \mathcal{A}^{(K-1)} \cdots \circ \sigma^{(1)} \circ \mathcal{A}^{(0)},$$

where $\sigma^{(k)} \colon \mathbb{R}^{n_k} \to \mathbb{R}^{n_k}$ is the $k$th layer's activation function and $\mathcal{A}^{(k)} \colon \mathbb{R}^{n_k} \to \mathbb{R}^{n_{k+1}}$ is the affine map given by $\mathcal{A}^{(k)}(z) = W^{(k)} z + b^{(k)}$. Note that $n_0 = n_x$ and $n_K = n_y$.

Now, suppose that $f_L, f_U \colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ are two functions satisfying

$$f_L(x) \leq f(x) \leq f_U(x) \text{ for all } x \in \mathcal{X},$$

which are to be determined. Then, define the function $f' \colon \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ by

$$f'_i(x) = \begin{cases} (f_L(x))_i & \text{if } a_i \geq 0, \\ (f_U(x))_i & \text{if } a_i < 0, \end{cases}$$

for all $i \in \{1, 2, \ldots, n_y\}$ and all $x \in \mathbb{R}^{n_x}$. It is immediately clear that $a^\top f'(x)+b \leq a^\top f(x)+b$ for all $x \in \mathcal{X}$, so $f(x) \in \mathcal{S}$ for all $x \in \mathcal{X}$ such that $f'(x) \in \mathcal{S}$. This shows that

$$\mathbb{P}_X(f'(X) \in \mathcal{S}) \leq \mathbb{P}_X(f(X) \in \mathcal{S}).$$

Therefore, to certify the probabilistic robustness of $f$, it suffices to apply our certification procedure to the function $f'$. By bounding the deep layers' activations in $f$ by affine functions, we will reduce the problem to analyzing a simpler and shallower network $f'$ that allows for faster sampling of the outputs $y_j$. For notational simplicity, we let $\phi^{(k)} = \sigma^{(k)} \circ \mathcal{A}^{(k-1)} \circ \cdots \circ \sigma^{(1)} \circ \mathcal{A}^{(0)}$ for all $k \in \{1, 2, \ldots, K\}$, so that $\phi^{(k)}(x)$ is the activation at layer $k$ corresponding to the input $x$. Let $\phi^{(0)}$ be the identity map on $\mathbb{R}^{n_x}$. We now recall the notion of preactivation bounds, and make two assumptions.

**Definition 9.** A vector $l^{(k)} \in \mathbb{R}^{n_k}$ satisfying $l^{(k)} \leq \mathcal{A}^{(k-1)} \circ \phi^{(k-1)}(x)$ for all $x \in \mathcal{X}$ is called a *kth layer preactivation lower bound*. A vector $u^{(k)} \in \mathbb{R}^{n_k}$ satisfying $\mathcal{A}^{(k-1)} \circ \phi^{(k-1)}(x) \leq u^{(k)}$ for all $x \in \mathcal{X}$ is called a *kth layer preactivation upper bound*.

**Assumption 6.** For all $k \in \{1, 2, \ldots, K\}$, there exist $k$th layer preactivation lower and upper bounds $l^{(k)}$ and $u^{(k)}$, respectively.

**Assumption 7.** For all $k \in \{1, 2, \ldots, K\}$, there exist functions $\mathcal{L}^{(k)}, \mathcal{U}^{(k)} \colon \mathbb{R}^{n_k} \to \mathbb{R}^{n_k}$ given by

$$\mathcal{L}^{(k)}(z) = W_L^{(k)} z + b_L^{(k)}, \quad \mathcal{U}^{(k)}(z) = W_U^{(k)} z + b_U^{(k)},$$

that satisfy $\mathcal{L}^{(k)}(z) \leq \sigma^{(k)}(z) \leq \mathcal{U}^{(k)}(z)$ for all $z \in [l^{(k)}, u^{(k)}]$.

Definition 9, Assumption 6, and Assumption 7 are standard in the adversarial robustness literature. Notice that in many common architectures, $n_1 > n_0$ and the rank of $\mathcal{A}^{(0)}$ is $n_0$, and in this case Assumption 6 requires the input set $\mathcal{X}$ to be bounded. For most common activation functions and input sets, there exist a variety of methods for computing the above preactivation bounds and affine bounding functions—see, e.g., Zhang et al. [168].

The following lemma transforms our affine bounds on each activation function $\sigma^{(k)}$ into affine bounds relating the activation of one layer to the activation of the next layer.

**Lemma 6.** *Suppose that Assumption 6 and Assumption 7 hold. For all $k \in \{1, 2, \ldots, K\}$, it holds for all $x \in \mathcal{X}$ that*

$$\tilde{W}_L^{(k)} \phi^{(k-1)}(x) + \tilde{b}_L^{(k)} \leq \phi^{(k)}(x) \leq \tilde{W}_U^{(k)} \phi^{(k-1)}(x) + \tilde{b}_U^{(k)},$$

*where*

$$\begin{aligned} \tilde{W}_L^{(k)} &= W_L^{(k)} W^{(k-1)}, \quad \tilde{b}_L^{(k)} = W_L^{(k)} b^{(k-1)} + b_L^{(k)}, \\ \tilde{W}_U^{(k)} &= W_U^{(k)} W^{(k-1)}, \quad \tilde{b}_U^{(k)} = W_U^{(k)} b^{(k-1)} + b_U^{(k)}. \end{aligned} \tag{4.8}$$

*Proof.* Let $k \in \{1, 2, \ldots, K\}$ and let $x \in \mathcal{X}$. Define $z = \mathcal{A}^{(k-1)} \circ \phi^{(k-1)}(x)$. Then, since $z \in [l^{(k)}, u^{(k)}]$, it holds that $\mathcal{L}^{(k)}(z) \leq \sigma^{(k)}(z) \leq \mathcal{U}^{(k)}(z)$. Expanding this inequality using the matrix-vector representation of the affine maps $\mathcal{L}^{(k)}, \mathcal{U}^{(k)}, \mathcal{A}^{(k-1)}$, we obtain

$$\begin{aligned} W_L^{(k)}(W^{(k-1)}\phi^{(k-1)}(x) + b^{(k-1)}) + b_L^{(k)} &\leq \sigma^{(k)}(\mathcal{A}^{(k-1)} \circ \phi^{(k-1)}(x)) \\ &\leq W_U^{(k)}(W^{(k-1)}\phi^{(k-1)}(x) + b^{(k-1)}) + b_U^{(k)}, \end{aligned}$$

which gives the desired result upon substituting the definitions of $\tilde{W}_L^{(k)}, \tilde{W}_U^{(k)}, \tilde{b}_L^{(k)}, \tilde{b}_U^{(k)}$ and using the fact that $\sigma^{(k)}(\mathcal{A}^{(k-1)} \circ \phi^{(k-1)}(x)) = \phi^{(k)}(x)$. □

Next, we use the affine bounds between each neighboring layer in Lemma 6 to develop one overall affine bound relating the activation at some layer $k^\star$ to the output $\phi^{(K)}(x)$ of the neural network. The proof technique follows the idea developed in Weng et al. [148] and Zhang et al. [168], albeit allows for more general activation functions and allows us to "start" the affine bounding within the interior of the neural network architecture.

**Proposition 15.** *Suppose that Assumption 6 and Assumption 7 hold, and assume that $K \geq 3$. Let $k^\star \in \{1, 2, \ldots, K-2\}$ and define $M = K - k^\star$. Consider the matrices $\tilde{W}_L^{(k)}, \tilde{W}_U^{(k)}$ and vectors $\tilde{b}_L^{(k)}, \tilde{b}_U^{(k)}$ defined in (4.8). Define $E_1 = \tilde{W}_L^{(k^\star+1)}$, $F_1 = \tilde{b}_L^{(k^\star+1)}$, $G_1 = \tilde{W}_U^{(k^\star+1)}$, and $H_1 = \tilde{b}_U^{(k^\star+1)}$. Also, for $n \in \{2, 3, \ldots, M\}$, define*

$$\begin{aligned} E_n &= \min\{0, \tilde{W}_L^{(k^\star+n)}\}G_{n-1} + \max\{0, \tilde{W}_L^{(k^\star+n)}\}E_{n-1}, \\ F_n &= \min\{0, \tilde{W}_L^{(k^\star+n)}\}H_{n-1} + \max\{0, \tilde{W}_L^{(k^\star+n)}\}F_{n-1} + \tilde{b}_L^{(k^\star+n)}, \\ G_n &= \max\{0, \tilde{W}_U^{(k^\star+n)}\}G_{n-1} + \min\{0, \tilde{W}_U^{(k^\star+n)}\}E_{n-1}, \\ H_n &= \max\{0, \tilde{W}_U^{(k^\star+n)}\}H_{n-1} + \min\{0, \tilde{W}_U^{(k^\star+n)}\}F_{n-1} + \tilde{b}_U^{(k^\star+n)}. \end{aligned}$$

*Then, for all $x \in \mathcal{X}$, it holds that*

$$E_M\phi^{(k^\star)}(x) + F_M \leq \phi^{(K)}(x) \leq G_M\phi^{(k^\star)}(x) + H_M.$$

*Proof.* Let $x \in \mathcal{X}$. Then, by Lemma 6, it holds that

$$\tilde{W}_L^{(k)}\phi^{(k-1)}(x) + \tilde{b}_L^{(k)} \leq \phi^{(k)}(x) \leq \tilde{W}_U^{(k-1)}\phi^{(k-1)}(x) + \tilde{b}_U^{(k)} \tag{4.9}$$

for all $k \in \{1, 2, \ldots, K\}$. For all $n \in \{1, 2, \ldots, M+1\}$, define

$$x_n = \phi^{(k^\star+n-1)}(x).$$

Also, for all $n \in \{1, 2, \ldots, M\}$, define

$$A_n = \tilde{W}_L^{(k^\star+n)}, \quad B_n = \tilde{b}_L^{(k^\star+n)}, \quad C_n = \tilde{W}_U^{(k^\star+n)}, \quad D_n = \tilde{b}_U^{(k^\star+n)}.$$

Then it holds that $E_1 = A_1$, $F_1 = B_1$, $G_1 = C_1$, $H_1 = D_1$, and

$$\begin{aligned} E_n &= \min\{0, A_n\}G_{n-1} + \max\{0, A_n\}E_{n-1}, \\ F_n &= \min\{0, A_n\}H_{n-1} + \max\{0, A_n\}F_{n-1} + B_n, \\ G_n &= \max\{0, C_n\}G_{n-1} + \min\{0, C_n\}E_{n-1}, \\ H_n &= \max\{0, C_n\}H_{n-1} + \min\{0, C_n\}F_{n-1} + D_n, \end{aligned}$$

for $n \in \{2, 3, \ldots, M\}$. Also, (4.9) gives that

$$A_n x_n + B_n \leq x_{n+1} \leq C_n x_n + D_n$$

for all $n \in \{1, 2, \ldots, M\}$, so by Lemma 8, we conclude that

$$E_n x_1 + F_n \leq x_{n+1} \leq G_n x_1 + H_n$$

for all $n \in \{1, 2, \ldots, M\}$. In particular, for $n = M$, this yields the following bound on $x_{M+1} = \phi^{(K)}(x)$ in terms of $x_1 = \phi^{(k^\star)}(x)$:

$$E_M \phi^{(k^\star)}(x) + F_M \leq \phi^{(K)}(x) \leq G_M \phi^{(k^\star)}(x) + H_M,$$

which is the desired result. □

Since $\phi^{(K)}(x) = f(x)$, Proposition 15 shows that we may take the functions $f_L, f_U$ to be $f_L = \mathcal{A}_L \circ \phi^{(k^\star)}$ and $f_U = \mathcal{A}_U \circ \phi^{(k^\star)}$, where $\mathcal{A}_L(z) = E_M z + F_M$ and $\mathcal{A}_U(z) = G_M z + H_M$. In this case, our function $f'$ becomes

$$f'_i(x) = \begin{cases} \left(\mathcal{A}_L \circ \phi^{(k^\star)}(x)\right)_i & \text{if } a_i \geq 0, \\ \left(\mathcal{A}_U \circ \phi^{(k^\star)}(x)\right)_i & \text{if } a_i < 0. \end{cases}$$

This function $f'$ is a new neural network with the same first $k^\star < K$ nonlinear layers as $f$, and with one final affine transformation. Thus, a lower bound on the probabilistic robustness level of this shallow surrogate network $f'$ is also a lower bound on the probabilistic robustness level of the deep original network $f$.

When $k^\star$ is chosen to be small, the depth of this surrogate network is reduced, making it more efficient to sample outputs from it. As $k^\star$ increases, our method incorporates more of the underlying nonlinear nature of the network $f$ into the samples that we use to assess $f$'s robustness, meaning that the robustness certificate becomes tighter, but at the expense of increased sampling time. Specifically, in the common setting where every activation $\sigma^{(k)}$ is an element-wise operator with the time complexity $O(n_k)$, the time complexity of the sampling procedure for $f$ is $O(N(n_0 n_1 + n_1 n_2 + \cdots + n_{K-1} n_K))$, whereas the time complexity for $f'$ is $O(N(n_0 n_1 + n_1 n_2 + \cdots + n_{k^\star - 1} n_{k^\star} + n_{k^\star} n_K))$. If, for example, every number $n_k$ is of order $O(n)$, then $f$ would have the sampling time complexity $O(NKn^2)$, whereas $f'$ would be of order $O(Nk^\star n^2)$, giving a factor of $k^\star/K$ reduction in time complexity. As we will see in Section 4.6, this reduced time complexity is particularly helpful in deep neural network settings.

## 4.6 Numerical Simulations

### Illustrative Example

We consider the distributed linear system $x(t+1) = Ax(t)+Bu(t)$ for times $t \in \{0, 1, \ldots, T\}$, $T = 20$, as constructed in Gama and Sojoudi [59]. The system has $n = 10$ nodes, with a single state and input associated with every node; $x(t), u(t) \in \mathbb{R}^n$. The system and control matrices $A, B$ respect the underlying graph topology of the system, encoded by the support matrix $S$—see Gama and Sojoudi [59].

The control law is defined by a graph neural network:

$$u(t) = \Phi(x(t), S) \coloneqq \sum_{k=0}^{K-1} h_{k+1}^{(2)} S^k \sigma \left( \sum_{j=0}^{J-1} h_{j+1}^{(1)} S^j x(t) \right),$$

with $\sigma(\cdot) = \mathrm{ReLU}(\cdot)$, $K = J = 3$, $h^{(1)} = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$, and $h^{(2)} = (1, 1, 1)$. This neural network controller, defined in terms of $S$, respects the distributed nature of the system [59]. In this simulation, we consider the case where the graph support of the control law may be randomly perturbed, so that $u(t) = \Phi(x(t), S')$ for some $S' \in \mathbb{R}^{n\times n}$ with $S'_{ij} = XS_{ij}$, where $X$ is a Bernoulli random variable equal to 1 with probability 0.8; the controller loses an edge in its support graph with probability 0.2. We fix a (normal random) initial condition $x(0) \in \mathbb{R}^n$, and we consider the map $f \colon \mathbb{R}^{n\times n} \to \mathbb{R}^2$ given by $f(S') = (x_1(T), x_2(T))$, where $x(T)$ is the terminal state of the system under the control law given by $u(t) = \Phi(x(t), S')$. The safe set is defined by $\mathcal{S}_1 = \{y \in \mathbb{R}^2 : a^\top y + b \geq 0\}$, where $a = (1, 0)$ and $b = 0.05$. We seek to certify that the first two elements of the (random) terminal state are safe even under the perturbed control support $S'$, i.e., that $f(S') \in \mathcal{S}_1$.

The norm ball class $\mathcal{H}$ of Example 3, Example 4, and Example 5 is employed with $\|\cdot\|$ being the $\ell_2$-norm, and with probability levels $\epsilon = 0.05$ and $\delta = 10^{-5}$. We set $N = \left\lceil \frac{2}{\epsilon}(\log \frac{1}{\delta} + p) \right\rceil = 581$, then sample $N$ inputs $S'_j$ and compute their corresponding outputs $f(S'_j)$ by running the system. As shown in Example 4, $h(\cdot)$ is an affine set-valued function, and therefore $\Theta$ and $h(\cdot)$ satisfy the conditions of Theorem 10. We choose the regularizer for the scenario problem (4.7) to be the square of the norm ball radius, i.e., $v(\bar{y}, r) = r^2$. The optimization problem is convex as guaranteed by Theorem 10. We solve the scenario problem first without regularization, and then with two different levels of regularization: $\lambda_1 = 1$ and $\lambda_2 = 100$. The respective solutions are denoted by $\theta^\star$, $\theta^\star_{\lambda_1}$, and $\theta^\star_{\lambda_2}$. Each instance takes approximately 15 seconds to solve using CVX in MATLAB on a standard laptop with a 2.6 GHz dual-core i5 processor. The resulting approximate robustness levels are $\hat{r}(\theta^\star) = 0.0058$, $\hat{r}(\theta^\star_{\lambda_1}) = 0.0054$, and $\hat{r}(\theta^\star_{\lambda_2}) = -0.0061$. In the instances without regularization and with regularization level $\lambda_1$, Theorem 11 guarantees that the perturbed terminal state $(x_1(T), x_2(T))$ has a safety level of 0.005 with our prescribed high probability, granting the probablistic robustness certificate we seek. On the other hand, since $\hat{r}(\theta^\star_{\lambda_2}) < 0$, the scenario problem using regularization level $\lambda_2$ is not able to certify the safety of the terminal state. This is due to the inherent tradeoff between localization and certification, which we now discuss.

The optimal $\epsilon$-covers $h(\theta^\star)$, $h(\theta^\star_{\lambda_1})$, and $h(\theta^\star_{\lambda_2})$ are shown in Figure 4.1. The unregularized set $h(\theta^\star)$ is massively over-conservative due to the choice $\lambda = 0$, which corresponds to pure robustness certification. Indeed, $h(\theta^\star)$ is the $\epsilon$-cover from our class of sets that is furthest from the boundary of the safe set, making $\hat{r}(\theta^\star)$ the tightest lower bound on $\bar{r}(\epsilon)$. On the other hand, the optimal $\epsilon$-covers using $\lambda = \lambda_1$ and $\lambda = \lambda_2$ are seen to give tighter localizations of the terminal state $(x_1(T), x_2(T))$. The approximate robustness level using regularization $\lambda_1$ is only slightly lower than the unregularized value, but the regularization $\lambda_2$ is large enough to cause the approximate robustness level $\hat{r}(\theta^\star_{\lambda_2})$ to become negative at the expense of localization. This shows how overemphasizing localization may harm the certification aspect of robustness assessment, and empirically demonstrates why output set estimation methods may not be adequate for issuing robustness certificates. This is explored further in our comparison to output set estimation below.

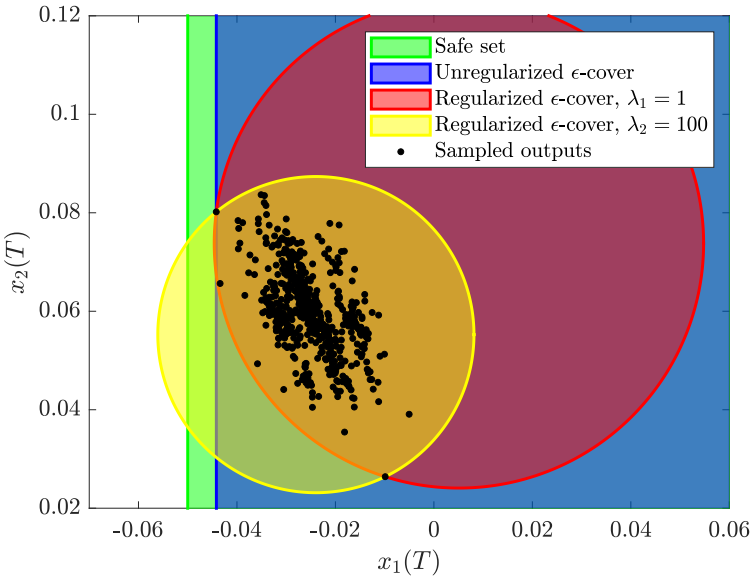



Figure 4.1: Optimal $\ell_2$-norm ball $\epsilon$-covers for safe set $\mathcal{S}_1$.

We repeat the simulation with the more complicated safe set $\mathcal{S}_2 = \{y \in \mathbb{R}^2 : Ay + b \geq 0\}$, where $A = \left[\begin{smallmatrix} 1 & 0 \\ -1 & 0 \end{smallmatrix}\right]$ and $b = (0.05, 0)$, applying our method to each row of $\mathcal{S}_2$ individually. To do so, we set $\epsilon' = \epsilon/2$, $\delta' = \delta/2$, and $N' = \left\lceil \frac{2}{\epsilon'} \left( \log \frac{1}{\delta'} + p \right) \right\rceil = 1217$. For each of the two half-spaces defining $\mathcal{S}_2$, we solve the scenario problem using $N'$ independent and identically distributed samples, and then intersect the two resulting $\epsilon'$-covers. Doing so, we obtain an $\epsilon$-cover with probability at least $1 - \delta$. We repeat this process again using regularization levels $\lambda_1 = 1$ and $\lambda_2 = 100$, and we find that each scenario problem takes approximately 30 seconds to solve. As seen in Figure 4.2, some samples may reside outside the resulting intersection $\epsilon$-covers—this is valid, and the robustness certificates still hold.

Again, we find robustness certificates for $\lambda = 0$ and $\lambda = \lambda_1$. However, for $\lambda = \lambda_2$, the optimal $\epsilon$-covers corresponding to both half-spaces are found to intersect the unsafe region

of the state space, due to the increased emphasis on localization. Interestingly, the overall localization after intersecting the two $\epsilon'$-covers for $\lambda = \lambda_2$ is in a sense looser than that of the case $\lambda = \lambda_1$, indicating that moderate regularization levels, like $\lambda_1$ in this simulation, may simultaneously perform best for both localization and certification in the case of safe sets defined by more than one half-space. Optimizing $\lambda$ in general poses an interesting problem for future research.

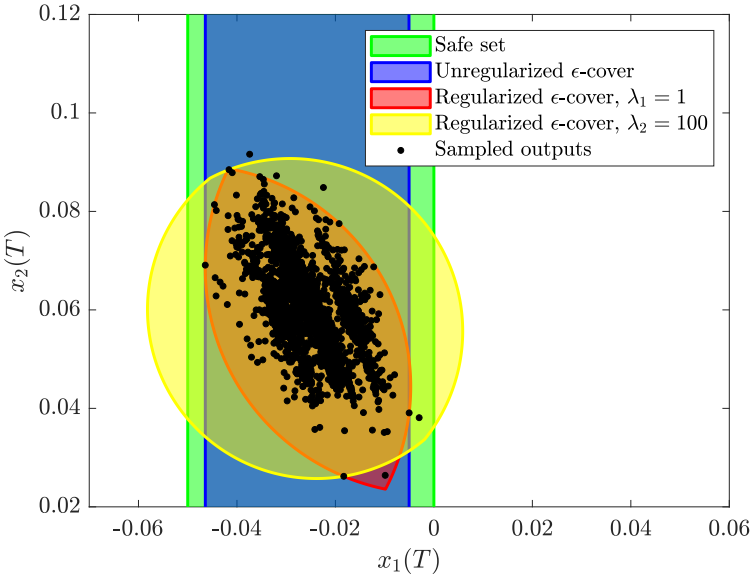



Figure 4.2: Optimal intersections of two $\ell_2$-norm ball $\frac{\epsilon}{2}$-covers for safe set $\mathcal{S}_2$.

## Comparison to Output Set Estimation

In this example, we compare our proposed method to an alternate approach. In the second approach, we first estimate the output set of the neural network using the scenario-based reachability analysis in Devonport and Arcak [40]. We then use the resulting output set estimate to assess robustness. Recall that our proposed scenario optimization (4.6) generalizes the reachability analysis of Devonport and Arcak [40]. In addition to localizing the network outputs, our approach directly takes the goal of robustness certification into account, whereas the estimation technique of Devonport and Arcak [40] does not.

To illustrate our comparison, consider a simple ReLU neural network given by $f\colon \mathbb{R}^2 \to \mathbb{R}^2$, where $f_i(x) = \max\{0, x_i\}$ for $i \in \{1, 2\}$. The noisy input $X$ is distributed uniformly on the input set $\mathcal{X} = \{x \in \mathbb{R}^2 : \|x - \bar{x}\|_1 \leq 1\}$, where $\bar{x} = (1, 0)$. The safe set is given as $\mathcal{S} = \{y \in \mathbb{R}^2 : a^\top y + b \geq 0\}$, where $a = (0, 1)$ and $b = 0.5$. It is straightforward to show that the output set is the top-half of the input set, namely, $\mathcal{Y} = \mathcal{X} \cap \{y \in \mathbb{R}^2 : y_2 \geq 0\}$. Hence, if $y \in \mathcal{Y}$ then $a^\top y + b = y_2 + b \geq b \geq 0$. Therefore, $\mathcal{Y} \subseteq \mathcal{S}$, and so the random output $Y = f(X)$ is safe with probability one.

We now perform the two assessments at hand, computing our proposed solution first. We choose the $\ell_2$-norm ball class for our candidate $\epsilon$-covers and draw sufficiently many output samples $\{y_j\}_{j=1}^N$ according to Theorem 11 with $\epsilon = 0.1$ and $\delta = 10^{-5}$. Next, we choose the regularizer $v(\bar{y}, r) = r^2$ with $\lambda = 0.1$ and solve the scenario problem (4.7) for the $\ell_2$-norm ball class. The solution correctly certifies that network outputs are safe with high probability; see the blue set in Figure 4.3.

We now turn to the alternative method. We use the same $\ell_2$-norm ball class as above and solve for the minimum volume $\epsilon$-cover using the same $N$ sampled outputs. The estimated output set is shown in red in Figure 4.3. Despite being a tighter localization, a substantial portion of the estimated output set exits the safe set, meaning that this approach cannot certify the robustness of the network, even though the random output is truly safe with probability one. This comparison shows that a good estimate of the output set may not be the most informative set to use for assessing output safety. This observation endorses our proposed method, which simultaneously encodes both goals of certification and localization.

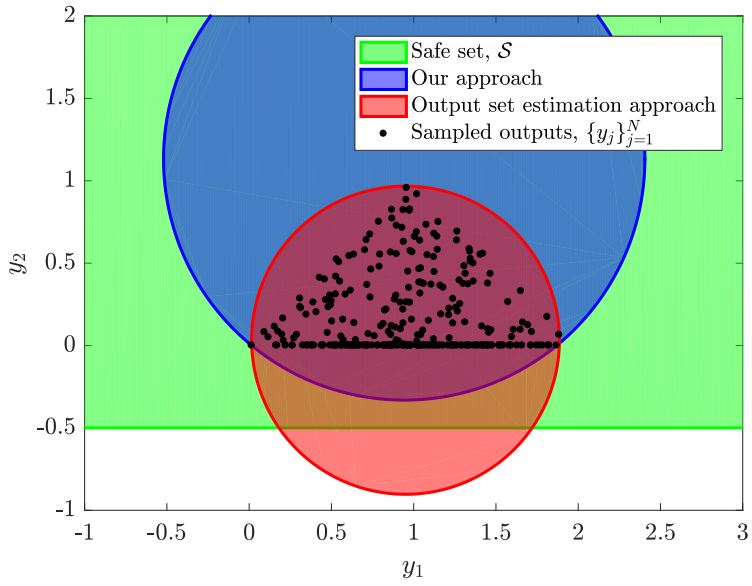



Figure 4.3: The tightest $\epsilon$-cover of the output set (red) does not correctly certify robustness. Our approach (blue) correctly certifies robustness and maintains reasonable localization.

## Comparison to PROVEN

In this simulation, we compare our approach using the half-space class $\mathcal{H} = \big\{\{y \in \mathbb{R}^{n_y} : c^\top y + d \geq 0\} : (c, d) \in \mathbb{R}^{n_y} \times \mathbb{R}\big\}$, for which we solve the scenario problem using its closed-form solution (see Appendix 4.D), to the state-of-the-art algorithm, PROVEN [147], for assessing robustness against random input noise. Throughout, we use open-source neural

Table 4.1: Average probabilistic robustness level lower bounds $\hat{r}(\theta^\star)$ for MNIST ReLU networks subject to uniform noise over $\ell_\infty$-norm ball. All values are averaged over 10 nominal inputs with randomly chosen target classes $i$. Lower bounds giving certified robustness (on average) are bolded, and the average certified adversarial radii computed using Zhang et al. [168] are italicized.

(a) $2 \times [20]$ network.

| Radius | $\epsilon = 0.001$ | | $\epsilon = 0.1$ | | $\epsilon = 0.25$ | |
|---|---|---|---|---|---|---|
| | PRVN | Ours | PRVN | Ours | PRVN | Ours |
| 0.01 | **24.51** | **14.11** | **24.79** | **14.26** | **24.88** | **14.28** |
| | 1.491 s | 0.605 s | 1.405 s | 0.004 s | 1.370 s | 0.003 s |
| *0.027* | **14.71** | **13.36** | **15.45** | **13.77** | **15.68** | **13.85** |
| | 1.434 s | 0.599 s | 1.540 s | 0.004 s | 1.427 s | 0.003 s |
| 0.05 | −1.33 | **12.34** | **0.02** | **13.11** | **0.44** | **13.26** |
| | 1.511 s | 0.597 s | 1.468 s | 0.004 s | 1.423 s | 0.003 s |
| 0.1 | −42.09 | **10.25** | −39.43 | **11.66** | −38.61 | **12.00** |
| | 1.485 s | 0.623 s | 1.437 s | 0.004 s | 1.437 s | 0.002 s |
| 0.5 | −404.42 | −7.21 | −391.05 | −0.05 | −386.96 | **1.71** |
| | 1.525 s | 0.645 s | 1.472 s | 0.004 s | 1.432 s | 0.002 s |

(b) $3 \times [20]$ network.

| Radius | $\epsilon = 0.001$ | | $\epsilon = 0.1$ | | $\epsilon = 0.25$ | |
|---|---|---|---|---|---|---|
| | PRVN | Ours | PRVN | Ours | PRVN | Ours |
| 0.01 | **29.24** | **17.28** | **29.59** | **17.45** | **29.70** | **17.49** |
| | 1.416 s | 0.380 s | 1.345 s | 0.003 s | 1.388 s | 0.001 s |
| *0.022* | **18.80** | **16.65** | **19.49** | **17.02** | **19.71** | **17.10** |
| | 1.382 s | 0.362 s | 1.345 s | 0.003 s | 1.364 s | 0.001 s |
| 0.05 | −22.31 | **15.19** | −20.67 | **16.00** | −20.17 | **16.19** |
| | 1.377 s | 0.345 s | 1.325 s | 0.003 s | 1.374 s | 0.001 s |
| 0.1 | −114.83 | **12.57** | −111.59 | **14.19** | −110.60 | **14.58** |
| | 1.351 s | 0.372 s | 1.343 s | 0.003 s | 1.340 s | 0.002 s |
| 0.5 | −866.28 | −9.36 | −857.82 | −0.55 | −855.22 | **0.36** |
| | 1.385 s | 0.368 s | 1.351 s | 0.003 s | 1.336 s | 0.003 s |

(c) $2 \times [1024]$ network.

| Radius | $\epsilon = 0.001$ | | $\epsilon = 0.1$ | | $\epsilon = 0.25$ | |
|---|---|---|---|---|---|---|
| | PRVN | Ours | PRVN | Ours | PRVN | Ours |
| 0.01 | **51.07** | **27.73** | **51.40** | **27.87** | **51.50** | **27.93** |
| | 0.899 s | 1.102 s | 0.877 s | 0.008 s | 0.874 s | 0.004 s |
| *0.032* | **30.34** | **26.69** | **31.37** | **27.13** | **31.69** | **27.32** |
| | 0.869 s | 1.202 s | 0.858 s | 0.008 s | 0.846 s | 0.004 s |
| 0.05 | **6.95** | **25.83** | **8.59** | **26.53** | **9.09** | **26.82** |
| | 0.846 s | 1.125 s | 0.851 s | 0.008 s | 0.844 s | 0.004 s |
| 0.1 | −77.53 | **23.46** | −74.13 | **24.84** | −73.09 | **25.42** |
| | 0.861 s | 1.137 s | 0.854 s | 0.008 s | 0.881 s | 0.004 s |
| 0.5 | −914.36 | **4.83** | −900.22 | **11.79** | −895.89 | **14.45** |
| | 0.869 s | 1.159 s | 0.883 s | 0.008 s | 0.868 s | 0.004 s |

(d) $3 \times [1024]$ network.

| Radius | $\epsilon = 0.001$ | | $\epsilon = 0.1$ | | $\epsilon = 0.25$ | |
|---|---|---|---|---|---|---|
| | PRVN | Ours | PRVN | Ours | PRVN | Ours |
| 0.01 | **68.87** | **36.86** | **69.28** | **37.06** | **69.41** | **37.12** |
| | 2.535 s | 1.782 s | 2.382 s | 0.015 s | 2.464 s | 0.009 s |
| *0.024* | **44.14** | **35.97** | **45.18** | **36.44** | **45.50** | **36.58** |
| | 2.434 s | 2.026 s | 2.448 s | 0.013 s | 2.510 s | 0.008 s |
| 0.05 | −111.09 | **34.32** | −108.25 | **35.32** | −107.39 | **35.59** |
| | 2.739 s | 2.258 s | 2.671 s | 0.013 s | 2.761 s | 0.007 s |
| 0.1 | −729.24 | **31.10** | −723.45 | **33.10** | −721.68 | **33.69** |
| | 3.081 s | 2.325 s | 2.916 s | 0.014 s | 2.912 s | 0.007 s |
| 0.5 | −6872.3 | **6.89** | −6849.5 | **15.85** | −6842.5 | **18.56** |
| | 2.877 s | 1.955 s | 2.996 s | 0.014 s | 3.012 s | 0.007 s |

network models provided in Weng et al. [147]. The underlying framework of PROVEN relies on bounding a classifier's margin function by affine functions. PROVEN uses the affine functions to give closed-form bounds on the misclassification probability. We remark that, since PROVEN does not rely on sampling, their lower bound on $\bar{r}(\epsilon)$ is deterministic, whereas our bound holds with probability $1 - \delta$, which is taken to be $1 - 10^{-5} = 0.99999$ in this simulation. The results in this section are computed using TensorFlow in Python on a standard laptop with a 2.6 GHz dual-core i5 processor.

We first consider a variety of pretrained MNIST digit classification networks with ReLU activation functions [84]. A network model with $m$ hidden layers, each having $n$ neurons, is denoted by $m \times [n]$. We model the noisy input $X$ as being distributed uniformly on $\mathcal{X} = \{x \in \mathbb{R}^{n_x} : \|x - \bar{x}\|_\infty \leq \epsilon_x\}$. For 10 randomly selected nominal inputs $\bar{x}$, we compute a lower bound $\hat{r}(\theta^\star)$ on the probabilistic robustness level $\bar{r}(\epsilon)$. The robustness level of a network (for a particular pair $(\epsilon, \epsilon_x)$) is evaluated by computing the average robustness level

Table 4.2: Average probabilistic robustness level lower bounds $\hat{r}(\theta^\star)$ for various other models. Values for Models 1 and 2 are averaged over 10 inputs, and for Model 3 they are averaged over 100 network realizations. Lower bounds giving certified robustness (on average) are bolded, and the average certified adversarial radii computed using Zhang et al. [168] are italicized.

| Model 1 | | | Model 2 | | | Model 3 | | |
|---|---|---|---|---|---|---|---|---|
| Radius | PRVN | Ours | Radius | PRVN | Ours | Radius | PRVN | Ours |
| 0.005 | **7.81** | **19.01** | 0.001 | **48.80** | **33.26** | 0.01 | **1.96** | **1.97** |
| *0.0068* | **2.20** | **18.96** | *0.0023* | **10.90** | **33.17** | 0.05 | **1.71** | **1.79** |
| 0.01 | −26.31 | **18.88** | 0.003 | −91.20 | **33.12** | 0.1 | **1.40** | **1.56** |
| 0.05 | −1769.97 | **17.83** | 0.005 | −1056.85 | **32.98** | 0.5 | −1.06 | −0.29 |
| 0.1 | −4493.02 | **16.48** | 0.01 | −8717.05 | **32.62** | 1.0 | −4.13 | −2.60 |

lower bound across the 10 inputs.[5] This is done for probability levels $\epsilon \in \{0.001, 0.1, 0.25\}$ (with corresponding sample sizes $N \in \{25026, 251, 101\}$) and for a variety of noise levels $\epsilon_x$. We include the certified adversarial radius computed using Zhang et al. [168], which is a lower bound on the smallest radius such that $\mathcal{X}$ contains an input that yields an unsafe output. The targeted class $i$, which defines the margin function $g_i$ relative to the nominal input's true class $i^\star$, is randomly chosen for each input tested. See Example 1 and Example 2 for more information on this application. The average lower bound values computed using our approach (denoted Ours) and PROVEN's (denoted PRVN) are shown in Table 4.1.

As seen in Table 4.1, our method is able to certify larger input sets than PROVEN for every network tested. Although PROVEN's lower bound is tighter for small radii, at large radii our bound is significantly tighter than PROVEN's, particularly for the larger networks in Table 4.1c and Table 4.1d. This indicates that our method is especially powerful for certifying deep neural networks. The end-to-end affine bounding scheme in PROVEN tends to become looser as the network becomes deeper and as the input set becomes larger [147]. The technique comes from the adversarial robustness literature, and therefore it being embedded into PROVEN is likely the reason why PROVEN fails for radii larger than the certified adversarial radius. Our method bypasses this preliminary bound altogether. We also remark that our method certifies much larger input set radii (sometimes up to 20 times larger) compared to the certified adversarial radii (italicized) computed using the state-of-

[5]Despite $\bar{r}(\epsilon)$ being an input-specific quantity, we follow the literature's standard practice and average our robustness metric over a collection of test inputs. This standard was popularized in Szegedy et al. [131], where model robustness is evaluated using average certified input set radii. Our average robustness level lower bound immediately gives an average certified input set radius when the bound is nonnegative. In the probabilistic setting, it can be more natural to evaluate models in terms of misclassification probability, like our bounds do, instead of in terms of certified input set radii, see, e.g., Zakrzewski [162], Webb et al. [146], Fazlyab, Morari, and Pappas [49], and Couellan [36].

the-art worst-case analysis Zhang et al. [168]. The exact minimum adversarial radii (averaged across the 10 inputs) for the $2 \times [20]$ and $3 \times [20]$ ReLU networks are efficiently computed to be around 0.07 using mixed-integer linear programming [135]. With the tolerance $\epsilon = 0.001$, our method certifies radii over 0.1 for these networks. This evidences the claim that worst-case approaches, including exact ones, are over-conservative when applied to settings where a small amount of risk may be tolerable, in effect justifying our data-driven framework.

In Table 4.2, we repeat the simulation using three variants in the neural network model. Model 1 is an MNIST classifier with $\tanh(\cdot)$ activation functions of size $4 \times [1024]$. On the other hand, Model 2 is a CIFAR-10 network with ReLU activations of size $5 \times [2048]$. We see that both Model 1 and Model 2 exhibit the same behavior as before; for small input set radii, the lower bounds provided by PROVEN and our method are similar and both yield high-probability robustness certificates. For larger radii, our lower bound significantly outperforms PROVEN's. Since the affine bounds in PROVEN are relatively tight for small input sets radii, we suspect the PROVEN bound to closer match our bound for large input set radii in the special case of linear classification networks.

Model 3 is a linear classifier, i.e., of the form $f(x) = Wx + b$, with 50 inputs, 10 outputs, and weights, biases, and nominal input all chosen randomly with elements uniform on $[0, 1]$. We computed lower bounds on $\bar{r}(\epsilon)$ for 100 such models and averaged the results. Table 4.2 shows that indeed the PROVEN bound closely matches our bound for every radius tested in this special case, and that the two methods succeed and fail to issue robustness certificates simultaneously. These results show that the worst-case bounding techniques used in the adversarial robustness literature may work satisfactorily for simple models with random inputs, such as linear classifiers, but that these bounds are too loose for general nonlinear networks.

## Exploiting Network Structure

In this simulation, we implement the complexity-reducing method of Section 4.5. We consider networks with 10 inputs, 10 outputs, and 250 neurons in every hidden layer. The number of layers $K$ varies from 3 to 25. The weights and biases for every architecture are chosen randomly (with Gaussian elements, then normalized). Every activation function $\sigma^{(k)}$ is chosen to be ReLU, with preactivation and affine bounds derived according to Weng et al. [148]. We consider (randomly chosen Gaussian) clean inputs $\bar{x}$ with uniform additive random noise on the $\ell_\infty$-norm ball with radius $\epsilon_x = 0.1$, so that the noisy inputs $X$ are distributed uniformly on $\{x \in \mathbb{R}^{n_x} : \|x - \bar{x}\|_\infty \leq \epsilon_x\}$.

For every architecture, we lower-bound the probabilistic robustness level for 50 different realizations of the weights, biases, and inputs, where for each realization we solve the scenario optimization problem using the class $\mathcal{H} = \left\{\{y \in \mathbb{R}^{n_y} : c^\top y + d \geq 0\} : (c, d) \in \mathbb{R}^{n_y} \times \mathbb{R}\right\}$ of half-spaces with $N = 1000$ sampled inputs. This is done both using our baseline methodology, maintaining the full nonlinearity of each deep network, as well as using the shallow surrogate networks proposed in Section 4.5. Figure 4.4 displays the ratio $T_{f'}/T_f$ between the sampling time $T_{f'}$ (averaged over all realizations of a given depth) for the shallow

surrogate network $f'$ and the sampling time $T_f$ (again, averaged) for the deep network $f$. We see that, when $k^\star = O(K)$, meaning that the majority of nonlinearity is maintained in $f'$, the sampling times remain roughly the same. On the other hand, when $k^\star = O(1)$, meaning the majority of nonlinearity is replaced by affine bounds, the sampling time is reduced by nearly two orders of magnitude, and the reduction follows the expected rate of $k^\star/K = O(1/K)$. For in-between surrogate architectures using $k^\star = O(\log K)$ and $k^\star = O(\sqrt{K})$, we find respectable time complexity reductions, nearing an order of magnitude decrease in sampling time. The decreases in the lower bound on the probabilistic robustness level are also shown in Figure 4.4. The average lower bound $r_f$ without exploiting structure is 0.1. Therefore, the degradation of the bound incurred by using the shallow surrogate networks is relatively constant and minimal. The simulation results in the same conclusions when using tanh activation functions, and when using smaller and larger input set radii $\epsilon_x$.

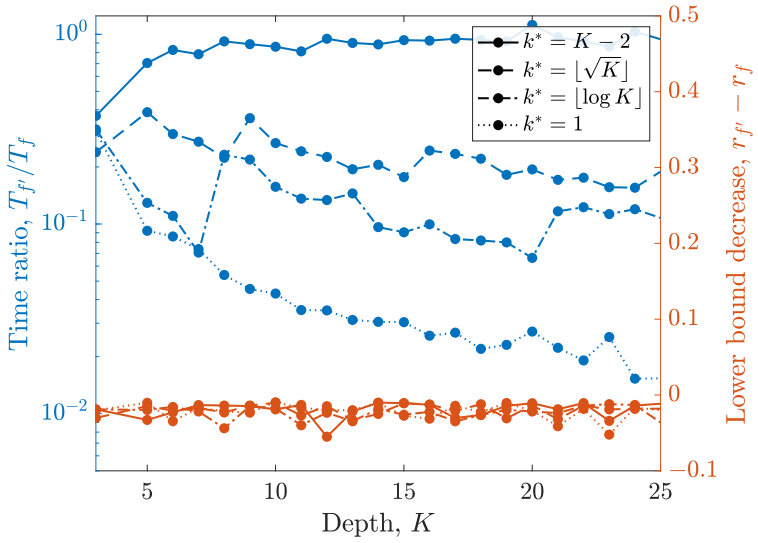



Figure 4.4: Ratio between the average sampling time of the shallow surrogate network $f'$ and that of the deep original network $f$, and the corresponding decrease in the lower bound on the probabilistic robustness level.

## 4.7 Conclusions

In this chapter, we propose a data-driven method for certifying the robustness of neural networks against random input noise. Sufficient conditions are developed for the convexity of the resulting optimization, as well as on the number of samples to issue a high-probability guarantee for the safety of the output. The method applies to general neural networks and general input noise distributions. In cases where the activation functions can be affinely bounded, we show how to exploit the network structure to reduce sample complexity. The

unified framework allows the user to balance the strength of the robustness bound with the tightness of the resulting output set estimate. Our numerical simulations show that the proposed method gives less conservative robustness bounds than the prior state-of-the-art techniques, as it is capable of certifying larger input uncertainty regions on synthetic, MNIST, and CIFAR-10 networks. In situations where neural network failure modes may exist but are unlikely and hence robustness amounts to achieving tolerable risk, these results suggest that re-tooling worst-case analysis techniques from the adversarial robustness literature results in overly conservative bounds. We conclude that taking a data-driven approach to generate probabilistic robustness guarantees, as developed in this chapter, is the better option in these contexts.

# Appendices

## 4.A Supporting Lemmas

**Lemma 7.** *Suppose that* $A_1x_1 + B_1 \leq x_2 \leq C_1x_1 + D_1$ *and* $A_2x_2 + B_2 \leq x_3 \leq C_2x_2 + D_2$ *for vectors* $x_k, B_k, D_k$ *and matrices* $A_k, C_k$*, all of compatible dimensions. Then*

$$E_2x_1 + F_2 \leq x_3 \leq G_2x_1 + H_2,$$

*where*

$$\begin{aligned}
E_2 &= \min\{0, A_2\}C_1 + \max\{0, A_2\}A_1, \\
F_2 &= \min\{0, A_2\}D_1 + \max\{0, A_2\}B_1 + B_2, \\
G_2 &= \max\{0, C_2\}C_1 + \min\{0, C_2\}A_1, \\
H_2 &= \max\{0, C_2\}D_1 + \min\{0, C_2\}B_1 + D_2.
\end{aligned}$$

*Proof.* Let $(z)_i$ denote the $i$th element of a vector $z$ and $(Z)_{ij}$ denote the $(i,j)$th element of a matrix $Z$. It holds for all indices $i$ that

$$\begin{aligned}
(x_3)_i &\leq (C_2x_2 + D_2)_i \\
&= \sum_j (C_2)_{ij}(x_2)_j + (D_2)_i \\
&= \sum_{j:(C_2)_{ij}\geq 0} (C_2)_{ij}(x_2)_j + \sum_{j:(C_2)_{ij}<0} (C_2)_{ij}(x_2)_j + (D_2)_i \\
&\leq \sum_{j:(C_2)_{ij}\geq 0} (C_2)_{ij}(C_1x_1 + D_1)_j + \sum_{j:(C_2)_{ij}<0} (C_2)_{ij}(A_1x_1 + B_1)_j + (D_2)_i \\
&= \sum_j (\max\{0, (C_2)_{ij}\}(C_1x_1 + D_1)_j + \min\{0, (C_2)_{ij}\}(A_1x_1 + B_1)_j) + (D_2)_i \\
&= \big(\max\{0, C_2\}(C_1x_1 + D_1) + \min\{0, C_2\}(A_1x_1 + B_1) + D_2\big)_i,
\end{aligned}$$

so

$$x_3 \leq (\max\{0, C_2\}C_1 + \min\{0, C_2\}A_1)x_1 + \max\{0, C_2\}D_1 + \min\{0, C_2\}B_1 + D_2 = G_2x_1 + H_2,$$

which proves the upper bound on $x_3$.

To prove the lower bound on $x_3$, note that $-x_3 \leq (-A_2)x_2 + (-B_2)$, so the above analysis yields that

$$\begin{aligned} -x_3 &\leq (\max\{0, -A_2\}C_1 + \min\{0, -A_2\}A_1)x_1 + \max\{0, -A_2\}D_1 + \min\{0, -A_2\}B_1 - B_2 \\ &= -(\min\{0, A_2\}C_1 + \max\{0, A_2\}A_1)x_1 - (\min\{0, A_2\}D_1 + \max\{0, A_2\}B_1 + B_2) \\ &= -E_2x_1 - F_2, \end{aligned}$$

which concludes the proof. □

**Lemma 8.** *Let $M \in \mathbb{N}$, $M > 1$. Suppose that*

$$A_n x_n + B_n \leq x_{n+1} \leq C_n x_n + D_n$$

*for all $n \in \{1, 2, \ldots, M\}$, where the vectors $x_n, B_n, D_n$ and the matrices $A_n, C_n$ are all of compatible dimensions. For all $n \in \{2, 3, \ldots, M\}$, define*

$$\begin{aligned} E_n &= \min\{0, A_n\}G_{n-1} + \max\{0, A_n\}E_{n-1}, \\ F_n &= \min\{0, A_n\}H_{n-1} + \max\{0, A_n\}F_{n-1} + B_n, \\ G_n &= \max\{0, C_n\}G_{n-1} + \min\{0, C_n\}E_{n-1}, \\ H_n &= \max\{0, C_n\}H_{n-1} + \min\{0, C_n\}F_{n-1} + D_n, \end{aligned}$$

*where $E_1 = A_1$, $F_1 = B_1$, $G_1 = C_1$, and $H_1 = D_1$. Then*

$$E_n x_1 + F_n \leq x_{n+1} \leq G_n x_1 + H_n \tag{4.10}$$

*holds for all $n \in \{1, 2, \ldots, M\}$.*

*Proof.* The result holds for $n = 1$ by assumption. We prove the result for $n \in \{2, 3, \ldots, M\}$ by induction on $n$. Lemma 7 shows that the result holds for the base case $n = 2$. Now, suppose that the result holds for some arbitrary $n \in \{2, 3, \ldots, M-1\}$, so that

$$E_n x_1 + F_n \leq x_{n+1} \leq G_n x_1 + H_n. \tag{4.11}$$

Make the following definitions:

$$\begin{gathered} x_1' = x_1, \quad x_2' = x_{n+1}, \quad x_3' = x_{n+2}, \\ A_1' = E_n, \quad B_1' = F_n, \quad C_1' = G_n, \quad D_1' = H_n, \\ A_2' = A_{n+1}, \quad B_2' = B_{n+1}, \quad C_2' = C_{n+1}, \quad D_2' = D_{n+1}. \end{gathered}$$

Then by the assumption that $A_{n+1}x_{n+1} + B_{n+1} \leq x_{n+2} \leq C_{n+1}x_{n+1} + D_{n+1}$ it holds that

$$A_2' x_2' + B_2' \leq x_3' \leq C_2' x_2' + D_2'. \tag{4.12}$$

Also, by the induction hypothesis (4.11), it holds that

$$A_1' x_1' + B_1' \leq x_2' \leq C_1' x_1' + D_1'. \tag{4.13}$$

Therefore, (4.12) and (4.13) together with Lemma 7 give that

$$E_2' x_1' + F_2' \leq x_3' \leq G_2' x_1' + H_2', \tag{4.14}$$

where

$$\begin{aligned} E_2' &= \min\{0, A_2'\} C_1' + \max\{0, A_2'\} A_1', \\ F_2' &= \min\{0, A_2'\} D_1' + \max\{0, A_2'\} B_1' + B_2', \\ G_2' &= \max\{0, C_2'\} C_1' + \min\{0, C_2'\} A_1', \\ H_2' &= \max\{0, C_2'\} D_1' + \min\{0, C_2'\} B_1' + D_2'. \end{aligned}$$

Substituting our earlier definitions for these values gives that $E_2' = E_{n+1}$, $F_2' = F_{n+1}$, $G_2' = G_{n+1}$, and $H_2' = H_{n+1}$, and therefore in light of the fact that $x_1' = x_1$ and $x_3' = x_{n+2}$, (4.14) becomes

$$E_{n+1} x_1 + F_{n+1} \leq x_{n+2} \leq G_{n+1} x_1 + H_{n+1},$$

so the induction step has been proven. Thus, the result (4.10) holds for all $n \in \{1, 2, \ldots, M\}$. □

## 4.B Extension to General Polyhedral Safe Sets

In this section, we explicitly walk through the steps of generalizing our proposed assessment method to the case where the safe set is a general polyhedral set defined by the intersection of finitely many half-spaces.

Consider the polyhedral safe set $\mathcal{S} = \{y \in \mathbb{R}^{n_y} : Ay + b \geq 0\}$, where $A \in \mathbb{R}^{n_s \times n_y}$ and $b \in \mathbb{R}^{n_s}$. Denote the $i$th row of $A$ by $a_i^\top$ and the $i$th element of $b$ by $b_i$. In this setting, the condition $y \in \mathcal{S}$ is equivalent to $\min_{i \in \{1,2,\ldots,n_s\}} a_i^\top y + b_i \geq 0$. Therefore, the deterministic robustness level is naturally formulated as

$$r^\star = \inf_{y \in \mathcal{Y}} \min_{i \in \{1,2,\ldots,n_s\}} a_i^\top y + b_i,$$

so that $r^\star \geq 0$ certifies that $Y = f(X)$ is safe with probability one. Then the approximate robustness level using a surrogate output set $\hat{\mathcal{Y}}$ becomes

$$\hat{r}(\hat{\mathcal{Y}}) = \inf_{y \in \hat{\mathcal{Y}}} \min_{i \in \{1,2,\ldots,n_s\}} a_i^\top y + b_i.$$

Moreover, the condition that $f(X)$ has safety level at least $r$ with high probability is naturally encoded in the following probabilistic robustness level:

$$\bar{r}(\epsilon) = \sup \left\{ r \in \mathbb{R} : \mathbb{P}_X \left( \min_{i \in \{1,2,\ldots,n_s\}} a_i^\top f(X) + b_i \geq r \right) \geq 1 - \epsilon \right\}.$$

With our robustness levels defined for the general polyhedral safe set, we now outline the procedure to generalize our main results from the single half-space case presented in the chapter. To this end, start by prescribing probability levels $\epsilon, \delta \in [0,1]$ close to zero, and define $\epsilon' = \epsilon/n_s$ and $\delta' = \delta/n_s$. Then, for all $i \in \{1, 2, \ldots, n_s\}$, perform the proposed assessment method for the single half-space setting using the parameters $\epsilon'$, $\delta'$, $a_i$, and $b_i$ in place of $\epsilon$, $\delta$, $a$, and $b$, respectively. In particular, for every $i$, use $N' \geq \frac{2}{\epsilon'}\left(\log\frac{1}{\delta'} + p\right)$ independent and identically distributed samples in the scenario problem (4.6), ensuring that the samples across different values of $i$ are also independent. Notice that the sample size $N'$ grows with $n_s$ like $n_s \log(n_s)$, so the increase in the number of samples used for each scenario problem is modest, as it is nearly linear. However, the total number of samples needed across all $n_s$ scenario problems grows with $n_s$ like $n_s^2 \log(n_s)$, and therefore the computational cost may become prohibitive in the case the safe set is defined by a large number of half-spaces. To remedy this, one may first compute a polyhedral inner-approximation of $\mathcal{S}$ with a much smaller number of half-spaces, and then apply the methods outlined in this section.

Now, let $\theta_i^\star$ denote the solution to the scenario problem (4.6) corresponding to row $i$ of the safe set. Remark that the solutions $\theta_i^\star$ are all random, although they are not necessarily defined on the same probability space, as their distributions depend on the particular values for $a_i$ and $b_i$ used to compute them. For notational convenience, denote the probability distribution of $\theta_i^\star$ by $\mathbb{P}_i$, and denote by $\mathbb{P}$ the product probability measure associated with $(\theta_1^\star, \theta_2^\star, \ldots, \theta_{n_s}^\star)$. Then, Theorem 11 gives for all $i$ that, with probability at least $1-\delta'$, the set $h(\theta_i^\star)$ is an $\epsilon'$-cover of $\mathcal{Y} = f(\mathcal{X})$. That is,

$$\mathbb{P}_i(\mathbb{P}_X(f(X) \in h(\theta_i^\star)) \geq 1 - \epsilon') \geq 1 - \delta'.$$

Note that if $\hat{\mathcal{Y}}_i$ are $\frac{\epsilon}{n_c}$-covers of $\mathcal{Y}$ for all $i \in \{1, 2, \ldots, n_c\}$, then $\bigcap_{i=1}^{n_c} \hat{\mathcal{Y}}_i$ is an $\epsilon$-cover, since

$$\begin{aligned}
\mathbb{P}_X\left(f(X) \in \bigcap_{i=1}^{n_c} \hat{\mathcal{Y}}_i\right) &= 1 - \mathbb{P}_X\left(f(X) \in \bigcup_{i=1}^{n_c} \hat{\mathcal{Y}}_i^c\right) \\
&\geq 1 - \sum_{i=1}^{n_c} \mathbb{P}_X(f(X) \in \hat{\mathcal{Y}}_i^c) \\
&= 1 - \sum_{i=1}^{n_c} (1 - \mathbb{P}_X(f(X) \in \hat{\mathcal{Y}}_i)) \\
&\geq 1 - \sum_{i=1}^{n_c} \frac{\epsilon}{n_c} \\
&= 1 - \epsilon,
\end{aligned}$$

where $\hat{\mathcal{Y}}_i^c$ denotes the complement $\mathbb{R}^{n_y} \setminus \hat{\mathcal{Y}}_i$. Using the monotonicity and subadditivity of the

product measure $\mathbb{P}$, we can apply this result to the sets $h(\theta_i^\star)$ to find that

$$\begin{aligned}
\mathbb{P}\left(\mathbb{P}_X\left(f(X) \in \bigcap_{i=1}^{n_s} h(\theta_i^\star)\right) \geq 1-\epsilon\right) &\geq \mathbb{P}(\mathbb{P}_X\left(f(X) \in h(\theta_i^\star)\right) \geq 1-\epsilon' \text{ for all } i) \\
&= 1 - \mathbb{P}(\mathbb{P}_X(f(X) \in h(\theta_i^\star)) < 1-\epsilon' \text{ for some } i) \\
&\geq 1 - \sum_{i=1}^{n_s} \mathbb{P}_i(\mathbb{P}_X(f(X) \in h(\theta_i^\star)) < 1-\epsilon') \\
&= 1 - \sum_{i=1}^{n_s} (1 - \mathbb{P}_i(\mathbb{P}_X(f(X) \in h(\theta_i^\star)) \geq 1-\epsilon')) \\
&\geq 1 - \sum_{i=1}^{n_s} \delta' \\
&= 1 - \delta.
\end{aligned}$$

Therefore, we conclude that $\bigcap_{i=1}^{n_s} h(\theta_i^\star)$ is an $\epsilon$-cover of $\mathcal{Y}$ with probability at least $1-\delta$.

Note that every individual $h(\theta_i^\star)$ is an $\epsilon$-cover of $\mathcal{Y}$ with probability at least $1-\delta$ as well, since it is an $\epsilon'$-cover with probability at least $1-\delta' \geq 1-\delta$ by construction, and an $\epsilon'$-cover is certainly an $\epsilon$-cover since $\epsilon' \leq \epsilon$. However, it is important to remark that the intersection $\bigcap_{i=1}^{n_s} h(\theta_i^\star)$ is clearly a tighter $\epsilon$-cover than any of the individual covers $h(\theta_i^\star)$, and therefore it gives better probabilistic localization of the output. Furthermore, recall that the $\epsilon$-cover is used as a surrogate output set to compute the approximate robustness level in order to lower-bound the probabilistic robustness level. Therefore, we'd like to choose the $\epsilon$-cover so that the approximate robustness level is maximal. Since

$$\begin{aligned}
\hat{r}\left(\bigcap_{i=1}^{n_s} h(\theta_i^\star)\right) &= \inf\left\{s(y) : y \in \bigcap_{i=1}^{n_s} h(\theta_i^\star)\right\} \\
&\geq \inf \bigcap_{i=1}^{n_s} \{s(y) : y \in h(\theta_i^\star)\} \\
&\geq \max_{i \in \{1,2,\ldots,n_s\}} \inf\{s(y) : y \in h(\theta_i^\star)\} \\
&= \max_{i \in \{1,2,\ldots,n_s\}} \hat{r}(h(\theta_i^\star)),
\end{aligned}$$

where $s(y) = \min_{i \in \{1,2,\ldots,n_s\}} a_i^\top y + b_i$ is the safety level of $y$ with respect to the general polyhedral safe set $\mathcal{S}$, it is clear that using the intersection $\bigcap_{i=1}^{n_s} h(\theta_i^\star)$ will give a tighter bound on the probabilistic robustness level than any of the individual covers $h(\theta_i^\star)$.

Since we know that $\bigcap_{i=1}^{n_s} h(\theta_i^\star)$ is an $\epsilon$-cover with probability $1-\delta$, the only result that remains to be generalized is the second conclusion from Theorem 11, i.e., we want to formally guarantee that $\hat{r}\left(\bigcap_{i=1}^{n_s} h(\theta_j^\star)\right) \leq \bar{r}(\epsilon)$ with probability $1-\delta$. This follows readily from the fact that $\mathbb{P}\left(\mathbb{P}_X\left(f(X) \in \bigcap_{i=1}^{n_s} h(\theta_i^\star)\right) \geq 1-\epsilon\right) \geq 1-\delta$ together with Proposition 14 and the law of total probability, just as in the proof of Theorem 11.

Finally, note that when the class $\mathcal{H}$ of surrogate output sets is designed so that $h(\theta)$ is convex for all $\theta \in \Theta$, the value $\hat{r}\left(\bigcap_{i=1}^{n_s} h(\theta_i^\star)\right) = \inf_{y \in \bigcap_{j=1}^{n_s} h(\theta_j^\star)} \min_{i \in \{1,2,\ldots,n_s\}} a_i^\top y + b_i = \min_{i \in \{1,2,\ldots,n_s\}} \inf_{y \in \bigcap_{j=1}^{n_s} h(\theta_j^\star)} a_i^\top y + b_i$ is easily computable, as it involves $n_s$ minimizations of affine functions over the convex set $\bigcap_{i=1}^{n_s} h(\theta_i^\star)$. This completes the generalization of our assessment method to the case with a general polyhedral safe set.

# 4.C Distributionally Robust Extension

In this section, we formulate a distributionally robust variant of the proposed assessment procedure. Consider the case where the neural network input $X$ has a finite number of known possible probability distributions $\mathbb{P}_1, \mathbb{P}_2, \ldots, \mathbb{P}_q$, but that at any point in time, the true input distribution $\mathbb{P}_X$ is unknown. For simplicity, we use the notation $\mathbb{P}_k(P(X))$, where $P$ is a mathematical predicate, to mean the probability of the event $P(X)$ when $X$ is distributed according to $\mathbb{P}_k$. Naturally, we formulate the following distributionally robust variant to the chance-constrained problem (4.5):

$$\begin{aligned} \underset{\theta \in \Theta}{\text{maximize}} \quad & \hat{r}(\theta) - \lambda v(\theta) \\ \text{subject to} \quad & \min_{k \in \{1,2,\ldots,q\}} \mathbb{P}_k(f(X) \in h(\theta)) \geq 1 - \epsilon. \end{aligned}$$

As before, we consider a scenario-based approximation to the above chance-constrained problem. However, instead of directly analyzing the above problem, consider treating each distribution separately. That is, let $\delta' = \delta/q$, and for all $k \in \{1, 2, \ldots, q\}$, formulate the scenario problem

$$\begin{aligned} \underset{\theta \in \Theta}{\text{maximize}} \quad & \hat{r}(\theta) - \lambda v(\theta) \\ \text{subject to} \quad & y_{j,k} \in h(\theta) \text{ for all } j \in \{1, 2, \ldots, N'\}, \end{aligned}$$

where we take the sample size to be $N' = \left\lceil \frac{2}{\epsilon}\left(\log \frac{1}{\delta'} + p\right)\right\rceil$, the samples $x_{1,k}, x_{2,k}, \ldots, x_{N',k}$ are drawn independently and identically from $\mathbb{P}_k$, and $y_{j,k} = f(x_{j,k})$. Denote the solution to the $k$th such scenario problem as $\theta_k^\star$ and its associated probability distribution as $\mathbb{P}_{\theta_k^\star}$. Also, denote by $\mathbb{P}_{\theta^\star}$ the product probability measure associated with $(\theta_1^\star, \theta_2^\star, \ldots, \theta_q^\star)$. Then by Theorem 11, we have for all $k$ that $\mathbb{P}_{\theta_k^\star}(\mathbb{P}_k(f(X) \in h(\theta_k^\star)) \geq 1 - \epsilon) \geq 1 - \delta'$. Therefore,

performing a similar line of analysis as in Appendix 4.B, we find that

$$\begin{aligned}
\mathbb{P}_{\theta^\star}\left(\mathbb{P}_X\left(f(X)\in\bigcup_{k=1}^{q}h(\theta_k^\star)\right)\geq 1-\epsilon\right) &\geq \mathbb{P}_{\theta^\star}\left(\mathbb{P}_j\left(f(X)\in\bigcup_{k=1}^{q}h(\theta_k^\star)\right)\geq 1-\epsilon \text{ for all } j\right)\\
&\geq \mathbb{P}_{\theta^\star}\left(\mathbb{P}_j(f(X)\in h(\theta_j^\star))\geq 1-\epsilon \text{ for all } j\right)\\
&= 1-\mathbb{P}_{\theta^\star}\left(\mathbb{P}_j(f(X)\in h(\theta_j^\star))< 1-\epsilon \text{ for some } j\right)\\
&\geq 1-\sum_{j=1}^{q}\mathbb{P}_{\theta_j^\star}\left(\mathbb{P}_j(f(X)\in h(\theta_j^\star))< 1-\epsilon\right)\\
&\geq 1-\sum_{j=1}^{q}\delta'\\
&= 1-\delta.
\end{aligned}$$

Therefore, the set $\bigcup_{k=1}^{q}h(\theta_k^\star)$ is an $\epsilon$-cover of $\mathcal{Y}$ with probability at least $1-\delta$. Note that this distributionally robust approach naturally leads to the union of precomputed covers, in contrast to the intersection found in Appendix 4.B. This is to be expected, since each set $h(\theta_k^\star)$ in the current discussion brings new information about where outputs could be located when the input is distributed according to $\mathbb{P}_k$, and this new information should be included in the final $\epsilon$-cover so as to ensure good localization of the output in a distributionally robust sense. In contrast, since all of the samples in Appendix 4.B come from the same distribution, it is reasonable to intersect the resulting output set estimates and still obtain a good estimate of the true output set $\mathcal{Y}$ with high-probability localization guarantees.

Now, following the same analysis as in Appendix 4.B, it is easy to see that, with probability $1-\delta$, the probabilistic robustness level is lower-bounded as $\hat{r}\left(\bigcup_{k=1}^{q}h(\theta_k^\star)\right)\leq\bar{r}(\epsilon)$, where $\hat{r}(\cdot)$ and $\bar{r}(\cdot)$ are as defined in (4.2) and (4.3), respectively. Finally, note that computing $\hat{r}\left(\bigcup_{k=1}^{q}h(\theta_k^\star)\right)$ is simple, since

$$\begin{aligned}
\hat{r}\left(\bigcup_{k=1}^{q}h(\theta_k^\star)\right) &= \inf\left\{a^\top y+b : y\in\bigcup_{k=1}^{q}h(\theta_k^\star)\right\}\\
&= \inf\bigcup_{k=1}^{q}\{a^\top y+b : y\in h(\theta_k^\star)\}\\
&= \min_{k\in\{1,2,\ldots,q\}}\inf\{a^\top y+b : y\in h(\theta_k^\star)\}\\
&= \min_{k\in\{1,2,\ldots,q\}}\hat{r}(\theta_k^\star),
\end{aligned}$$

and the values $\hat{r}(\theta_k^\star)$ have already been computed using convex optimization.

## 4.D Special Case: Class of Half-Spaces

In this section, we consider the special case of the scenario problem (4.6) where $\lambda = 0$, $\Theta = \mathbb{R}^{n_y} \times \mathbb{R}$, and $h\colon \Theta \to \mathcal{P}(\mathbb{R}^{n_y})$ is given by $h(c,d) = \{y \in \mathbb{R}^{n_y} : c^\top y + d \geq 0\}$. Then $\mathcal{H}$ is the class of all half-spaces within the output space $\mathbb{R}^{n_y}$. We will show that, 1) the scenario problem has a closed-form solution, and 2) the scenario problem coincides with the optimization obtained by applying the scenario approach directly to the definition of $\bar{r}(\epsilon)$.

Under the given conditions, the approximate robustness level becomes $\hat{r}(c,d) = \inf\{a^\top y + b : c^\top y + d \geq 0\}$. The Lagrangian for this minimization problem is

$$L(y,\mu) = a^\top y + b - \mu(c^\top y + d) = (a - \mu c)^\top y + b - \mu d,$$

where $\mu \geq 0$ denotes the Lagrange multiplier. Since the Lagrangian is affine in $y$, the dual function is

$$g(\mu) = \inf_{y \in \mathbb{R}^{n_y}} L(y,\mu) = \begin{cases} b - \mu d & \text{if } a = \mu c, \\ -\infty & \text{otherwise.} \end{cases}$$

Therefore, the dual problem corresponding to the primal minimization over $y$ becomes

$$\begin{aligned} \underset{\mu \in \mathbb{R}}{\text{maximize}} \quad & b - \mu d \\ \text{subject to} \quad & a = \mu c,\ \mu \geq 0. \end{aligned}$$

Now, since the primal problem over $y$ is a feasible linear program, we have that strong duality holds [24]. Hence, $\hat{r}(c,d) = \sup\{b - \mu d : a = \mu c,\ \mu \geq 0\}$. Therefore, the scenario optimization problem (4.6) reduces to

$$\begin{aligned} \underset{c \in \mathbb{R}^{n_y},\ d,\mu \in \mathbb{R}}{\text{maximize}} \quad & b - \mu d \\ \text{subject to} \quad & a = \mu c,\ \mu \geq 0, \\ & c^\top y_j + d \geq 0 \text{ for all } j \in \{1,2,\ldots,N\}. \end{aligned} \tag{4.15}$$

We now solve the scenario problem (4.15) in closed form. First, under the assumption that the safe set is nontrivial, i.e., $a \neq 0$, we remark that the constraint $a = \mu c$ implies that $\mu \neq 0$, and therefore can be rewritten as $c = \frac{1}{\mu} a$. Eliminating $c$, the optimization becomes

$$\begin{aligned} \underset{d,\mu \in \mathbb{R}}{\text{maximize}} \quad & b - \mu d \\ \text{subject to} \quad & \mu > 0,\ \frac{1}{\mu} a^\top y_j + d \geq 0,\ j \in \{1,2,\ldots,N\}. \end{aligned}$$

Defining $\tilde{r} = b - \mu d$, the problem further reduces to

$$\begin{aligned} \underset{\tilde{r} \in \mathbb{R}}{\text{maximize}} \quad & \tilde{r} \\ \text{subject to} \quad & a^\top y_j + b \geq \tilde{r} \text{ for all } j \in \{1,2,\ldots,N\}. \end{aligned} \tag{4.16}$$

It is clear that the reduced scenario problem (4.16) matches the formulation obtained by directly applying the scenario approach to estimate $\bar{r}(\epsilon) = \sup\{r \in \mathbb{R} : \mathbb{P}_X(a^\top f(X) + b \geq r) \geq 1 - \epsilon\}$. In fact, since the optimization defining $\bar{r}(\epsilon)$ is univariate, whereas the scenario problem (4.15) over $\Theta$ is $(n_y + 1)$-dimensional, the number of samples indicated by Theorem 11 is conservative for this problem. Instead of $\frac{2}{\epsilon}(\log\frac{1}{\delta} + n_y + 1)$ samples, only $N \geq \frac{2}{\epsilon}(\log\frac{1}{\delta} + 1)$ samples are needed to obtain the high-probability guarantees provided by Theorem 11. We also note that the optimization (4.16) is a univariate linear program, and is clearly solved in closed-form by $\tilde{r}^\star = \min_{j\in\{1,2,\ldots,N\}} a^\top y_j + b$. This derivation results in the following proposition:

**Proposition 16.** *Let $\epsilon, \delta \in [0, 1]$, $N \geq \frac{2}{\epsilon}(\log\frac{1}{\delta} + 1)$, and $\{x_j : j \in \{1, 2, \ldots, N\}\}$ be a set of $N$ independently and identically distributed samples drawn from $\mathbb{P}_X$. Let $y_j = f(x_j)$ for all $j \in \{1, 2, \ldots, N\}$. Then with probability $1 - \delta$, the probabilistic robustness level $\bar{r}(\epsilon)$ is lower-bounded by $\tilde{r}^\star = \min_{j\in\{1,2,\ldots,N\}} a^\top y_j + b$.*

Despite being derived from our general framework, this special case reduces to a solution that is remarkably simple and coincides with a heuristic one may first try using in practice. That is, upon choosing $\mathcal{H}$ to be the class of half-spaces, the optimal sample-based method for lower-bounding the probabilistic robustness level via the approximate robustness level is to compute the minimum safety level amongst the collection of sampled outputs. If sufficiently many samples are used and the minimum safety level is nonnegative, then we certify with high probability that the unknown random output $Y = f(X)$ is safe in practice. Although this special case does not yield meaningful localization of the outputs and uses a crude class of surrogate output sets, it certainly provides a fast analytical method for certifying the network's robustness against random input noise. Furthermore, our derivation mathematically justifies the use of this otherwise heuristic method, and, contrarily, the natural intuition behind this statistical estimator validates our framework that generalizes it.

# 4.E Additional Numerical Simulations

## Alternate Illustrative Example

In this section, we showcase an illustrative example similar to that in Section 4.6, albeit now we consider the multilayer perceptron proposed in Example 2 in Xiang, Tran, and Johnson [155]. The network uses a $\tanh(\cdot)$ activation function and has two inputs and two outputs, making the visualization of the input and output sets possible. See Xiang, Tran, and Johnson [155] for more details of the network. The noisy input is distributed uniformly on $\mathcal{X} = \{x \in \mathbb{R}^2 : |x_1 - 0.5| \leq 1.5,\ |x_2 - 0.5| \leq 0.1\}$, where $\bar{x} = (0.5, 0.5)$ is the nominal input. The safe set is $\mathcal{S}_1 = \{y \in \mathbb{R}^2 : a^\top y + b \geq 0\}$, where $a = (1, 0)$ and $b = 3.7$.

The norm ball class $\mathcal{H}$ of Example 3, Example 4, and Example 5 is employed with $\|\cdot\|$ being the $\ell_2$-norm.

The probability levels are chosen as $\epsilon = 0.1$ and $\delta = 10^{-5}$. We set $N = \left\lceil \frac{2}{\epsilon}(\log \frac{1}{\delta} + p) \right\rceil = 291$, then uniformly sample $N$ inputs $x_j$ from $\mathcal{X}$ and compute their corresponding outputs $y_j$. As shown in Example 5, the scenario problem takes the form given in (4.7). We choose the regularizer to be the square of the norm ball radius, i.e., $v(\bar{y}, r) = r^2$. The optimization problem is convex as guaranteed by Theorem 10.

We solve the scenario problem first without regularization, and then with two different levels of regularization: $\lambda_1 = 0.001$ and $\lambda_2 = 1$. The respective solutions are denoted by $\theta^\star$, $\theta^\star_{\lambda_1}$, and $\theta^\star_{\lambda_2}$. Each instance takes approximately 5 seconds to solve using CVX in MATLAB on a standard laptop with a 2.6 GHz dual-core i5 processor. The resulting approximate robustness levels are $\hat{r}(\theta^\star) = 0.423$, $\hat{r}(\theta^\star_{\lambda_1}) = 0.419$, and $\hat{r}(\theta^\star_{\lambda_2}) = -1.107$. In the instances without regularization and with regularization level $\lambda_1$, Theorem 11 guarantees that the probabilistic robustness level $\bar{r}(0.1)$ is at least 0.4 with probability at least 0.99999. In other words, the random output $Y = f(X)$ has a safety level of 0.4 with high probability, granting the probablistic robustness certificate we seek. On the other hand, since $\hat{r}(\theta^\star_{\lambda_2}) < 0$, the scenario problem using regularization level $\lambda_2$ is not able to certify the safety of the output. This is due to the inherent tradeoff between localization and certification, which we now discuss further.

The optimal $\epsilon$-covers $h(\theta^\star)$, $h(\theta^\star_{\lambda_1})$, and $h(\theta^\star_{\lambda_2})$ are shown in Figure 4.5. The unregularized set $h(\theta^\star)$ is massively over-conservative due to the choice $\lambda = 0$, which corresponds to pure robustness certification. Indeed, $h(\theta^\star)$ is the $\epsilon$-cover from our class of sets that is furthest from the boundary of the safe set, making $\hat{r}(\theta^\star)$ the tightest lower bound on $\bar{r}(\epsilon)$. On the other hand, the optimal $\epsilon$-covers using $\lambda = \lambda_1$ and $\lambda = \lambda_2$ are seen to give tighter localizations of the output $Y$. The approximate robustness level using regularization $\lambda_1$ is only slightly lower than the unregularized value, but the regularization $\lambda_2$ is large enough to cause the approximate robustness level $\hat{r}(\theta^\star_{\lambda_2})$ to become negative at the expense of localization. This shows how overemphasizing localization may actually harm the certification aspect of robustness assessment, and empirically demonstrates why output set estimation methods may not be adequate for issuing robustness certificates.

We now repeat the same simulation with a more complicated safe set. In particular, we add an additional constraint to the safe set to match Example 2 given in Xiang, Tran, and Johnson [155], so that it now takes the form $\mathcal{S}_2 = \{y \in \mathbb{R}^2 : Ay + b \geq 0\}$, where $A = \left[\begin{smallmatrix} 1 & 0 \\ -1 & 0 \end{smallmatrix}\right]$ and $b = (3.7, -1.5)$. In this case, we apply our proposed method to each row of the safe set individually. To do so, we set $\epsilon' = \epsilon/2$ and $\delta' = \delta/2$, then define $N' = \left\lceil \frac{2}{\epsilon'} \left(\log \frac{1}{\delta'} + p\right) \right\rceil = 609$. For each of the two half-spaces defining the safe set, we solve the scenario problem using $N'$ independent and identically distributed input-output samples, and then we take the intersection of the two resulting $\epsilon'$-covers. Doing so, we obtain an $\epsilon$-cover of the output set with probability at least $1 - \delta$. We repeat this process again using regularization levels $\lambda_1 = 0.001$ and $\lambda_2 = 1$, and we find that each scenario problem takes approximately 11 seconds to solve. The resulting covers are shown in Figure 4.6.

We find that the approximate robustness levels corresponding to $\lambda = 0$ and $\lambda = \lambda_1$ are strictly positive for both half-spaces, certifying that the random output $Y$ is safe with the prescribed probability. However, for $\lambda = \lambda_2$, the optimal $\epsilon$-covers corresponding to both

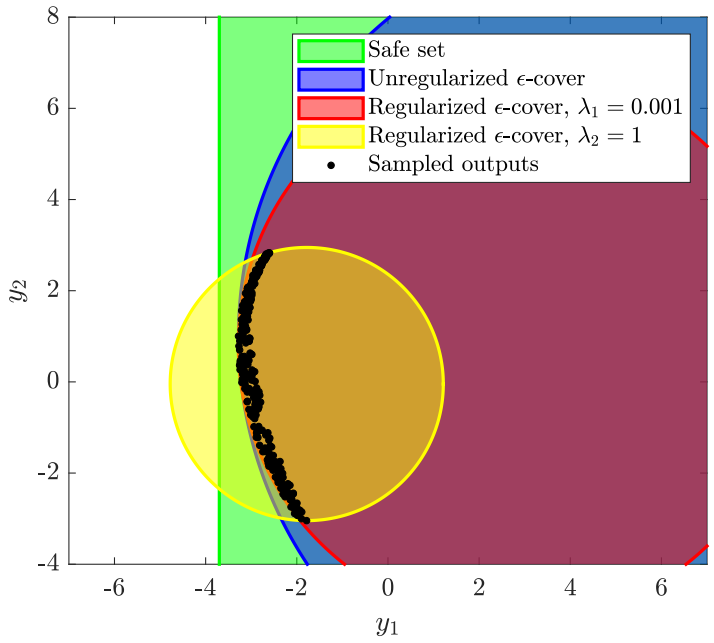



Figure 4.5: Optimal $\ell_2$-norm ball $\epsilon$-covers for safe set $\mathcal{S}_1$.

half-spaces are found to intersect the unsafe region of the output space, due to the increased emphasis on localization. Interestingly, the overall localization after intersecting the two $\epsilon'$-covers for $\lambda = \lambda_2$ is in a sense looser than that of the case $\lambda = \lambda_1$, indicating that moderate regularization levels, like $\lambda_1$ in this simulation, may simultaneously perform best for both localization and certification in the case of general polyhedral safe sets defined by more than one half-space. Optimizing $\lambda$ in general poses an interesting problem for future research.

## Comparison to DeepPAC

In this simulation, we use the half-space special case of our proposed robustness certification method, as presented in Appendix 4.D. Recall that this method uses all optimization efforts to certify the robustness of the network; the outputs are not localized within the output space. For this example, we append a ReLU layer with normal random weights and biases to the neural network presented in our illustrative example of Section 4.6, maintaining $n_y = 2$, and we consider the resulting network as a classifier. Ten nominal inputs are chosen randomly at which we will perform robustness certification. The noisy input $X$ is distributed uniformly on the input set $\mathcal{X} = \{x \in \mathbb{R}^2 : \|x - \bar{x}\| \leq \epsilon_x\}$, where the radius $\epsilon_x$ is varied from 0.1 to 1. We set the probabilistic confidence levels to be $\epsilon = 0.1$ and $\delta = 10^{-5}$.

For each nominal input and input set radius, we compute a lower bound on the probabilistic robustness level $\bar{r}(\epsilon)$ first using our proposed methodology, and then using the

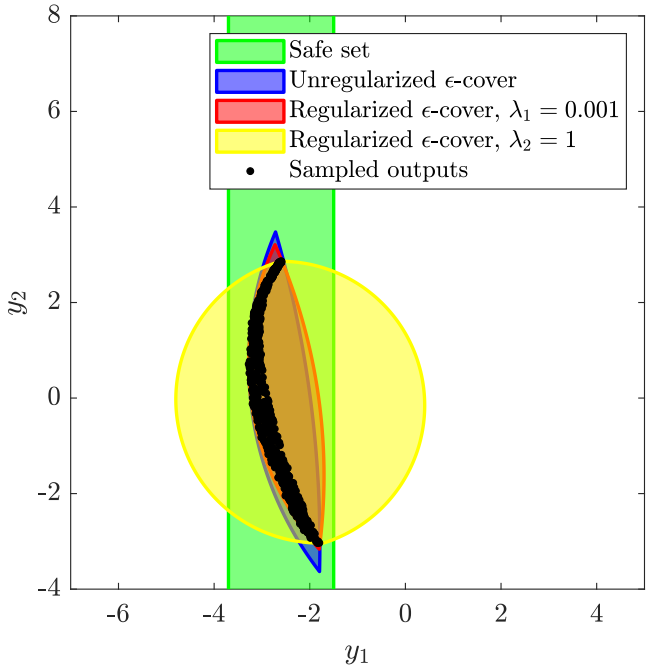



Figure 4.6: Optimal intersections of two $\ell_2$-norm ball $\frac{\epsilon}{2}$-covers for safe set $\mathcal{S}_2$.

scenario-based approach presented in Li et al. [89], termed DeepPAC.[6] Solving for such a lower bound using DeepPAC first entails solving a scenario linear program for an affine bound on the classifier's margin function, and then requires optimizing this bound over the input set. We remark that DeepPAC requires more samples (and therefore optimization constraints) than our approach, specifically, DeepPAC requires $N \geq \frac{2}{\epsilon}(\log \frac{1}{\delta} + (n_x + 1)(n_y - 1) + 1)$, and therefore we restrict our comparison to DeepPAC to this moderately sized example for computational convenience. See our comparison to PROVEN in Section 4.6 for applications of our approach to large MNIST and CIFAR-10 networks.

After computing the lower bounds on $\bar{r}(\epsilon)$ using both methods, we average the values over the nominal inputs, independently for each input set radius. The results are shown in Figure 4.7. As seen, the lower bounds between the two methods remain close for small input set radii, but our approach offers a tighter lower bound as the radii increase. At input set radius $\epsilon_x = 0.4$, our approach is able to issue a high-probability robustness certificate on average, whereas DeepPAC fails. These observations are explained as follows.

DeepPAC works by using samples to learn an affine approximation to the nonlinear margin function over $\mathcal{X}$, so that high-probability bounds on the margin function values at noisy inputs can be made using the learned affine function. Using affine functions to bound the margin function is a technique that naturally applies when considering worst-

[6]Like our robustness certificates, those given by DeepPAC are of the probably approximately correct form (see Remark 5), hence the name DeepPAC.

case or adversarial inputs (e.g., Weng et al. [148] and Zhang et al. [168]). However, as the input set becomes larger, affine approximations are no longer able to accurately capture the nonlinearities of the margin function. Consequently, DeepPAC's high-probability affine bounds on the margin function values become loose, resulting in a looser lower bound on the probabilistic robustness level. Our approach avoids this worst-case analysis technique by directly learning a set in the output space instead of learning a mapping from the input to the output space. This behavior is also seen in our comparison to PROVEN in Section 4.6.

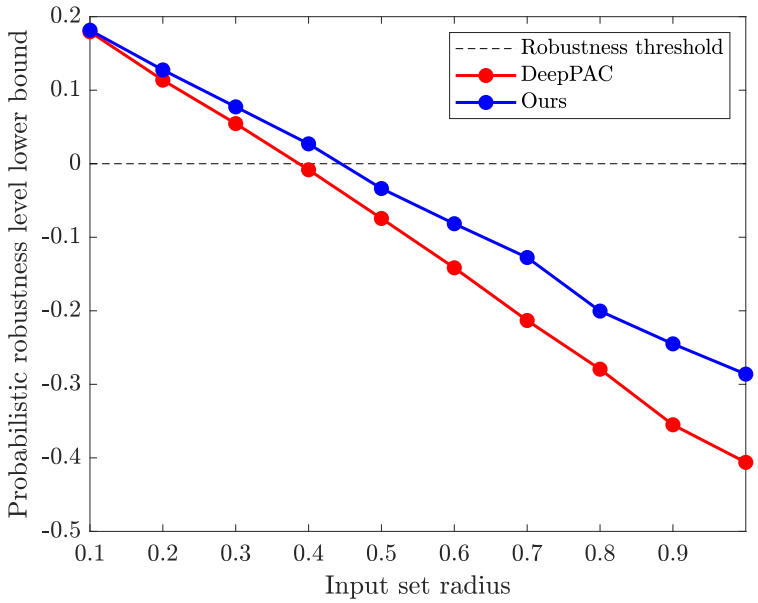



Figure 4.7: Our lower bound closely matches that of DeepPAC for small input set radii, but becomes noticeably tighter than DeepPAC as the input set becomes larger.

## Comparison to Direct Bayesian Certification

In this section, we repeat the simulation from our comparison to PROVEN in Section 4.6 using the sample-based certification method given in Zakrzewski [162]. Recall that Zakrzewski [162] imposes a Bayesian framework on the problem by directly assuming that the failure probability follows a uniform prior distribution. In doing so, Zakrzewski [162] is able to certify with probability $1-\delta$ that $\mathbb{P}_X(f(X) \in \mathcal{S}) \geq 1-\epsilon$, so long as the number of samples used is $N \geq \frac{1}{\epsilon}\log\frac{1}{\delta} - 1$ and $f(x_j) \in \mathcal{S}$ for all sampled inputs $x_j$, $j \in \{1, 2, \ldots, N\}$. In comparing our method to Zakrzewski [162], two important remarks should be made. First, our method is much more general, as we are able to localize the outputs in arbitrary surrogate output sets, so long as they satisfy the assumptions of Theorem 10, yielding a convex scenario optimization problem, whereas the method of Zakrzewski [162] is only able to certify whether or not the outputs are contained in the safe half-space. Thus, to compare the

methods, we restrict our method to the special case of half-space surrogate output sets from Appendix 4.D. Second, for the same amount of samples ($O(\frac{1}{\epsilon}\log\frac{1}{\delta})$), our method provides more information regarding the robustness of the network than Zakrzewski [162] does. In particular, we lower bound the probabilistic robustness level $\bar{r}(\epsilon)$ with a *continuous value*, whereas Zakrzewski [162] is only able to issue a *binary* certificate asserting whether or not outputs are contained in the safe set with high probability. With this in mind, our method is the only one of the two that is able to quantify *how* safe the outputs are by certifying a continuous-valued safety margin.

Table 4.3: Average probabilistic robustness level lower bounds $\hat{r}(\theta^\star)$ for MNIST ReLU networks subject to uniform noise over $\ell_\infty$-norm ball. All values are averaged over 10 nominal inputs with randomly chosen target classes $i$. Also reported are the percentages of inputs that each method is able to certify. Lower bounds giving certified robustness (on average) are bolded, and the average certified adversarial radii computed using Zhang et al. [168] are italicized.

(a) $2 \times [20]$ network.

| Radius | $\epsilon = 0.001$ | | $\epsilon = 0.1$ | | $\epsilon = 0.25$ | |
|---|---|---|---|---|---|---|
| | [162] | Ours | [162] | Ours | [162] | Ours |
| 0.01 | **0.00** | **14.11** | **0.00** | **14.26** | **0.00** | **14.28** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| *0.027* | **0.00** | **13.36** | **0.00** | **13.77** | **0.00** | **13.85** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.05 | **0.00** | **12.34** | **0.00** | **13.11** | **0.00** | **13.26** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.1 | **0.00** | **10.25** | **0.00** | **11.66** | **0.00** | **12.00** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.5 | N/A | −7.21 | N/A | −0.05 | N/A | **1.71** |
| | 20% | 20% | 40% | 40% | 50% | 50% |

(b) $3 \times [20]$ network.

| Radius | $\epsilon = 0.001$ | | $\epsilon = 0.1$ | | $\epsilon = 0.25$ | |
|---|---|---|---|---|---|---|
| | [162] | Ours | [162] | Ours | [162] | Ours |
| 0.01 | **0.00** | **17.28** | **0.00** | **17.45** | **0.00** | **17.48** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| *0.022* | **0.00** | **16.65** | **0.00** | **17.02** | **0.00** | **17.10** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.05 | **0.00** | **15.19** | **0.00** | **16.00** | **0.00** | **16.19** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.1 | **0.00** | **12.57** | **0.00** | **14.19** | **0.00** | **14.58** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.5 | N/A | −9.36 | N/A | −0.55 | N/A | **0.36** |
| | 0% | 0% | 50% | 50% | 62.5% | 62.5% |

(c) $2 \times [1024]$ network.

| Radius | $\epsilon = 0.001$ | | $\epsilon = 0.1$ | | $\epsilon = 0.25$ | |
|---|---|---|---|---|---|---|
| | [162] | Ours | [162] | Ours | [162] | Ours |
| 0.01 | **0.00** | **27.73** | **0.00** | **27.87** | **0.00** | **27.93** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| *0.032* | **0.00** | **26.69** | **0.00** | **27.13** | **0.00** | **27.32** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.05 | **0.00** | **25.83** | **0.00** | **26.53** | **0.00** | **26.82** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.1 | **0.00** | **23.46** | **0.00** | **24.84** | **0.00** | **25.42** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.5 | N/A | **4.83** | **0.00** | **11.79** | **0.00** | **14.45** |
| | 80% | 80% | 100% | 100% | 100% | 100% |

(d) $3 \times [1024]$ network.

| Radius | $\epsilon = 0.001$ | | $\epsilon = 0.1$ | | $\epsilon = 0.25$ | |
|---|---|---|---|---|---|---|
| | [162] | Ours | [162] | Ours | [162] | Ours |
| 0.01 | **0.00** | **36.86** | **0.00** | **37.06** | **0.00** | **37.12** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| *0.024* | **0.00** | **35.97** | **0.00** | **36.44** | **0.00** | **36.58** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.05 | **0.00** | **34.32** | **0.00** | **35.32** | **0.00** | **35.59** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.1 | **0.00** | **31.10** | **0.00** | **33.10** | **0.00** | **33.69** |
| | 100% | 100% | 100% | 100% | 100% | 100% |
| 0.5 | N/A | **6.89** | N/A | **15.85** | **0.00** | **18.56** |
| | 80% | 80% | 90% | 90% | 100% | 100% |

The results of the experimental comparison are given in Table 4.3. It is observed that our method is always able to issue a robustness certificate whenever Zakrzewski [162] does,

and furthermore, our lower bounds on the probabilistic robustness level are generally much less conservative than those granted by Zakrzewski [162] (which are binary; either a lower bound of 0, or failure to issue a certificate altogether). This emphasizes that, for the same number of samples, a special case of our method is strictly more informative as Zakrzewski [162], and always succeeds in issuing a robustness certificate whenever their method does.

# Part II

# Designing Robust Models

# Chapter 5

# Feature-Convex Neural Networks

Real-world adversarial attacks on machine learning models often feature an asymmetric structure wherein adversaries only attempt to induce false negatives (e.g., classify a spam email as not spam). In this chapter, we formalize the asymmetric robustness certification problem and correspondingly present the *feature-convex neural network* architecture, which composes an input-convex neural network (ICNN) with a Lipschitz continuous feature map in order to achieve asymmetric adversarial robustness. We consider the aforementioned binary setting with one "sensitive" class, and for this class we prove deterministic, closed-form, and easily-computable certified robust radii for arbitrary $\ell_p$-norms. We theoretically justify the use of these models by extending the universal approximation theorem for ICNN regression to the classification setting, and proving a lower bound on the probability that such models perfectly fit even unstructured uniformly distributed data in sufficiently high dimensions. Numerical simulations on Malimg malware classification and subsets of the MNIST, Fashion-MNIST, and CIFAR-10 datasets show that feature-convex classifiers attain substantial certified $\ell_1$-, $\ell_2$-, and $\ell_\infty$-radii while being far more computationally efficient than competitive baselines.

This chapter is based on the following previously published work:

[113] Samuel Pfrommer*, Brendon G. Anderson*, Julien Piet, and Somayeh Sojoudi, "Asymmetric certified robustness via feature-convex neural networks," *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. *Co-first author and equal contribution.

## 5.1 Introduction

The strength of neural network robustness certificates can be highly dependent on network architecture. For instance, general off-the-shelf models tend to have large Lipschitz constants, leading to loose Lipschitz-based robustness guarantees [65, 51, 160]. Consequently, lines of work that impose certificate-amenable structures onto networks have been popularized,

e.g., specialized model layers [136, 165], randomized smoothing-based networks [88, 35, 164, 158, 8], and ReLU networks that are certified using convex optimization and mixed-integer programming [150, 148, 116, 4, 94]. The first category only directly certifies against one specific choice of norm, producing poorly scaled radii for other norms in high dimensions. The latter two approaches incur serious computational challenges: randomized smoothing typically requires the classification of thousands of randomly perturbed samples per input, while optimization-based solutions scale poorly to large networks.

Despite the moderate success of these certifiable classifiers, conventional assumptions in the literature are unnecessarily restrictive for many practical adversarial settings. Specifically, most works consider a multiclass setting where certificates are desired for inputs of any class. By contrast, many real-world adversarial attacks involve a binary setting with only one *sensitive class* that must be robust to adversarial perturbations. Consider the representative problem of spam classification; a malicious adversary crafting a spam email will only attempt to fool the classifier toward the "not-spam" class—never conversely [38]. Similar logic applies for a range of applications such as malware detection [64], malicious network traffic filtering [121], fake news and social media bot detection [37], hate speech removal [63], insurance claims filtering [53], and financial fraud detection [29].

The important asymmetric nature of these classification problems has long been recognized in various subfields, and some domain-specific attempts at robustification have been proposed with this in mind. This commonly involves robustifying against adversaries appending features to the classifier input. In spam classification, such an attack is known as the "good word" attack [91]. In malware detection, numerous approaches have been proposed to provably counter such additive-only adversaries using special classifier structures such as non-negative networks [54] and monotonic classifiers [69]. We note these works strictly focus on *additive* adversaries and cannot handle general adversarial perturbations of the input that are capable of perturbing existing features. In this chapter, we propose adding this important asymmetric structure to the study of norm ball-certifiably robust classifiers. This narrowing of the problem to the asymmetric setting provides prospects for novel certifiable architectures, and we present feature-convex neural networks as one such possibility.

## Problem Statement and Contributions

This section formalizes the *asymmetric robustness certification problem* for general norm-bounded adversaries. Specifically, we assume a binary classification setting wherein one class is "sensitive"—meaning we seek to certify that, if some input is classified into this sensitive class, then adversarial perturbations of sufficiently small magnitude cannot change the prediction.

Formally, consider a binary classifier $f_\tau \colon \mathbb{R}^d \to \{1, 2\}$, where class 1 is the sensitive class for which we desire certificates. We take $f_\tau$ to be a standard thresholded version of a soft classifier $g \colon \mathbb{R}^d \to \mathbb{R}$, expressible as $f_\tau(x) = T_\tau(g(x))$, where $T_\tau \colon \mathbb{R} \to \{1, 2\}$ is the

thresholding function defined by

$$T_\tau(y) = \begin{cases} 1 & \text{if } y + \tau > 0, \\ 2 & \text{if } y + \tau \leq 0, \end{cases} \tag{5.1}$$

with $\tau \in \mathbb{R}$ being a user-specified parameter that shifts the classification threshold. A classifier $f_\tau$ is considered certifiably robust at a class 1 input $x \in \mathbb{R}^d$ with a radius $r(x) \in \mathbb{R}_+$ if $f_\tau(x+\delta) = f_\tau(x) = 1$ for all $\delta \in \mathbb{R}^d$ with $\|\delta\| < r(x)$ for some norm $\|\cdot\|$. Thus, $\tau$ induces a tradeoff between the clean accuracy on class 2 and certification performance on class 1. As $\tau \to \infty$, $f_\tau$ approaches a constant classifier which achieves infinite class 1 certified radii but has zero class 2 accuracy.

For a particular choice of $\tau$, the performance of $f_\tau$ can be analyzed similarly to a typical certified classifier. Namely, it exhibits a class 2 clean accuracy $\alpha_2(\tau) \in [0,1]$ as well as a class 1 certified accuracy surface $\Gamma$ with values $\Gamma(r,\tau) \in [0,1]$ that capture the fraction of the class 1 samples that can be certifiably classified by $f_\tau$ at radius $r \in \mathbb{R}_+$. The class 1 clean accuracy $\alpha_1(\tau) = \Gamma(0,\tau)$ is inferable from $\Gamma$ as the certified accuracy at $r = 0$.

The full asymmetric certification performance of the family of classifiers $f_\tau$ can be captured by plotting the surface $\Gamma(r,\tau)$, as will be shown in Figure 5.1a. Instead of plotting against $\tau$ directly, we plot against the more informative difference in clean accuracies $\alpha_1(\tau) - \alpha_2(\tau)$. This surface can be viewed as an asymmetric robustness analogue to the classic receiver operating characteristic curve.

Note that while computing the asymmetric robustness surface is possible for our feature-convex architecture (to be defined shortly), it is computationally prohibitive for conventional certification methods. We therefore standardize our comparisons throughout this chapter to the certified accuracy cross section $\Gamma(r,\tau^\star)$ for a $\tau^\star$ such that clean accuracies are balanced in the sense that $\alpha_2(\tau^\star) = \alpha_1(\tau^\star)$, noting that $\alpha_1$ monotonically increases in $\tau$ and $\alpha_2$ mononically decreases in $\tau$. We discuss finding such a $\tau^\star$ in Appendix 5.D. This choice allows for a direct comparison of the resulting certified accuracy curves without considering the non-sensitive class clean accuracy.

With the above formalization in place, the goal at hand is two-fold: 1) develop a classification architecture tailored for the asymmetric setting with high robustness, as characterized by the surface $\Gamma$, and 2) provide efficient methods for computing the certified robust radii $r(x)$ used to generate $\Gamma$.

### Contributions

We tackle the above two goals by proposing *feature-convex neural networks* and achieve the following contributions:

1. We exploit the feature-convex structure of the proposed classifier to provide asymmetrically tailored closed-form class 1 certified robust radii for arbitrary $\ell_p$-norms, solving the second goal above and yielding efficient computation of $\Gamma$.

2. We characterize the decision region geometry of convex classifiers, extend the universal approximation theorem for input-convex ReLU neural networks to the classification setting, and show that convex classifiers are sufficiently expressive for high-dimensional data.

3. We evaluate against several baselines on MNIST 3-8 [84], Malimg malware classification [106], Fashion-MNIST shirts [156], and CIFAR-10 cats-dogs [80], and show that our classifiers yield certified robust radii competitive with the state-of-the-art, empirically addressing the first goal listed above.

## Related Works

### Certified Adversarial Robustness

As mentioned in previous chapters, three of the most popular approaches for generating robustness certificates are Lipschitz-based bounds, randomized smoothing, and optimization-based methods. Successfully bounding the Lipschitz constant of a neural network can give rise to an efficient certified radius of robustness, e.g., via the methods proposed in Hein and Andriushchenko [65]. However, in practice such Lipschitz constants are too large to yield meaningful certificates, or it is computationally burdensome to compute or bound the Lipschitz constants in the first place [139, 51, 160]. To overcome these computational limitations, certain methods impose special structures on their model layers to provide immediate Lipschitz guarantees. Specifically, Trockman and Kolter [136] uses the Cayley transform to derive convolutional layers with immediate $\ell_2$-Lipschitz constants, and Zhang et al. [165] introduces a $\ell_\infty$-distance neuron that provides similar Lipschitz guarantees with respect to the $\ell_\infty$-norm. We compare with both these approaches in our numerical simulations of this chapter.

Randomized smoothing, popularized by Lecuyer et al. [85], Li et al. [88], and Cohen, Rosenfeld, and Kolter [35], uses the expected prediction of a model when subjected to Gaussian input noise. These works derive $\ell_2$-norm balls around inputs on which the smoothed classifier remains constant, but suffer from nondeterminism and high computational burden. Follow-up works generalize randomized smoothing to certify input regions defined by different metrics, e.g., Wasserstein, $\ell_1$-, and $\ell_\infty$-norms [86, 133, 158]. Other works focus on enlarging the certified regions by optimizing the smoothing distribution [164, 46, 7], incorporating adversarial training into the base classifier [122, 166], and employing dimensionality reduction at the input [112].

Optimization-based certificates typically seek to derive a tractable over-approximation of the set of possible outputs when the input is subject to adversarial perturbations, and show that this over-approximation is safe. Various over-approximations have been proposed, e.g., based on linear programming and bounding [150, 148], semidefinite programming [116], and branch-and-bound [4, 94, 143]. The $\alpha, \beta$-CROWN method [143] uses an efficient bound propagation to linearly bound the neural network output in conjunction with a per-neuron branching heuristic to achieve state-of-the-art certified radii, winning both the 2021 and the

2022 VNN certification competitions [18, 102]. In contrast to optimization-based methods, our approach in this chapter directly exploits the convex structure of input-convex neural networks to derive closed-form robustness certificates, altogether avoiding any efficiency-tightness tradeoffs.

### Input-Convex Neural Networks

Input-convex neural networks, popularized by Amos, Xu, and Kolter [3], are a class of parameterized models whose input-output mapping is convex. The authors develop tractable methods to learn input-convex neural networks, and show that such models yield state-of-the-art results in a variety of domains where convexity may be exploited, e.g., optimization-based inference. Subsequent works propose novel applications of input-convex neural networks in areas such as optimal control and reinforcement learning [34, 163], optimal transport [96], and optimal power flow [33, 169]. Other works have generalized input-convex networks to input-invex networks [124, 108] and global optimization networks [172] so as to maintain the benign optimization properties of input-convexity. The authors of Siahkamari et al. [126] present algorithms for efficiently learning convex functions, while Chen, Shi, and Zhang [34] and Kim and Kim [76] derive universal approximation theorems for input-convex neural networks in the convex regression setting. The work Sivaprasad et al. [127] shows that input-convex neural networks do not suffer from overfitting, and generalize better than multilayer perceptrons on common benchmark datasets. In this chapter, we incorporate input-convex neural networks as a part of our feature-convex architecture and leverage convexity properties to derive novel robustness guarantees.

## Notations

The sets of natural numbers, real numbers, and nonnegative real numbers are denoted by $\mathbb{N}$, $\mathbb{R}$, and $\mathbb{R}_+$ respectively. The $d \times d$ identity matrix is written as $I_d \in \mathbb{R}^{d\times d}$, and the identity map on $\mathbb{R}^d$ is denoted by $\mathrm{Id}\colon x \mapsto x$. For $A \in \mathbb{R}^{n\times d}$, we define $|A| \in \mathbb{R}^{n\times d}$ by $|A|_{ij} = |A_{ij}|$ for all $i, j$, and we write $A \geq 0$ if and only if $A_{ij} \geq 0$ for all $i, j$. The $\ell_p$-norm on $\mathbb{R}^d$ is given by $\|\cdot\|_p\colon x \mapsto (|x_1|^p + \cdots + |x_d|^p)^{1/p}$ for $p \in [1, \infty)$ and by $\|\cdot\|_p\colon x \mapsto \max\{|x_1|, \ldots, |x_d|\}$ for $p = \infty$. The dual norm of $\|\cdot\|_p$ is denoted by $\|\cdot\|_{p,*}$. The convex hull of a set $X \subseteq \mathbb{R}^d$ is denoted by $\mathrm{conv}(X)$. The subdifferential of a convex function $g\colon \mathbb{R}^d \to \mathbb{R}$ at $x \in \mathbb{R}^d$ is denoted by $\partial g(x)$. If $\epsilon\colon \Omega \to \mathbb{R}^d$ is a random variable on a probability space $(\Omega, \mathcal{B}, \mathbb{P})$ and $P$ is a predicate defined on $\mathbb{R}^d$, then we write $\mathbb{P}(P(\epsilon))$ to mean $\mathbb{P}(\{\omega \in \Omega : P(\epsilon(\omega))\})$. Lebesgue measure on $\mathbb{R}^d$ is denoted by $m$. We define $\mathrm{ReLU}\colon \mathbb{R} \to \mathbb{R}$ as $\mathrm{ReLU}(x) = \max\{0, x\}$, and if $x \in \mathbb{R}^d$, $\mathrm{ReLU}(x)$ denotes $(\mathrm{ReLU}(x_1), \ldots, \mathrm{ReLU}(x_d))$. We recall the threshold function $T_\tau\colon \mathbb{R} \to \{1, 2\}$ defined by (5.1), and we define $T = T_0$. For a function $\varphi\colon \mathbb{R}^d \to \mathbb{R}^q$ and $p \in [1, \infty]$, we define $\mathrm{Lip}_p(\varphi) = \inf\{K \geq 0 : \|\varphi(x) - \varphi(x')\|_p \leq K\|x - x'\|_p \text{ for all } x, x' \in \mathbb{R}^d\}$, and if $\mathrm{Lip}_p(\varphi) < \infty$ we say that $\varphi$ is Lipschitz continuous with constant $\mathrm{Lip}_p(\varphi)$ (with respect to the $\ell_p$-norm).

## 5.2 Feature-Convex Classifiers

Let $d, q \in \mathbb{N}$ and $p \in [1, \infty]$ be fixed, and consider the task of classifying inputs from a subset of $\mathbb{R}^d$ into a fixed set of classes $\mathcal{Y} \subseteq \mathbb{N}$. In what follows, we restrict to the binary setting where $\mathcal{Y} = \{1, 2\}$ and class 1 is the sensitive class for which we desire robustness certificates (Section 5.1). In Appendix 5.A, we briefly discuss avenues to generalize our framework to multiclass settings using one-versus-all and sequential classification methodologies and provide a proof-of-concept example for the Malimg dataset.

We now formally define the classifiers considered in this chapter. Note that the classification threshold $\tau$ discussed in Section 5.1 is omitted for simplicity.

**Definition 10.** Let $f\colon \mathbb{R}^d \to \{1, 2\}$ be defined by $f(x) = T(g(\varphi(x)))$ for some $\varphi\colon \mathbb{R}^d \to \mathbb{R}^q$ and some $g\colon \mathbb{R}^q \to \mathbb{R}$. Then $f$ is said to be a *feature-convex classifier* if the *feature map* $\varphi$ is Lipschitz continuous with constant $\mathrm{Lip}_p(\varphi) < \infty$ and $g$ is a convex function.

We denote the class of all feature-convex classifiers by $\mathcal{F}$. Furthermore, for $q = d$, the subclass of all feature-convex classifiers with $\varphi = \mathrm{Id}$ is denoted by $\mathcal{F}_{\mathrm{Id}}$.

As we will see in Section 5.3, defining our classifiers using the composition of a convex classifier with a Lipschitz feature map enables the fast computation of certified regions in the input space. This naturally arises from the global underestimation of convex functions by first-order Taylor approximations. Since sublevel sets of such $g$ are restricted to be convex, the feature map $\varphi$ is included to increase the representation power of our architecture (see Appendix 5.B for a motivating example). In practice, we find that it suffices to choose $\varphi$ to be a simple map with a small closed-form Lipschitz constant. For example, in our numerical simulations that follow with $q = 2d$, we choose $\varphi(x) = (x - \mu, |x - \mu|)$ with a constant channel-wise dataset mean $\mu$, yielding $\mathrm{Lip}_1(\varphi) \leq 2$, $\mathrm{Lip}_2(\varphi) \leq \sqrt{2}$, and $\mathrm{Lip}_\infty(\varphi) \leq 1$. Although this particular choice of $\varphi$ is convex, the function $g$ need not be monotone, and therefore the composition $g \circ \varphi$ is nonconvex in general. The prediction and certification of feature-convex classifiers are illustrated in Figure 5.1b.

In practice, we implement feature-convex classifiers using parameterizations of $g$, which we now make explicit. Following Amos, Xu, and Kolter [3], we instantiate $g$ as a neural network with nonnegative weight matrices and nondecreasing convex nonlinearities. Specifically, we consider ReLU nonlinearities, which is not restrictive, as our universal approximation result in Theorem 13 proves.

**Definition 11.** A *feature-convex* ReLU *neural network* is a function $\hat{f}\colon \mathbb{R}^d \to \{1, 2\}$ defined by $\hat{f}(x) = T(\hat{g}(\varphi(x)))$ with $\varphi\colon \mathbb{R}^d \to \mathbb{R}^q$ Lipschitz continuous with constant $\mathrm{Lip}_p(\varphi) < \infty$ and $\hat{g}\colon \mathbb{R}^q \to \mathbb{R}$ defined by

$$\begin{aligned} \hat{g}(x^{(0)}) &= A^{(L)}x^{(L-1)} + b^{(L)} + C^{(L)}x^{(0)}, \\ x^{(l)} &= \mathrm{ReLU}\left(A^{(l)}x^{(l-1)} + b^{(l)} + C^{(l)}x^{(0)}\right), \end{aligned}$$

for all $l \in \{1, 2, \ldots, L-1\}$ for some $L \in \mathbb{N}$, $L > 1$, and for some consistently sized matrices $A^{(l)}, C^{(l)}$ and vectors $b^{(l)}$ satisfying $A^{(l)} \geq 0$ for all $l \in \{2, 3, \ldots, L\}$.

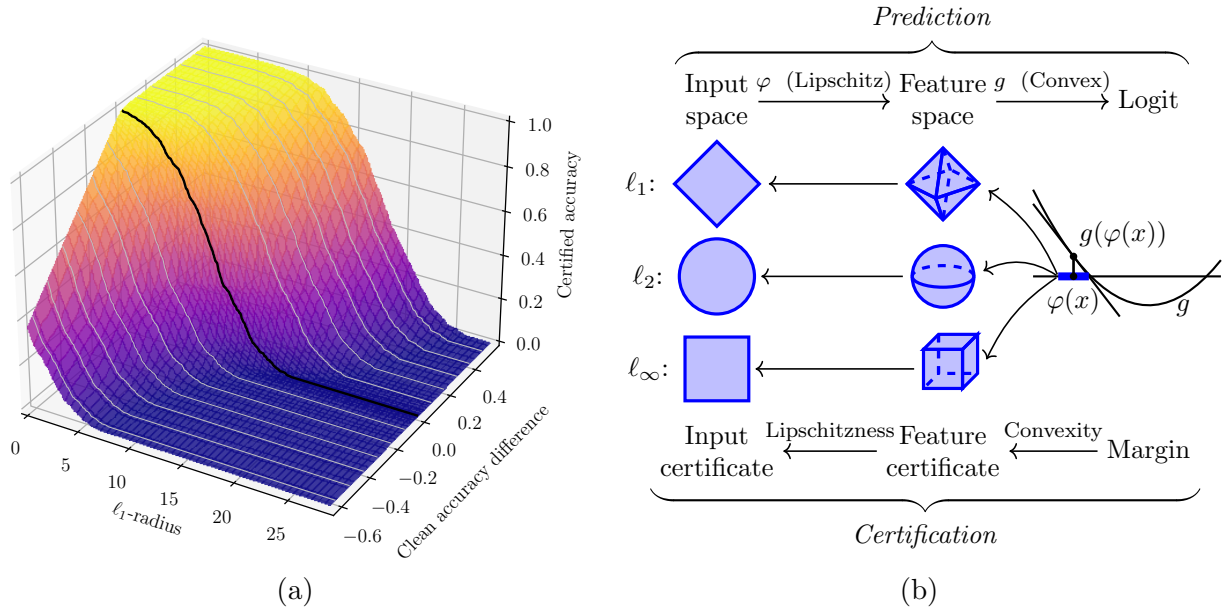



Figure 5.1: (a) The asymmetric certified accuracy surface $\Gamma(r, \tau)$ for MNIST 3-8, as described in Section 5.1. The "clean accuracy difference" axis plots $\alpha_1(\tau) - \alpha_2(\tau)$, and the black line highlights the certified robustness curve for when clean accuracy is equal across the two classes. (b) Illustration of feature-convex classifiers and their certification. Since $g$ is convex, it is globally underapproximated by its tangent plane at $\varphi(x)$, yielding certified sets for norm balls in the higher-dimensional feature space. Lipschitzness of $\varphi$ then yields appropriately scaled certificates in the original input space.

Going forward, we denote the class of all feature-convex ReLU neural networks by $\hat{\mathcal{F}}$. Furthermore, if $q = d$, the subclass of all feature-convex ReLU neural networks with $\varphi = \mathrm{Id}$ is denoted by $\hat{\mathcal{F}}_{\mathrm{Id}}$, which corresponds to the input-convex ReLU neural networks proposed in Amos, Xu, and Kolter [3].

For every $\hat{f} \in \hat{\mathcal{F}}$, it holds that $\hat{g}$ is convex due to the rules for composition and nonnegatively weighted sums of convex functions [24, Section 3.2], and therefore $\hat{\mathcal{F}} \subseteq \mathcal{F}$ and $\hat{\mathcal{F}}_{\mathrm{Id}} \subseteq \mathcal{F}_{\mathrm{Id}}$. The "passthrough" weights $C^{(l)}$ were originally included by Amos, Xu, and Kolter [3] to improve the practical performance of the architecture. In some of our more challenging numerical simulations that follow, we remove these passthrough operations and instead add residual identity mappings between hidden layers, which also preserves convexity. We note that the transformations defined by $A^{(l)}$ and $C^{(l)}$ can be taken to be convolutions, which are nonnegatively weighted linear operations and thus preserve convexity [3].

## 5.3 Certification and Analysis of Feature-Convex Classifiers

We present our main theoretical results in this section. First, we derive asymmetric robustness certificates (Theorem 12) for our feature-convex classifiers. Then, we introduce the notion of convexly separable sets in order to theoretically characterize the representation power of our classifiers. Our primary representation results give a universal function approximation theorem for our classifiers with $\varphi = \mathrm{Id}$ and ReLU activation functions (Theorem 13) and show that such classifiers can perfectly fit convexly separable datasets (Theorem 14), including the CIFAR-10 cats-dogs training data (Fact 1). We also show that this strong learning capacity generalizes by proving that feature-convex classifiers can perfectly fit high-dimensional uniformly distributed data with high probability (Theorem 15).

### Certified Robustness Guarantees

In this section, we address the asymmetric certified robustness problem by providing class 1 robustness certificates for feature-convex classifiers $f \in \mathcal{F}$. Such robustness corresponds to proving the absence of false negatives in the case that class 1 represents positives and class 2 represents negatives. For example, if in a malware detection setting class 1 represents malware and class 2 represents non-malware, the following certificate gives a lower bound on the magnitude of the malware file alteration needed in order to misclassify the file as non-malware.

**Theorem 12.** *Let $f \in \mathcal{F}$ be as in Definition 10 and let $x \in f^{-1}(\{1\}) = \{x' \in \mathbb{R}^d : f(x') = 1\}$. If $\nabla g(\varphi(x)) \in \mathbb{R}^q$ is a nonzero subgradient of the convex function $g$ at $\varphi(x)$, then $f(x+\delta) = 1$ for all $\delta \in \mathbb{R}^d$ such that*

$$\|\delta\|_p < r(x) \coloneqq \frac{g(\varphi(x))}{\mathrm{Lip}_p(\varphi)\|\nabla g(\varphi(x))\|_{p,*}}.$$

*Proof.* Suppose that $\nabla g(\varphi(x)) \in \mathbb{R}^q$ is a nonzero subgradient of $g$ at $\varphi(x)$, so that $g(y) \geq g(\varphi(x)) + \nabla g(\varphi(x))^\top (y - \varphi(x))$ for all $y \in \mathbb{R}^q$. Let $\delta \in \mathbb{R}^d$ be such that $\|\delta\|_p < r(x)$. Then it holds that

$$\begin{aligned} g(\varphi(x+\delta)) &\geq g(\varphi(x)) + \nabla g(\varphi(x))^\top (\varphi(x+\delta) - \varphi(x)) \\ &\geq g(\varphi(x)) - \|\nabla g(\varphi(x))\|_{p,*} \|\varphi(x+\delta) - \varphi(x)\|_p \\ &\geq g(\varphi(x)) - \|\nabla g(\varphi(x))\|_{p,*} \mathrm{Lip}_p(\varphi)\|\delta\|_p \\ &> 0, \end{aligned}$$

so indeed $f(x+\delta) = 1$. □

*Remark* 6. For $f \in \mathcal{F}$ and $x \in f^{-1}(\{1\})$, a subgradient $\nabla g(\varphi(x)) \in \mathbb{R}^q$ of $g$ always exists at $\varphi(x)$, since the subdifferential $\partial g(\varphi(x))$ is a nonempty closed bounded convex set, as $g$ is a

finite convex function on all of $\mathbb{R}^q$—see Theorem 23.4 in Rockafellar [119] and the discussion thereafter. Furthermore, if $f$ is not a constant classifier, such a subgradient $\nabla g(\varphi(x))$ must necessarily be nonzero, since, if it were zero, then $g(y) \geq g(\varphi(x)) + \nabla g(\varphi(x))^\top (y - \varphi(x)) = g(\varphi(x)) > 0$ for all $y \in \mathbb{R}^q$, implying that $f$ identically predicts class 1, which is a contradiction. Thus, the certified radius given in Theorem 12 is always well-defined in practical settings.

Theorem 12 is derived from the fact that a convex function is globally underapproximated by any tangent plane. The nonconstant terms in Theorem 12 afford an intuitive interpretation: the radius scales proportionally to the confidence $g(\varphi(x))$ and inversely with the input sensitivity $\|\nabla g(\varphi(x))\|_{p,*}$. In practice, $\mathrm{Lip}_p(\varphi)$ can be made quite small as mentioned in Section 5.2, and furthermore the subgradient $\nabla g(\varphi(x))$ is easily evaluated as the Jacobian of $g$ at $\varphi(x)$ using standard automatic differentiation packages. This provides fast, deterministic class 1 certificates for any $\ell_p$-norm without modification of the feature-convex network's training procedure or architecture. We emphasize that our robustness certificates of Theorem 12 are independent of the architecture of $f$.

## Representation Power Characterization

We now restrict our analysis to the class $\mathcal{F}_{\mathrm{Id}}$ of feature-convex classifiers with an identity feature map. This can be equivalently considered as the class of classifiers for which the input-to-logit map is convex. We therefore refer to models in $\mathcal{F}_{\mathrm{Id}}$ as *input-convex classifiers*. While the feature map $\varphi$ is useful in boosting the practical performance of our classifiers, the theoretical results in this section suggest that there is significant potential in using input-convex classifiers as a standalone solution.

### Classifying Convexly Separable Sets

We begin by introducing the notion of convexly separable sets, which are intimately related to decision regions representable by the class $\mathcal{F}_{\mathrm{Id}}$.

**Definition 12.** Let $X_1, X_2 \subseteq \mathbb{R}^d$. The ordered pair $(X_1, X_2)$ is said to be *convexly separable* if there exists a nonempty closed convex set $X \subseteq \mathbb{R}^d$ such that $X_2 \subseteq X$ and $X_1 \subseteq \mathbb{R}^d \setminus X$.

Notice that it may be the case that a pair $(X_1, X_2)$ is convexly separable yet the pair $(X_2, X_1)$ is not. Although low-dimensional intuition may raise concerns regarding the convex separability of binary-labeled data, we will soon see in Fact 1 and Theorem 15 that convex separability typically holds in high dimensions. We now show that convexly separable datasets possess the property that they may always be perfectly fit by input-convex classifiers.

**Proposition 17.** *For any nonempty closed convex set $X \subseteq \mathbb{R}^d$, there exists $f \in \mathcal{F}_{\mathrm{Id}}$ such that $X = f^{-1}(\{2\}) = \{x \in \mathbb{R}^d : f(x) = 2\}$. In particular, this shows that if $(X_1, X_2)$ is a*

*convexly separable pair of subsets of* $\mathbb{R}^d$*, then there exists* $f \in \mathcal{F}_{\text{Id}}$ *such that* $f(x) = 1$ *for all* $x \in X_1$ *and* $f(x) = 2$ *for all* $x \in X_2$*.*

*Proof.* Let $X \subseteq \mathbb{R}^d$ be a nonempty closed convex set. By Lemma 9, there exists a convex function $g\colon \mathbb{R}^d \to \mathbb{R}$ such that $X = \{x \in \mathbb{R}^d : g(x) \leq 0\}$. Define $f\colon \mathbb{R}^d \to \{1, 2\}$ by $f(x) = 1$ if $g(x) > 0$ and $f(x) = 2$ if $g(x) \leq 0$. Clearly, it holds that $f \in \mathcal{F}_{\text{Id}}$. Furthermore, for all $x \in X$ it holds that $g(x) \leq 0$, implying that $f(x) = 2$ for all $x \in X$. Conversely, if $x \in \mathbb{R}^d$ is such that $f(x) = 2$, then $g(x) \leq 0$, implying that $x \in X$. Hence, $X = \{x \in \mathbb{R}^d : f(x) = 2\}$.

If $(X_1, X_2)$ is a convexly separable pair of subsets of $\mathbb{R}^d$, then there exists a nonempty closed convex set $X \subseteq \mathbb{R}^d$ such that $X_2 \subseteq X$ and $X_1 \subseteq \mathbb{R}^d \setminus X$, and therefore there exists $f \in \mathcal{F}_{\text{Id}}$ such that $X_2 \subseteq X = f^{-1}(\{2\})$ and $X_1 \subseteq \mathbb{R}^d \setminus X = f^{-1}(\{1\})$, implying that indeed $f(x) = 1$ for all $x \in X_1$ and $f(x) = 2$ for all $x \in X_2$. □

We also show that the converse of Proposition 17 holds: the geometry of the decision regions of classifiers in $\mathcal{F}_{\text{Id}}$ consists of a convex set and its complement.

**Proposition 18.** *Let* $f \in \mathcal{F}_{\text{Id}}$*. The decision region under* $f$ *associated to class* 2*, namely* $X \coloneqq f^{-1}(\{2\}) = \{x \in \mathbb{R}^d : f(x) = 2\}$*, is a closed convex set.*

*Proof.* For all $x \in \mathbb{R}^d$, it holds that $f(x) = 2$ if and only if $g(x) \leq 0$. Since $f \in \mathcal{F}_{\text{Id}}$, $g$ is convex, and hence, $X = \{x \in \mathbb{R}^d : g(x) \leq 0\}$ is a (nonstrict) sublevel set of a convex function and is therefore a closed convex set. □

Note that this is not necessarily true for our more general feature-convex architectures with $\varphi \neq \text{Id}$. We continue our theoretical analysis of input-convex classifiers by extending the universal approximation theorem for regressing upon real-valued convex functions (given in Chen, Shi, and Zhang [34]) to the classification setting. In particular, Theorem 13 below shows that any input-convex classifier $f \in \mathcal{F}_{\text{Id}}$ can be approximated arbitrarily well on any compact set by ReLU neural networks with nonnegative weights. Here, "arbitrarily well" means that the set of inputs where the neural network prediction differs from that of $f$ can be made to have arbitrarily small Lebesgue measure.

**Theorem 13.** *For any* $f \in \mathcal{F}_{\text{Id}}$*, any compact convex subset* $X$ *of* $\mathbb{R}^d$*, and any* $\epsilon > 0$*, there exists* $\hat{f} \in \hat{\mathcal{F}}_{\text{Id}}$ *such that* $m(\{x \in X : \hat{f}(x) \neq f(x)\}) < \epsilon$*.*

*Proof.* Let $f \in \mathcal{F}_{\text{Id}}$ and let $X$ be a compact convex subset of $\mathbb{R}^d$. By Lemma 12, there exists a sequence $\{\hat{f}_n \in \hat{\mathcal{F}}_{\text{Id}} : n \in \mathbb{N}\} \subseteq \hat{\mathcal{F}}_{\text{Id}}$ such that $\hat{g}_n(x) < \hat{g}_{n+1}(x) < g(x)$ for all $x \in X$ and all $n \in \mathbb{N}$ and $\hat{g}_n$ converges uniformly to $g$ on $X$ as $n \to \infty$. Fix this sequence.

For all $n \in \mathbb{N}$, define

$$E_n = \{x \in X : \hat{f}_n(x) \neq f(x)\},$$

i.e., the set of points in $X$ for which the classification under $\hat{f}_n$ does not agree with that under $f$. Since $\hat{g}_n(x) < g(x)$ for all $x \in X$ and all $n \in \mathbb{N}$, we see that

$$\begin{aligned} E_n &= \{x \in X : \hat{g}_n(x) > 0 \text{ and } g(x) \leq 0\} \cup \{x \in X : \hat{g}_n(x) \leq 0 \text{ and } g(x) > 0\} \\ &= \{x \in X : \hat{g}_n(x) \leq 0 \text{ and } g(x) > 0\}. \end{aligned}$$

Since $g$ is a real-valued convex function on $\mathbb{R}^d$, it is continuous [119, Corollary 10.1.1], and therefore $g^{-1}((0,\infty)) = \{x \in \mathbb{R}^d : g(x) > 0\}$ is measurable. Similarly, $\hat{g}_n^{-1}((-\infty, 0]) = \{x \in \mathbb{R}^d : \hat{g}_n(x) \leq 0\}$ is also measurable for all $n \in \mathbb{N}$ since $\hat{g}_n$ is continuous. Furthermore, $X$ is measurable as it is compact. Therefore, $E_n$ is measurable for all $n \in \mathbb{N}$. Now, since $\hat{g}_n(x) < \hat{g}_{n+1}(x)$ for all $x \in X$ and all $n \in \mathbb{N}$, it holds that $E_{n+1} \subseteq E_n$ for all $n \in \mathbb{N}$. It is clear that to prove the result, it suffices to show that $\lim_{n\to\infty} m(E_n) = 0$. Therefore, if we show that $m\left(\bigcap_{n\in\mathbb{N}} E_n\right) = 0$, then the fact that $m(E_1) \leq m(X) < \infty$ together with Lebesgue measure's continuity from above yields that $\lim_{n\to\infty} m(E_n) = 0$, thereby proving the result.

It remains to be shown that $m\left(\bigcap_{n\in\mathbb{N}} E_n\right) = 0$. To this end, suppose for the sake of contradiction that $\bigcap_{n\in\mathbb{N}} E_n \neq \emptyset$. Then there exists $x \in \bigcap_{n\in\mathbb{N}} E_n$, meaning that $g(x) > 0$ and $\hat{g}_n(x) \leq 0$ for all $n \in \mathbb{N}$. Thus, for this $x \in X$, we find that $\limsup_{n\to\infty} \hat{g}_n(x) \leq 0 < g(x)$, which contradicts the fact that $\hat{g}_n$ uniformly converges to $g$ on $X$. Therefore, it must be that $\bigcap_{n\in\mathbb{N}} E_n = \emptyset$, and thus $m\left(\bigcap_{n\in\mathbb{N}} E_n\right) = 0$, which concludes the proof. □

An extension of the proof of Theorem 13 combined with Proposition 17 yields that input-convex ReLU neural networks can perfectly fit convexly separable sampled datasets.

**Theorem 14.** *If $(X_1, X_2)$ is a convexly separable pair of finite subsets of $\mathbb{R}^d$, then there exists $\hat{f} \in \hat{\mathcal{F}}_{\mathrm{Id}}$ such that $\hat{f}(x) = 1$ for all $x \in X_1$ and $\hat{f}(x) = 2$ for all $x \in X_2$.*

*Proof.* Throughout this proof, we denote the complement of a set $Y \subseteq \mathbb{R}^d$ by $Y^c = \mathbb{R}^d \setminus Y$.

Suppose that $X_1 = \{x^{(1)}, \ldots, x^{(M)}\} \subseteq \mathbb{R}^d$ and $X_2 = \{y^{(1)}, \ldots, y^{(N)}\} \subseteq \mathbb{R}^d$ are such that $(X_1, X_2)$ is convexly separable. Then, by definition of convex separability, there exists a nonempty closed convex set $X' \subseteq \mathbb{R}^d$ such that $X_2 \subseteq X'$ and $X_1 \subseteq \mathbb{R}^d \setminus X'$. Let $X = X' \cap \mathrm{conv}(X_2)$. Since $X_2 \subseteq X'$ and both sets $X'$ and $\mathrm{conv}(X_2)$ are convex, the set $X$ is nonempty and convex. By finiteness of $X_2$, the set $\mathrm{conv}(X_2)$ is compact, and therefore by closedness of $X'$, the set $X$ is compact and hence closed.

By Proposition 17, there exists $f \in \mathcal{F}_{\mathrm{Id}}$ such that $f^{-1}(\{2\}) = X$. Since $\mathrm{conv}(X_1 \cup X_2)$ is compact and convex, Lemma 12 gives that there exists a sequence $\{\hat{f}_n \in \hat{\mathcal{F}}_{\mathrm{Id}} : n \in \mathbb{N}\} \subseteq \hat{\mathcal{F}}_{\mathrm{Id}}$ such that $\hat{g}_n(x) < \hat{g}_{n+1}(x) < g(x)$ for all $x \in \mathrm{conv}(X_1 \cup X_2)$ and all $n \in \mathbb{N}$ and $\hat{g}_n$ converges uniformly to $g$ on $\mathrm{conv}(X_1 \cup X_2)$ as $n \to \infty$. Fix this sequence.

Let $x \in X_2$. Then, since $X_2 \subseteq X'$ and $X_2 \subseteq \mathrm{conv}(X_2)$, it holds that $x \in X' \cap \mathrm{conv}(X_2) = X = f^{-1}(\{2\})$, implying that $f(x) = 2$ and hence $g(x) \leq 0$. Since $\hat{g}_n(x) < g(x)$ for all $n \in \mathbb{N}$, this shows that $\hat{f}_n(x) = 2$ for all $n \in \mathbb{N}$. On the other hand, let $i \in \{1, \ldots, M\}$ and consider $x = x^{(i)} \in X_1$. Since $X_1 \subseteq \mathbb{R}^d \setminus X' = \mathbb{R}^d \cap (X')^c \subseteq \mathbb{R}^d \cap (X' \cap \mathrm{conv}(X_2))^c = \mathbb{R}^d \cap X^c = \mathbb{R}^d \cap f^{-1}(\{1\})$, it holds that $f(x) = 1$ and thus $g(x) > 0$. Suppose for the sake of contradiction that $\hat{f}_n(x) = 2$ for all $n \in \mathbb{N}$. Then $\hat{g}_n(x) \leq 0$ for all $n \in \mathbb{N}$. Therefore, for this $x \in X_1$, we find that $\limsup_{n\to\infty} \hat{g}_n(x) \leq 0 < g(x)$, which contradicts the fact that $\hat{g}_n$ uniformly converges to $g$ on $\mathrm{conv}(X_1 \cup X_2)$. Therefore, it must be that there exists $n_i \in \mathbb{N}$ such that $\hat{f}_{n_i}(x) = 1$, and thus $\hat{g}_{n_i}(x) > 0$. Since $\hat{g}_n(x) < \hat{g}_{n+1}(x)$ for all $n \in \mathbb{N}$, this implies that $\hat{g}_n(x) > 0$ for all $n \geq n_i$. Hence, $\hat{f}_n(x) = \hat{f}_n(x^{(i)}) = 1$ for all $n \geq n_i$.

Let $n^\star$ be the maximum of all such $n_i$, i.e., $n^\star = \max\{n_i : i \in \{1, \dots, M\}\}$. Then the above analysis shows that $\hat{f}_{n^\star}(x) = 2$ for all $x \in X_2$ and that $\hat{f}_{n^\star}(x) = 1$ for all $x \in X_1$. Since $\hat{f}_{n^\star} \in \hat{\mathcal{F}}_{\text{Id}}$, the claim has been proven. □

Theorem 13 and Theorem 14, being specialized to models with ReLU activation functions, theoretically justify the particular parameterization in Definition 11 for learning feature-convex classifiers to fit convexly separable data.

### Empirical Convex Separability

Interestingly, we find empirically that high-dimensional image training data is convexly separable. We illustrate this in Appendix 5.D by attempting to reconstruct a CIFAR-10 cat image from a convex combination of the dogs and vice versa; the error is significantly positive for *every* sample in the training dataset, and image reconstruction is visually poor. This fact, combined with Theorem 14, immediately yields the following result.

**Fact 1.** *There exists $\hat{f} \in \hat{\mathcal{F}}_{\text{Id}}$ such that $\hat{f}$ achieves perfect training accuracy for the unaugmented CIFAR-10 cats-versus-dogs dataset.*

The gap between this theoretical guarantee and our practical performance is large; without the feature map, our CIFAR-10 cats-dogs classifier achieves just 73.4% training accuracy (Table 5.4). While high training accuracy does not necessarily imply strong test set performance, Fact 1 demonstrates that the typical deep learning paradigm of overfitting to the training dataset is theoretically attainable [105]. We thus posit that there is substantial room for improvement in the design and optimization of input-convex classifiers. We leave the challenge of overfitting to the CIFAR-10 cats-dogs training data with an input-convex classifier as an open research problem for the field.

**Open Problem 1.** *Learn an input-convex* ReLU *neural network that achieves* 100% *training accuracy on the unaugmented CIFAR-10 cats-versus-dogs dataset.*

### Convex Separability in High Dimensions

We conclude by investigating *why* the convex separability property that allows for Fact 1 may hold for natural image datasets. We argue that dimensionality facilitates this phenomenon by showing that data is easily separated by some $f \in \hat{\mathcal{F}}_{\text{Id}}$ when $d$ is sufficiently large. In particular, although it may seem restrictive to rely on models in $\hat{\mathcal{F}}_{\text{Id}}$ with convex class 2 decision regions, we show in Theorem 15 below that even uninformative data distributions that are seemingly difficult to classify may be fit by such models with high probability as the dimensionality of the data increases.

**Theorem 15.** *Consider $M, N \in \mathbb{N}$. Let $X_1 = \{x^{(1)}, \dots, x^{(M)}\}$ and $X_2 = \{y^{(1)}, \dots, y^{(N)}\}$ be subsets of $\mathbb{R}^d$ with all samples $x_k^{(i)}, y_l^{(j)}$ drawn independently and identically from the uniform*

*probability distribution on* $[-1,1]$*. Then, it holds that*

$$\mathbb{P}\left((X_1, X_2) \text{ is convexly separable}\right) \geq \begin{cases} 1 - \left(1 - \frac{M!N!}{(M+N)!}\right)^d & \text{for all } d \in \mathbb{N}, \\ 1 & \text{if } d \geq M+N. \end{cases} \tag{5.2}$$

*In particular,* $\hat{\mathcal{F}}_{\mathrm{Id}}$ *contains an input-convex* ReLU *neural network that classifies all* $x^{(i)}$ *into class* 1 *and all* $y^{(j)}$ *into class* 2 *almost surely for sufficiently large dimensions* $d$*.*

*Proof.* Throughout the proof, we denote the cardinality of a set $S$ by $|S|$. For the reader's convenience, we also recall that, for $n \in \mathbb{N}$, the symmetric group $S_n$ consists of all permutations (i.e., bijections) on the set $\{1, 2, \ldots, n\}$, and that $|S_n| = n!$. If $\sigma\colon \{1, 2, \ldots, n\} \to \{1, 2, \ldots, n\}$ is a permutation in $S_n$, we denote the restriction of $\sigma$ to the domain $I \subseteq \{1, 2, \ldots, n\}$ by $\sigma|_I\colon I \to \{1, 2, \ldots, n\}$, which we recall is defined by $\sigma|_I(i) = \sigma(i)$ for all $i \in I$, and is not necessarily a permutation on $I$ in general.

Consider first the case where $d \geq M + N$. Let $b \in \mathbb{R}^{M+N}$ be the vector defined by $b_i = 1$ for all $i \in \{1, \ldots, M\}$ and $b_i = -1$ for all $i \in \{M+1, \ldots, M+N\}$. Then, since $x_k^{(i)}, y_l^{(j)}$ are independent uniformly distributed random variables on $[-1, 1]$, it holds that the matrix

$$\begin{bmatrix} {x^{(1)}}^\top \\ \vdots \\ {x^{(M)}}^\top \\ {y^{(1)}}^\top \\ \vdots \\ {y^{(N)}}^\top \end{bmatrix} \in \mathbb{R}^{(M+N)\times d}$$

has rank $M + N$ almost surely, and therefore the linear system of equations

$$\begin{bmatrix} {x^{(1)}}^\top \\ \vdots \\ {x^{(M)}}^\top \\ {y^{(1)}}^\top \\ \vdots \\ {y^{(N)}}^\top \end{bmatrix} a = b$$

has a solution $a \in \mathbb{R}^d$ with probability 1, and we note that from this solution we find that $X_2$ is a subset of the nonempty closed convex set $\{x \in \mathbb{R}^d : a^\top x \leq 0\}$ and that $X_1$ is a subset of its complement. Hence, $(X_1, X_2)$ is convexly separable with probability 1 in this case.

Now let us consider the general case: $d \in \mathbb{N}$ and in general it may be the case that $d < M + N$. For notational convenience, let $P$ be the probability of interest:

$$P = \mathbb{P}\left((X_1, X_2) \text{ is convexly separable}\right).$$

Suppose that there exists a coordinate $k \in \{1, 2, \dots, d\}$ such that $x_k^{(i)} < y_k^{(j)}$ for all pairs $(i, j) \in \{1, 2, \dots, M\} \times \{1, 2, \dots, N\}$ and that

$$a \coloneqq \min\{y_k^{(1)}, \dots, y_k^{(N)}\} < \max\{y_k^{(1)}, \dots, y_k^{(N)}\} =: b.$$

Then, let $X = \{x \in \mathbb{R}^d : x_k \in [a, b]\}$. That is, $X$ is the extrusion of the convex hull of the projections $\{y_k^{(1)}, \dots, y_k^{(N)}\}$ along all remaining coordinates. The set $X$ is a nonempty closed convex set, and it is clear by our supposition that $X_2 \subseteq X$ and $X_1 \subseteq \mathbb{R}^d \setminus X$. Therefore, the supposition implies that $(X_1, X_2)$ is convexly separable, and thus

$$\begin{aligned}
P &\geq \mathbb{P}\big(\text{there exists } k \in \{1, 2, \dots, d\} \text{ such that } x_k^{(i)} < y_k^{(j)} \text{ for all pairs } (i, j) \\
&\quad \text{and that } \min\{y_k^{(1)}, \dots, y_k^{(N)}\} < \max\{y_k^{(1)}, \dots, y_k^{(N)}\}\big) \\
&= 1 - \mathbb{P}\big(\text{for all } k \in \{1, 2, \dots, d\}, \text{ it holds that } x_k^{(i)} \geq y_k^{(j)} \text{ for some pair } (i, j) \\
&\quad \text{or that } \min\{y_k^{(1)}, \dots, y_k^{(N)}\} = \max\{y_k^{(1)}, \dots, y_k^{(N)}\}\big) \\
&= 1 - \prod_{k=1}^{d} \mathbb{P}\left(x_k^{(i)} \geq y_k^{(j)} \text{ for some pair } (i, j) \text{ or } \min\{y_k^{(1)}, \dots, y_k^{(N)}\} = \max\{y_k^{(1)}, \dots, y_k^{(N)}\}\right),
\end{aligned}$$

where the final equality follows from the independence of the coordinates of the samples. Since $\min\{y_k^{(1)}, \dots, y_k^{(N)}\} < \max\{y_k^{(1)}, \dots, y_k^{(N)}\}$ almost surely, we find that

$$\begin{aligned}
P &\geq 1 - \prod_{k=1}^{d} \Bigg( \mathbb{P}(x_k^{(i)} \geq y_k^{(j)} \text{ for some pair } (i, j)) \\
&\qquad + \mathbb{P}(\min\{y_k^{(1)}, \dots, y_k^{(N)}\} = \max\{y_k^{(1)}, \dots, y_k^{(N)}\}) \Bigg) \\
&= 1 - \prod_{k=1}^{d} \mathbb{P}(x_k^{(i)} \geq y_k^{(j)} \text{ for some pair } (i, j)) \\
&= 1 - \prod_{k=1}^{d} \Big(1 - \mathbb{P}(x_k^{(i)} < y_k^{(j)} \text{ for all pairs } (i, j))\Big) \\
&= 1 - \prod_{k=1}^{d} \left(1 - \mathbb{P}\left(\max_{i \in \{1,2,\dots,M\}} x_k^{(i)} < \min_{j \in \{1,2,\dots,N\}} y_k^{(j)}\right)\right) \\
&= 1 - \prod_{k=1}^{d} \left(1 - \mathbb{P}\left((x_k^{(1)}, \dots, x_k^{(M)}, y_k^{(1)}, \dots, y_k^{(N)}) \in \bigcup_{\sigma \in S} E_\sigma\right)\right),
\end{aligned} \tag{5.3}$$

where we define $S$ to be the set of permutations on $\{1, \dots, M + N\}$ whose restriction to $\{1, \dots, M\}$ is also a permutation;

$$S = \left\{\sigma \in S_{M+N} : \sigma|_{\{1,\dots,M\}} \in S_M\right\},$$

and where, for a permutation $\sigma \in S_{M+N}$, $E_\sigma$ is the event where an $(M+N)$-vector has indices ordered according to $\sigma$;

$$E_\sigma = \{z \in \mathbb{R}^{M+N} : z_{\sigma(1)} < \cdots < z_{\sigma(M+N)}\}.$$

We note that the final equality in (5.3) relies on the fact that $\mathbb{P}(x_k^{(i)} = x_k^{(i')}) = \mathbb{P}(y_k^{(j)} = y_k^{(j')}) = 0$ for all $i' \neq i$ and all $j' \neq j$, which is specific to our uniform distribution at hand.

Now, since $E_\sigma, E_{\sigma'}$ are disjoint for distinct permutations $\sigma, \sigma' \in S_{M+N}$, the bound (5.3) gives that

$$P \geq 1 - \prod_{k=1}^{d} \left(1 - \sum_{\sigma \in S} \mathbb{P}((x_k^{(1)}, \ldots, x_k^{(M)}, y_k^{(1)}, \ldots, y_k^{(N)}) \in E_\sigma)\right). \tag{5.4}$$

Since $x_k^{(1)}, \ldots, x_k^{(M)}, y_k^{(1)}, \ldots, y_k^{(N)}$ are independent and identically distributed samples, they define an exchangeable sequence of random variables, implying that

$$\mathbb{P}((x_k^{(1)}, \ldots, x_k^{(M)}, y_k^{(1)}, \ldots, y_k^{(N)}) \in E_\sigma) = \mathbb{P}(x_k^{(1)} < \cdots < x_k^{(M)} < y_k^{(1)} < \cdots < y_k^{(N)})$$

for all permutations $\sigma \in S_{M+N}$. Since, under the uniform distribution at hand,

$$(x_k^{(1)}, \ldots, x_k^{(M)}, y_k^{(1)}, \ldots, y_k^{(N)}) \in E_\sigma$$

for some $\sigma \in S_{M+N}$ almost surely, it holds that

$$\begin{aligned} 1 &= \mathbb{P}\left((x_k^{(1)}, \ldots, x_k^{(M)}, y_k^{(N)}, \ldots, y_k^{(N)}) \in \bigcup_{\sigma \in S_{M+N}} E_\sigma\right) \\ &= \sum_{\sigma \in S_{M+N}} \mathbb{P}((x_k^{(1)}, \ldots, x_k^{(M)}, y_k^{(1)}, \ldots, y_k^{(N)}) \in E_\sigma) \\ &= |S_{M+N}|\, \mathbb{P}(x_k^{(1)} < \cdots x_k^{(M)} < y_k^{(1)} < \cdots < y_k^{(N)}). \end{aligned}$$

This implies that

$$\mathbb{P}((x_k^{(1)}, \ldots, x_k^{(M)}, y_k^{(1)}, \ldots, y_k^{(N)}) \in E_\sigma) = \frac{1}{|S_{M+N}|} = \frac{1}{(M+N)!}$$

for all permutations $\sigma \in S_{M+N}$. Hence, our bound (5.4) becomes

$$P \geq 1 - \prod_{k=1}^{d} \left(1 - \frac{|S|}{(M+N)!}\right) = 1 - \left(1 - \frac{|S|}{(M+N)!}\right)^d.$$

Finally, we immediately see that that map $\Gamma\colon S_M \times S_N \to S_{M+N}$ defined by

$$\Gamma(\sigma, \sigma')(i) = \begin{cases} \sigma(i) & \text{if } i \in \{1, \ldots, M\}, \\ \sigma'(i-M) + M & \text{if } i \in \{M+1, \ldots, M+N\}, \end{cases}$$

is injective and has image $S$, implying that $|S| = |S_M \times S_N| = |S_M||S_N| = M!N!$. Thus,

$$P \geq 1 - \left(1 - \frac{M!N!}{(M+N)!}\right)^d,$$

which proves (5.2).

The unit probability of $\hat{\mathcal{F}}_{\text{Id}}$ containing a classifier that classifies all $x^{(i)}$ into class 1 and all $y^{(j)}$ into class 2 for large $d$ follows immediately from Theorem 14. □

Although the uniformly distributed data in Theorem 15 is unrealistic in practice, the result demonstrates that the class $\hat{\mathcal{F}}_{\text{Id}}$ of input-convex ReLU neural networks has sufficient complexity to fit even the most unstructured data in high dimensions. Despite this ability, researchers have found that current input-convex neural networks tend to not overfit in practice, yielding small generalization gaps relative to conventional neural networks [127]. Achieving the modern deep learning paradigm of overfitting to the training dataset with input-convex networks is an exciting open challenge [105].

## 5.4 Numerical Simulations

This section compares our feature-convex classifiers against a variety of state-of-the-art baselines in the asymmetric setting. Before discussing the results, we briefly describe the datasets, baselines, and architectures used. For a more in-depth description and hyperparameter details, see Appendix 5.D.

### Datasets

We use four datasets. First, we consider distinguishing between $28 \times 28$ greyscale MNIST digits 3 and 8 [84], which are generally more visually similar and challenging to distinguish than other digit pairs. Next, we consider identifying malware from the "Allaple.A" class in the Malimg dataset of $512 \times 512$ bytewise encodings of malware [106]. Next, we consider distinguishing between shirts and T-shirts in the Fashion-MNIST dataset of $28 \times 28$ greyscale images [156], which tend to be the hardest classes to distinguish [74]. Finally, we consider the $32 \times 32$ RGB CIFAR-10 cat and dog images since they are relatively difficult to distinguish [61, 90, 114]. The latter two datasets can be considered as our more challenging settings. All pixel values are normalized into the interval $[0, 1]$.

### Baseline Methods

We consider several state-of-the-art randomized and deterministic baselines. For all datasets, we evaluate the randomized smoothing certificates of Yang et al. [158] for the Gaussian, Laplacian, and uniform distributions trained with noise augmentation (denoted RS Gaussian, RS Laplacian, and RS Uniform, respectively), as well as the deterministic bound propagation

framework $\alpha, \beta$-CROWN [143], which is scatter plotted since certification is only reported as a binary answer at a given radius. We also evaluate, when applicable, deterministic certified methods for each norm ball. These include the splitting-noise $\ell_1$-certificates from Levine and Feizi [87] (denoted Splitting), the orthogonality-based $\ell_2$-certificates from Trockman and Kolter [136] (denoted Cayley), and the $\ell_\infty$-distance-based $\ell_\infty$-certificates from Zhang et al. [165] (denoted $\ell_\infty$-Net). The last two deterministic methods are not evaluated on the large-scale Malimg dataset due to their prohibitive runtime. Furthermore, the $\ell_\infty$-Net was unable to significantly outperform a random classifier on the CIFAR-10 cats-dogs dataset, and is therefore only included in the MNIST 3-8 and Fashion-MNIST shirts simulations. Notice that the three randomized smoothing baselines have fundamentally different predictions and certificates than the deterministic methods (including ours), namely, the predictions are random and the certificates hold only with high probability.

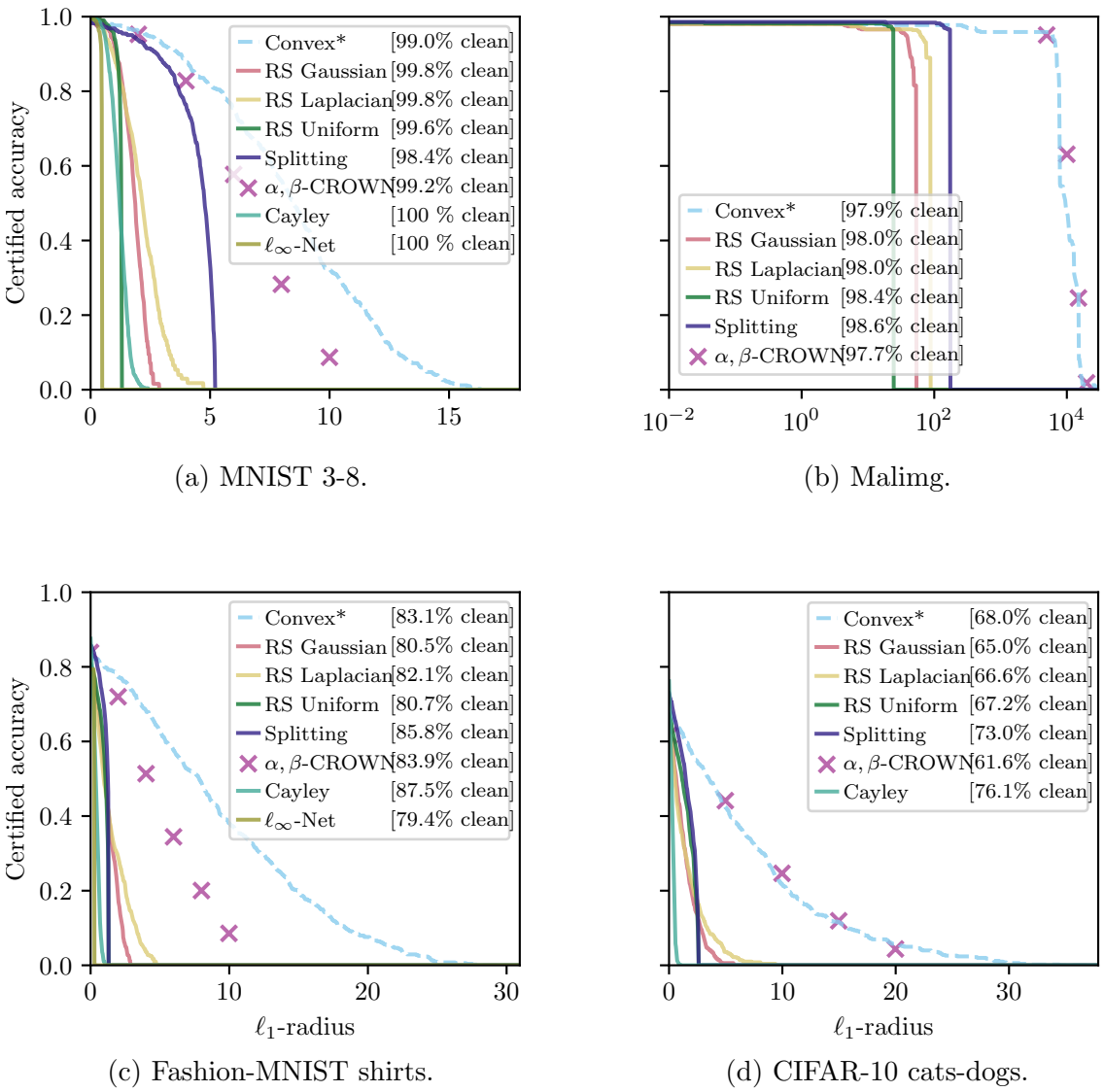
## Feature-Convex Architecture

Our simple experiments (MNIST 3-8 and Malimg) require no feature map to achieve high accuracy ($\varphi = \mathrm{Id}$). The Fashion-MNIST shirts dataset also benefited minimally from the feature map inclusion. For the CIFAR-10 cats-dogs task, we let our feature map be the concatenation $\varphi(x) = (x-\mu, |x-\mu|)$, as motivated by Appendix 5.B, where $\mu$ is the channel-wise dataset mean (e.g., size 3 for an RGB image) broadcasted to the appropriate dimensions. Our MNIST 3-8 and Malimg architecture then consists of a simple two-hidden-layer input-convex multilayer perceptron with $(n_1, n_2) = (200, 50)$ hidden features, ReLU nonlinearities, and passthrough weights. For the Fashion-MNIST shirts (CIFAR-10 cats-dogs, resp.) dataset, we use a convex ConvNet architecture consisting of 3 (5, resp.) convolutional, BatchNorm, and ReLU layers. All models are trained using SGD on the standard binary cross entropy loss with Jacobian regularization, and clean accuracies are balanced as described in Section 5.1 and Appendix 5.D to ensure a fair comparison of different robustness certificates.

## Results and Discussion

Experimental results for $\ell_1$-norm certification are reported in Figure 5.2, where our feature-convex classifier radii, denoted by Convex*, are similar or better than all other baselines across all datasets. Also reported is each method's clean test accuracy without any attacks, denoted by "clean." We defer the corresponding plots for $\ell_2$- and $\ell_\infty$-norm balls to Appendix 5.D, where our certified radii are not dominant but still comparable to methods tailored specifically for a particular norm. We accomplish this while maintaining completely deterministic, closed-form certificates with orders-of-magnitude faster computation time than competitive baselines.

For the MNIST 3-8 and Malimg datasets (Figure 5.2a and Figure 5.2b), all methods achieve high clean test accuracy. Our $\ell_1$-radii scale exceptionally well with the dimensionality of the input, with two orders of magnitude improvement over smoothing baselines for the Malimg dataset. The Malimg certificates in particular have an interesting concrete interpre-




(a) MNIST 3-8.

(b) Malimg.

(c) Fashion-MNIST shirts.

(d) CIFAR-10 cats-dogs.

Figure 5.2: Class 1 certified radii curves for the $\ell_1$-norm. Note the log-scale on the Malimg plot.

tation. As each pixel corresponds to one byte in the original malware file, an $\ell_1$-certificate of radius $r$ provides a robustness certificate for up to $r$ bytes in the file. Namely, even if a malware designer were to arbitrarily change $r$ malware bytes, they would be unable to fool our classifier into returning a false negative. We note that this is primarily illustrative and is unlikely to have an immediate practical impact as small semantic changes (e.g., reordering unrelated instructions) can induce large $\ell_p$-norm shifts.

Table 5.1: Average runtimes (seconds) per input for computing the $\ell_1$-, $\ell_2$-, and $\ell_\infty$-robust radii. $^*$ = our method. $^\dagger$ = per-property verification time. $^\ddagger$ = certified radius computed via binary search.

| | MNIST 3-8 | Malimg | Fashion-MNIST shirts | CIFAR-10 cats-dogs |
|---|---|---|---|---|
| Convex$^*$ | 0.00159 | 0.00295 | 0.00180 | 0.00180 |
| RS Gaussian | 2.16 | 111.9 | 2.41 | 5.78 |
| RS Laplacian | 2.23 | 114.8 | 2.51 | 5.81 |
| RS Uniform | 2.18 | 112.4 | 2.44 | 5.80 |
| Splitting | 0.597 | 994.5 | 0.185 | 0.774 |
| $\alpha, \beta$-CROWN$^\dagger$ | 6.088 | 6.138 | 6.425 | 9.133 |
| Cayley | 0.000505 | — | 0.0451 | 0.0441 |
| $\ell_\infty$-Net$^\ddagger$ | 0.138 | — | 0.115 | — |

While our method produces competitive robustness certificates for $\ell_2$- and $\ell_\infty$-norms (Appendix 5.D), it offers the largest improvement for $\ell_1$-certificates in the high-dimensional image spaces considered. This is likely due to the characteristics of the subgradient dual norm factor in the denominator of Theorem 12. The dual of the $\ell_1$-norm is the $\ell_\infty$-norm, which selects the largest magnitude element in the gradient of the output logit with respect to the input pixels. As the input image scales, it is natural for the classifier to become less dependent on any one specific pixel, shrinking the denominator in Theorem 12. Conversely, when certifying for the $\ell_\infty$-norm, one must evaluate the $\ell_1$-norm of the gradient, which scales proportionally to the input size. Nevertheless, we find in Appendix 5.D that our $\ell_2$- and $\ell_\infty$-radii are generally comparable those of the baselines while maintaining speed and determinism.

Our feature-convex neural network certificates are almost immediate, requiring just one forward pass and one backward pass through the network. This certification procedure requires a few milliseconds per sample on our hardware and scales well with network size. This is substantially faster than the runtime for randomized smoothing, which scales from several seconds per CIFAR-10 image to minutes for an ImageNet image [35]. The only method that rivaled our $\ell_1$-norm certificates was $\alpha, \beta$-CROWN; however, such bound propagation frameworks suffer from exponential computational complexity in network size, and even for small CIFAR-10 ConvNets typically take on the order of minutes to certify nontrivial radii. For computational tractability, we therefore used a smaller network in our simulations (Appendix 5.D). Certification time for all methods is reported in Table 5.1.

Unlike the randomized smoothing baselines, our method is completely deterministic in both prediction and certification. Randomized prediction poses a particular problem for randomized smoothing certificates: even for a perturbation of a "certified" magnitude, repeated evaluations at the perturbed point will eventually yield misclassification for any nontrivial classifier. While the splitting-based certificates of Levine and Feizi [87] are deterministic,

they only certify quantized (not continuous) $\ell_1$-perturbations, which scale poorly to $\ell_2$- and $\ell_\infty$-certificates (Appendix 5.D). Furthermore, the certification runtime grows linearly in the smoothing noise $\sigma$; evaluating the certified radii at $\sigma$ used for the Malimg experiment takes several minutes per sample.

Ablation tests examining the impact of Jacobian regularization, the feature map $\varphi$, and data augmentation are included in Appendix 5.D. We also illustrate the certification performance of our method across all combinations of MNIST classes in Appendix 5.D.

## 5.5 Conclusions

This chapter introduces the problem of asymmetric certified robustness, which we show naturally applies to a number of practical adversarial settings. We define feature-convex classifiers in this context and theoretically characterize their representation power from geometric, approximation theoretic, and statistical lenses. Closed-form sensitive-class certified robust radii for the feature-convex architecture are provided for arbitrary $\ell_p$-norms. We find that our $\ell_1$-robustness certificates in particular match or outperform those of the current state-of-the-art methods, with our $\ell_2$- and $\ell_\infty$-radii also competitive to methods tailored for a particular norm. Unlike smoothing and bound propagation baselines, we accomplish this with a completely deterministic and near-immediate computation scheme. We also show theoretically that significant performance improvements should be realizable for natural image datasets such as CIFAR-10 cats-versus-dogs. Possible directions for future research include bridging the gap between the theoretical power of feature-convex models and their practical implementation, as well as exploring more sophisticated choices of the feature map $\varphi$.

# Appendices

## 5.A Classification Framework Generalization

While outside the scope of this work, we note that there are two natural ways to extend our approach to a multiclass setting with one sensitive class. Let $\mathcal{Y} = \{1, 2, \ldots, c\}$, with class 1 being the sensitive class for which we aim to generate certificates.

One approach involves a two-step architecture, where a feature-convex classifier first distinguishes between the sensitive class 1 and all other classes $\{2, 3, \ldots, c\}$ and an arbitrary second classifier distinguishes between the classes $\{2, 3, \ldots, c\}$. The first classifier could then be used to generate class 1 certificates, as described in Section 5.3.

Alternatively, we could define $g$ to map directly to $c$ output logits, with the first logit convex in the input and the other logits concave in the input. Concavity can be easily achieved by negating the output of a convex network. Let the $i$th output logit then be denoted as $g_i$ and consider an input $x$ where the classifier predicts class 1 (i.e., $g_1(\varphi(x)) \geq g_i(\varphi(x))$ for all $i \in \{2, 3, \ldots, c\}$); since the difference of a convex and a concave function is convex, we can generate a certificate for the nonnegativity of each convex decision function $g_1 \circ \varphi - g_i \circ \varphi$ around $x$. Minimizing these certificates over all $i \in \{2, 3, \ldots, c\}$ yields a robustness certificate for the sensitive class.

Note that $g$ mapping to 2 or more logits, all convex in the input, would not yield any tractable certificates. This is because the classifier decision function would now be the difference of two convex functions and have neither convex nor concave structure. We therefore choose to instantiate our binary classification networks with a single convex output logit for clarity.

### Malimg Multiclass Extension

As a proof-of-concept, we provide a concrete realization of the first scheme above on the Malimg dataset. Namely, consider the setting where we want to distinguish between "clean" binaries and 24 classes of malware. A malware designer seeks to maliciously perturb the bytes in their binary to fool a classifier into falsely predicting that the malware is "clean." We therefore consider a cascading architecture where first a feature-convex classifier answers the "clean or malware" question, and then a subsequent classifier (not necessarily feature-convex) predicts the particular class of malware in the case that the feature-convex classifier

assigns a "malware" prediction. Note that, in the initial step, we can either certify the "clean" binaries or the collection of all 24 malware classes, simply by negating the feature-convex classifier output logit. We logically choose to certify the malware classes as done in our simulations of Section 5.4; these certificates provide guarantees against a piece of malware going undetected.

We use the same feature-convex architecture and training details as described in Appendix 5.D. For the cascaded malware classifier, we use a ResNet-18 architecture trained with Adam for 150 epochs with a learning rate of $10^{-3}$. The confusion plot for the multiclass classifier is provided in Figure 5.3, with an overall accuracy of 96.5%. With the exception of few challenging classes to distinguish, the classifier achieves reasonable performance despite the unbalanced class sizes.

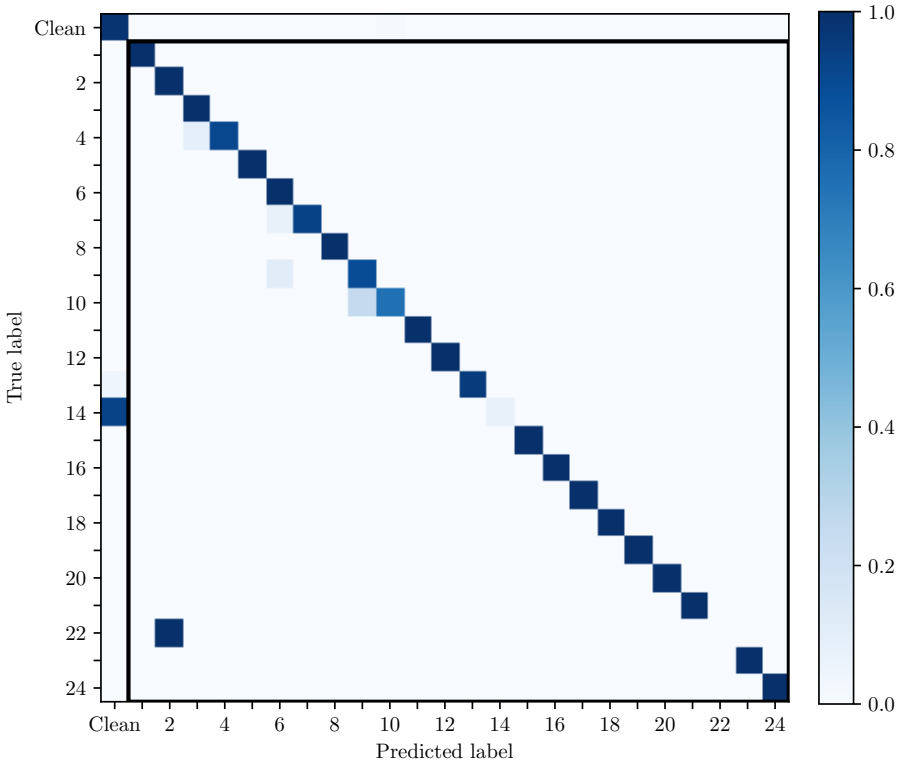



Figure 5.3: The row-normalized confusion plot for the Malimg multiclass classifier. The overall accuracy of the composite classifier is 96.5%. The various malware classes (1-24) are circumscribed with a black rectangle. These are certified against the class of "clean" binaries. See Section 5.4 for more details on the mock clean binaries.

Figure 5.4 visualizes the distribution of certified radii for the four most common malware classes in the dataset, excluding the "Yuner.A" class which featured duplicated images. Note that certification performance varies between classes, with high correlation across different

norms for a particular malware class. Classes which tend to have larger certificates can be interpreted as clustering further away from the clean binaries, requiring larger perturbations to fool the classifier.

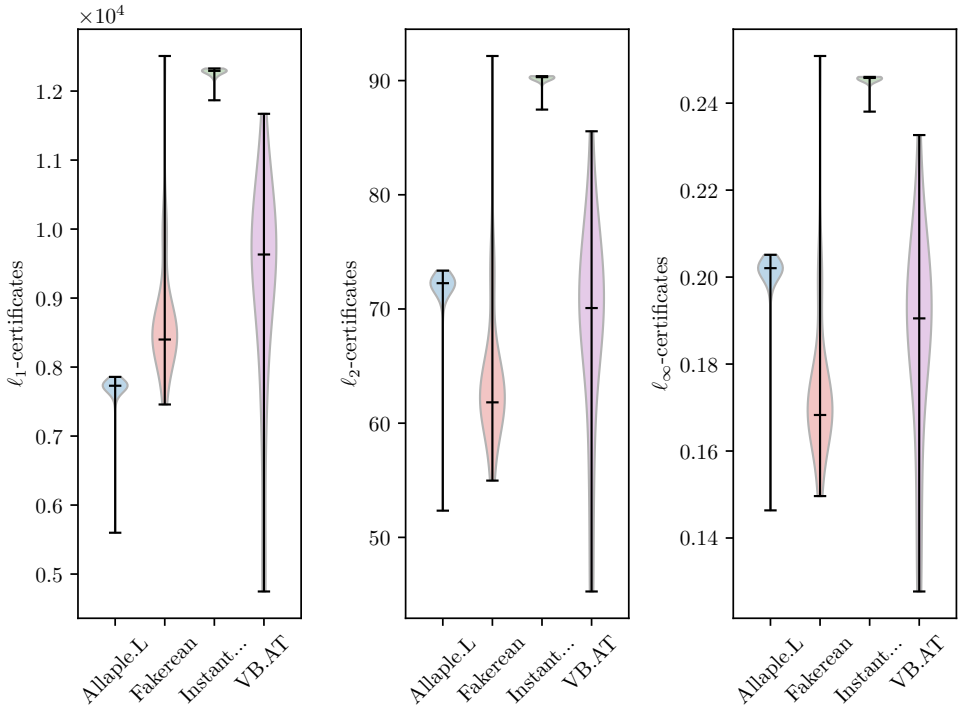
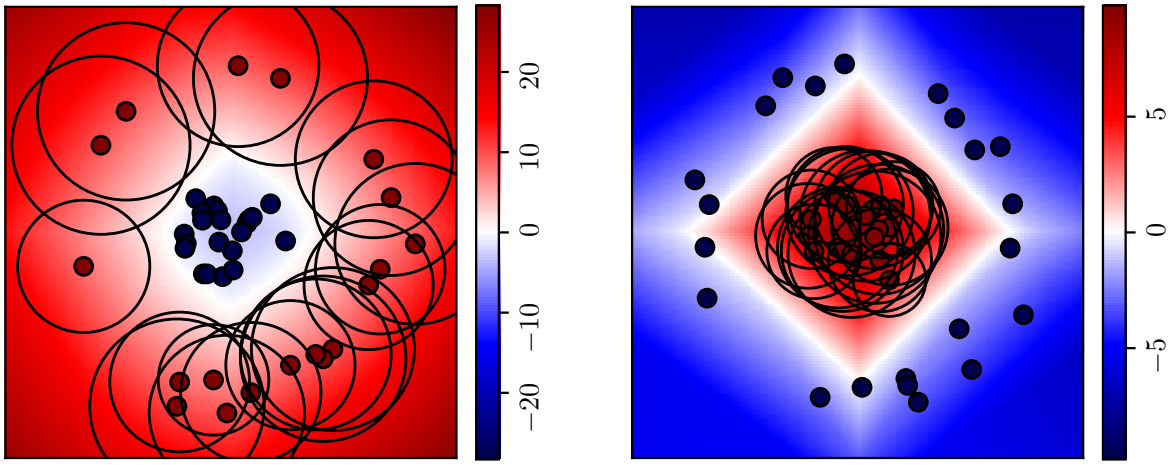



Figure 5.4: Certified radii distributions for four malware classes in the Malimg dataset.

## 5.B Feature Map Motivation

This section examines the importance of the feature map $\varphi$ with a low-dimensional example. Consider the binary classification setting where one class $X_2 \subseteq \mathbb{R}^d$ is clustered around the origin and the other class $X_1 \subseteq \mathbb{R}^d$ surrounds it in a ring. Here, the pair $(X_1, X_2)$ is convexly separable (see Definition 12) as an $\ell_2$-norm ball decision region covering $X_2$ is convex (Figure 5.5a). Note that the reverse pair $(X_2, X_1)$ is *not* convexly separable, as there does not exist a convex set containing $X_1$ but excluding $X_2$. A standard input-convex classifier with $\varphi = \mathrm{Id}$ would therefore be unable to discriminate between the classes in this direction (Proposition 18), i.e., we would be able to learn a classifier that generates certificates for points in $X_1$, but not $X_2$.

The above problem is addressed by choosing the feature map to be the simple concatenation $\varphi(x) = (x, |x|)$ mapping from $\mathbb{R}^d$ to $\mathbb{R}^q = \mathbb{R}^{2d}$, with associated Lipschitz constants $\mathrm{Lip}_1(\varphi) \leq 2$, $\mathrm{Lip}_2(\varphi) \leq \sqrt{2}$, and $\mathrm{Lip}_\infty(\varphi) \leq 1$. In this augmented feature space, $X_1$ and $X_2$




(a) (b)

Figure 5.5: Experiments demonstrating the role of the feature map $\varphi = (x, |x|)$ in $\mathbb{R}^2$, with the output logit shaded. Certified radii from our method are shown as black rings. (a) Certifying the outer class (dark red points). This is possible using an input-convex classifier as a convex sublevel set contains the inner class (dark blue points). (b) Certifying the inner class (dark red points). This would not be possible with $\varphi = \mathrm{Id}$ as there is no convex set containing the outer class (dark blue points) but excluding the inner. The feature map $\varphi$ enables this by permitting convex separability in the higher dimensional space. Note that although the shaded output logit is not convex in the input, we still generate certificates.

are convexly separable in both directions, as they are each contained in a convex set (specifically, a half-space) whose complement contains the other class. We are now able to learn a classifier that takes $X_2$ as the sensitive class for which certificates are required (Figure 5.5b). This parallels the motivation of the support vector machine "kernel trick," where inputs are augmented to a higher-dimensional space wherein the data is linearly separable (instead of convexly separable as in our case).

## 5.C Supporting Lemmas

We now introduce a preliminary lemma for the results in Section 5.3.

**Lemma 9.** *For any nonempty closed convex set $X \subseteq \mathbb{R}^d$, there exists a convex function $g\colon \mathbb{R}^d \to \mathbb{R}$ such that $X = g^{-1}((-\infty, 0]) = \{x \in \mathbb{R}^d : g(x) \leq 0\}$.*

*Proof.* Let $X \subseteq \mathbb{R}^d$ be a nonempty closed convex set. We take the distance function $g = d_X$ defined by $d_X(x) = \inf_{y \in X} \|y - x\|_2$. Since $X$ is closed and $y \mapsto \|y - x\|_2$ is coercive for all $x \in \mathbb{R}^d$, it holds that $y \mapsto \|y - x\|_2$ attains its infimum over $X$ [22, Proposition

A.8]. Let $x^{(1)}, x^{(2)} \in \mathbb{R}^d$ and let $\theta \in [0,1]$. Then there exist $y^{(1)}, y^{(2)} \in X$ such that $g(x^{(1)}) = \|y^{(1)} - x^{(1)}\|_2$ and $g(x^{(2)}) = \|y^{(2)} - x^{(2)}\|_2$. Since $X$ is convex, it holds that $\theta y^{(1)} + (1-\theta)y^{(2)} \in X$, and therefore

$$\begin{aligned} g(\theta x^{(1)} + (1-\theta)x^{(2)}) &= \inf_{y \in X} \|y - (\theta x^{(1)} + (1-\theta)x^{(2)})\|_2 \\ &\leq \|\theta y^{(1)} + (1-\theta)y^{(2)} - (\theta x^{(1)} + (1-\theta)x^{(2)})\|_2 \\ &\leq \theta \|y^{(1)} - x^{(1)}\|_2 + (1-\theta)\|y^{(2)} - x^{(2)}\|_2 \\ &= \theta g(x^{(1)}) + (1-\theta) g(x^{(2)}). \end{aligned}$$

Hence, $g = d_X$ is convex. Since $X = \{x \in \mathbb{R}^d : \inf_{y \in X} \|y - x\|_2 = 0\} = \{x \in \mathbb{R}^d : d_X(x) = 0\} = \{x \in \mathbb{R}^d : d_X(x) \leq 0\} = \{x \in \mathbb{R}^d : g(x) \leq 0\}$ by nonnegativity of $d_X$, the lemma holds. □

In order to apply the universal approximation results in Chen, Shi, and Zhang [34], we introduce their parameterization of input-convex ReLU neural networks. Note that it imposes the additional constraint that the first weight matrix $A^{(1)}$ is elementwise nonnegative.

**Definition 13.** Define $\tilde{\mathcal{F}}_{\text{Id}}$ to be the class of functions $\tilde{f} \colon \mathbb{R}^d \to \{1, 2\}$ given by $\tilde{f}(x) = T(\tilde{g}(x))$ with $\tilde{g} \colon \mathbb{R}^d \to \mathbb{R}$ given by

$$\begin{aligned} x^{(1)} &= \text{ReLU}\left(A^{(1)}x + b^{(1)}\right), \\ x^{(l)} &= \text{ReLU}\left(A^{(l)}x^{(l-1)} + b^{(l)} + C^{(l)}x\right), \ l \in \{2, 3, \ldots, L-1\}, \\ \tilde{g}(x) &= A^{(L)}x^{(L-1)} + b^{(L)} + C^{(L)}x, \end{aligned}$$

for some $L \in \mathbb{N}$, $L > 1$, and some consistently sized matrices $A^{(1)}, C^{(1)}, \ldots, A^{(L)}, C^{(L)}$, all of which have nonnegative elements, and some consistently sized vectors $b^{(1)}, \ldots, b^{(L)}$.

The following preliminary lemma relates the class $\hat{\mathcal{F}}_{\text{Id}}$ from Definition 11 to the class $\tilde{\mathcal{F}}_{\text{Id}}$ above.

**Lemma 10.** *It holds that* $\tilde{\mathcal{F}}_{\text{Id}} \subseteq \hat{\mathcal{F}}_{\text{Id}}$.

*Proof.* Let $\tilde{f} \in \tilde{\mathcal{F}}_{\text{Id}}$. Then certainly $A^{(l)} \geq 0$ for all $l \in \{2, 3, \ldots, L\}$, so indeed $\tilde{f} \in \hat{\mathcal{F}}_{\text{Id}}$. Hence, $\tilde{\mathcal{F}}_{\text{Id}} \subseteq \hat{\mathcal{F}}_{\text{Id}}$. □

Theorem 1 in Chen, Shi, and Zhang [34] shows that a Lipschitz convex function can be approximated within an arbitrary tolerance. We now provide a technical lemma adapting Theorem 1 in Chen, Shi, and Zhang [34] to show that convex functions can be *underapproximated* within an arbitrary tolerance on a compact convex subset.

**Lemma 11.** *For any convex function* $g \colon \mathbb{R}^d \to \mathbb{R}$*, any compact convex subset* $X$ *of* $\mathbb{R}^d$*, and any* $\epsilon > 0$*, there exists* $\hat{f} \in \hat{\mathcal{F}}_{\text{Id}}$ *such that* $\hat{g}(x) < g(x)$ *for all* $x \in X$ *and*

$$\sup_{x \in X} (g(x) - \hat{g}(x)) < \epsilon.$$

*Proof.* Let $g\colon \mathbb{R}^d \to \mathbb{R}$ be a convex function, let $X$ be a compact convex subset of $\mathbb{R}^d$, and let $\epsilon > 0$. Since $g - \epsilon/2$ is a real-valued convex function on $\mathbb{R}^d$ (and hence is proper), its restriction to the closed and bounded set $X$ is Lipschitz continuous [119, Theorem 10.4], and therefore Lemma 10 together with Theorem 1 in Chen, Shi, and Zhang [34] gives that there exists $\hat{f} \in \tilde{\mathcal{F}}_{\mathrm{Id}} \subseteq \hat{\mathcal{F}}_{\mathrm{Id}}$ such that $\sup_{x\in X} |(g(x) - \epsilon/2) - \hat{g}(x)| < \epsilon/2$. Thus, for all $x \in X$,

$$\begin{aligned} g(x) - \hat{g}(x) &= \left(g(x) - \frac{\epsilon}{2}\right) - \hat{g}(x) + \frac{\epsilon}{2} \\ &> \left(g(x) - \frac{\epsilon}{2}\right) - \hat{g}(x) + \sup_{y\in X} \left|\left(g(y) - \frac{\epsilon}{2}\right) - \hat{g}(y)\right| \\ &\geq \left(g(x) - \frac{\epsilon}{2}\right) - \hat{g}(x) + \left|\left(g(x) - \frac{\epsilon}{2}\right) - \hat{g}(x)\right| \\ &\geq 0. \end{aligned}$$

Furthermore,

$$\begin{aligned} \sup_{x\in X} (g(x) - \hat{g}(x)) &= \sup_{x\in X} |g(x) - \hat{g}(x)| \\ &= \sup_{x\in X} \left|\left(g(x) - \frac{\epsilon}{2}\right) - \hat{g}(x) + \frac{\epsilon}{2}\right| \\ &\leq \sup_{x\in X} \left|\left(g(x) - \frac{\epsilon}{2}\right) - \hat{g}(x)\right| + \frac{\epsilon}{2} \\ &< \epsilon, \end{aligned}$$

which proves the lemma. □

We leverage Lemma 11 to construct a uniformly converging sequence of underapproximating functions.

**Lemma 12.** *For all $f \in \mathcal{F}_{\mathrm{Id}}$ and all compact convex subsets $X$ of $\mathbb{R}^d$, there exists a sequence $\{\hat{f}_n \in \hat{\mathcal{F}}_{\mathrm{Id}} : n \in \mathbb{N}\} \subseteq \hat{\mathcal{F}}_{\mathrm{Id}}$ such that $\hat{g}_n(x) < \hat{g}_{n+1}(x) < g(x)$ for all $x \in X$ and all $n \in \mathbb{N}$ and $\hat{g}_n$ converges uniformly to $g$ on $X$ as $n \to \infty$.*

*Proof.* Let $f \in \mathcal{F}_{\mathrm{Id}}$ and let $X$ be a compact convex subset of $\mathbb{R}^d$. Let $\{\epsilon_n > 0 : n \in \mathbb{N}\}$ be a sequence such that $\epsilon_{n+1} < \epsilon_n$ for all $n \in \mathbb{N}$ and $\epsilon_n \to 0$ as $n \to \infty$. Such a sequence clearly exists, e.g., by taking $\epsilon_n = 1/n$ for all $n \in \mathbb{N}$. Now, for all $n \in \mathbb{N}$, the function $g - \epsilon_{n+1}$ is convex, and therefore by Lemma 11 there exists $\hat{f}_n \in \hat{\mathcal{F}}_{\mathrm{Id}}$ such that $\hat{g}_n(x) < g(x) - \epsilon_{n+1}$ for all $x \in X$ and $\sup_{x\in X} ((g(x) - \epsilon_{n+1}) - \hat{g}_n(x)) < \epsilon_n - \epsilon_{n+1}$. Fixing such $\hat{f}_n, \hat{g}_n$ for all $n \in \mathbb{N}$, we see that $\sup_{x\in X} ((g(x) - \epsilon_{n+2}) - \hat{g}_{n+1}(x)) < \epsilon_{n+1} - \epsilon_{n+2}$, which implies that

$$\hat{g}_{n+1}(x) > g(x) - \epsilon_{n+1} > \hat{g}_n(x)$$

for all $x \in X$, which proves the first inequality. The second inequality comes from the fact that $\hat{g}_{n+1}(x) < g(x) - \epsilon_{n+2} < g(x)$ for all $x \in X$. Finally, since $g(x) - \hat{g}_n(x) > \epsilon_{n+1} > 0$ for all $x \in X$ and all $n \in \mathbb{N}$, we see that

$$\sup_{x\in X} |g(x) - \hat{g}_n(x)| = \sup_{x\in X} (g(x) - \hat{g}_n(x)) < \epsilon_n \to 0 \text{ as } n \to \infty,$$

which proves that $\lim_{n\to\infty} \sup_{x\in X} |g(x) - \hat{g}_n(x)| = 0$, so indeed $\hat{g}_n$ converges uniformly to $g$ on $X$ as $n \to \infty$. □

# 5.D Additional Experimental Details

## CIFAR-10 Cats-versus-Dogs Convex Separability

In order to establish that the cat and dog images in CIFAR-10 are convexly separable, we experimentally attempt to reconstruct an image from one class using a convex combination of all images in the other class (without augmentation such as random crops, flips, etc.). Namely, if $x$ is drawn from one class and $y^{(1)}, \ldots, y^{(N)}$ represent the entirety of the other class, we form the following optimization problem:

$$\begin{aligned} \underset{\alpha\in\mathbb{R}^N}{\text{minimize}} \quad & \left\| x - \sum_{j=1}^{N} \alpha_j y^{(j)} \right\|_2 \\ \text{subject to} \quad & \alpha \geq 0, \\ & \sum_{j=1}^{N} \alpha_j = 1. \end{aligned}$$

The reverse experiment for the other class follows similarly. We solve the optimization using MOSEK [10], and report the various norms of $x - \sum_{j=1}^{N} \alpha_j y^{(j)}$ in Figure 5.6. Reconstruction accuracy is generally very poor, with no reconstruction achieving better than an $\ell_1$-error of 52. A typical reconstructed image is shown in Figure 5.7.

Yousefzadeh [161] and Balestriero, Pesenti, and LeCun [19] showed a related empirical result for CIFAR-10, namely, that no test set image can be reconstructed as a convex combination of training set images. However, we remark that their findings do not necessarily imply that a training set image cannot be reconstructed via other training set images; our new finding that the CIFAR-10 cats-versus-dogs training set is convexly separable is required in order to assert Fact 1.

## Experimental Setup

We include a detailed exposition of our experimental setup in this section, beginning with general details on our choice of epochs and batch size. We then discuss baseline methods, architecture choices for our method, class balancing, and data processing.

**Epochs and batch size.** Exempting the randomized smoothing baselines, for the MNIST 3-8 and Fashion-MNIST shirts simulations, we use 60 epochs for all methods. This is increased to 150 epochs for the Malimg dataset and CIFAR-10 cats-dogs simulations. The

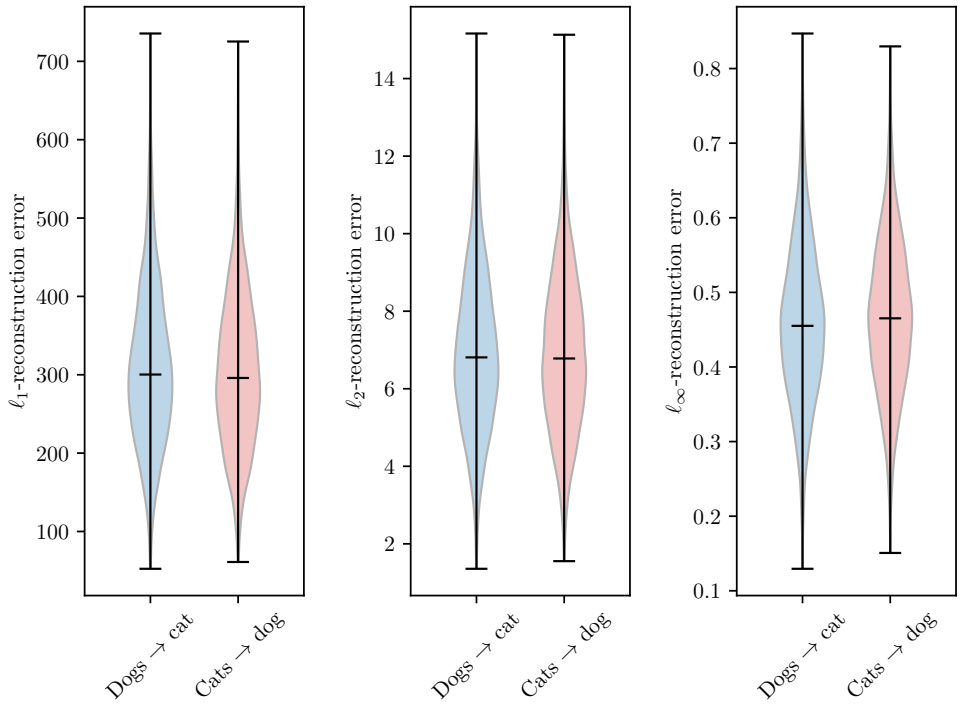



Figure 5.6: Reconstructing CIFAR-10 cat and dog images as convex combinations. The label "Dogs → cat" indicates that a cat image was attempted to be reconstructed as a convex combination of all 5000 dog images.

batch size is 64 for all datasets besides the $512 \times 512$ Malimg dataset, where it is lowered to 32.

To ensure a fair comparison, the randomized smoothing baseline epochs are scaled larger than the aforementioned methods according to the noise value specified in the sweeps below. The final epochs and smoothing noise values used are reported in Table 5.2. Note that as classifiers are typically more robust to the noise from splitting smoothing, larger values of $\sigma$ are used for only this smoothing method in the MNIST 3-8 and Malimg datasets. For Malimg, we find experimentally that even noise values of up to $\sigma = 100$ are tractable for the splitting method, outside the sweep range considered below. As verification at that $\sigma$ already takes several minutes per sample and runtime scales linearly with $\sigma$, we do not explore larger values of $\sigma$.

**Hardware.** All simulations were conducted on a single Ubuntu 20.04 instance with an Nvidia RTX A6000 GPU. Complete reproduction of the experiments takes approximately 0.08 GPU-years.

Figure 5.7: Reconstructing a CIFAR-10 cat image (left) from a convex combination of dog images (right). The reconstruction error norms are 294.57, 6.65, and 0.38 for the $\ell_1$-, $\ell_2$-, and $\ell_\infty$-norms, respectively. These are typical, as indicated by Figure 5.6.

Table 5.2: Randomized smoothing final noise and epoch hyperparameters.

| Dataset | Laplacian, Uniform, and Gaussian | Splitting |
|---|---|---|
| MNIST 3-8 | $(\sigma, n) = (0.75, 60)$ | $(\sigma, n) = (0.75 \cdot 4, 60 \cdot 4)$ |
| Malimg | $(\sigma, n) = (3.5 \cdot 4, 150 \cdot 4)$ | $(\sigma, n) = (100, 150 \cdot 4)$ |
| Fashion-MNIST shirts | $(\sigma, n) = (0.75, 60)$ | $(\sigma, n) = (0.75, 60)$ |
| CIFAR-10 cats-dogs | $(\sigma, n) = (0.75 \cdot 2, 600 \cdot 2)$ | $(\sigma, n) = (0.75 \cdot 2, 600 \cdot 2)$ |

### Datasets

We introduce the various datasets considered in this chapter. MNIST 3-8 and Malimg are relatively simple classification problems where near-perfect classification accuracy is attainable; the Malimg dataset falls in this category despite containing relatively large images. Our more challenging settings consist of a Fashion-MNIST shirts dataset as well as CIFAR-10 cats-versus-dogs dataset.

For consistency with Zhang et al. [165], we augment the MNIST and Fashion-MNIST training data with 1-pixel padding and random cropping. The CIFAR-10 dataset is augmented with 3-pixel edge padding, horizontal flips, and random cropping. The Malimg dataset is augmented with 20-pixel padding and random $512 \times 512$ cropping.

For CIFAR-10, MNIST, and Fashion-MNIST, we use the preselected test sets. For Malimg we hold out a random 20% test dataset, although this may not be entirely used during

testing. The training set is further subdivided by an 80%-20% validation split. For all simulations, we normalize pixel values into the interval $[0, 1]$, and we use the first 1000 test samples to evaluate our methods.

**MNIST 3-8.** For our MNIST binary classification problem, we choose the problem of distinguishing between 3 and 8 [84]. These were selected as 3 and 8 are generally more visually similar and challenging to distinguish than other digit pairs. Images are $28 \times 28$ pixels and greyscale.

**Malimg.** Our malware classification experiments use greyscale, bytewise encodings of raw malware binaries Nataraj et al. [106]. Each image pixel corresponds to one byte of data, in the range of 0–255, and successive bytes are added horizontally from left to right on the image until wrapping at some predetermined width. We use the extracted malware images from the seminal dataset Nataraj et al. [106], normalizing pixel values into $[0, 1]$, as well as padding and cropping images to be $512 \times 512$. Note that licensing concerns generally prevent the distribution of "clean" executable binaries. As this chapter is focused on providing a general approach to robust classification, in the spirit of reproducibility we instead report classification results between different kinds of malware. Namely, we distinguish between malware from the most numerous "Allaple.A" class (2949 samples) and an identically-sized random subset of all other 24 malware classes. To simulate a scenario where we must provide robustness against evasive malware, we provide certificates for the latter collection of classes.

**Fashion-MNIST shirts.** The hardest classes to distinguish in the Fashion-MNIST dataset are T-shirts vs shirts, which we take as our two classes [74, 156]. Images are $28 \times 28$ pixels and greyscale.

**CIFAR-10 cats-dogs.** We take as our two CIFAR-10 classes the cat and dog classes since they are relatively difficult to distinguish [61, 90, 114]. Other classes (e.g., ships) are typically easier to classify since large background features (e.g., blue water) are strongly correlated with the target label. Samples are $32 \times 32$ RGB images.

### Baseline Methods

We provide additional details on each of the baseline methods below.

**Randomized smoothing.** Since the certification runtime of randomized smoothing is large, especially for the $512 \times 512$ pixel Malimg images, we evaluate the randomized smoothing classifiers over $10^4$ samples and project the certified radius to $10^5$ samples by scaling the number fed into the Clopper-Pearson confidence interval, as described in Cohen, Rosenfeld, and Kolter [35]. This allows for a representative and improved certified accuracy curve while dramatically reducing the method's runtime. We take an initial guess for the certification

class with $n_0 = 100$ samples and set the incorrect prediction tolerance parameter $\alpha = 0.001$. For CIFAR-10 we use a depth-40 Wide ResNet base classifier, mirroring the choices from Cohen, Rosenfeld, and Kolter [35] and Yang et al. [158]; for all other datasets we use a ResNet-18. All networks are trained using SGD with an initial learning rate of 0.1, Nesterov momentum of 0.9, weight decay of $10^{-4}$, and cosine annealing scheduling as described in Yang et al. [158]. Final smoothing noise values are selected as in Table 5.2, and are determined from the noise level comparison sweeps below.

**Splitting noise.** As this method is a deterministic derivative of randomized smoothing, it avoids the many aforementioned hyperparameter choices. We use the same architectures described above for the other randomized smoothing simulations.

**Cayley convolutions.** To maintain consistency, we use a two-hidden-layer multilayer perceptron with $(n_1, n_2) = (200, 50)$ hidden features, CayleyLinear layers, and GroupSort activations for the MNIST simulation. For the more challenging Fashion-MNIST and CIFAR-10 simulations, we use the ResNet-9 architecture implementation from Trockman and Kolter [136]. Following the authors' suggestions, we train these networks using Adam with a learning rate of 0.001.

**$\ell_\infty$-distance nets.** As the architecture of the $\ell_\infty$-distance net [165] is substantially different from traditional architectures, we use the authors' 5-layer MNIST/Fashion-MNIST architecture and 6-layer CIFAR-10 architecture with 5120 neurons per hidden layer. Unfortunately, the classification accuracy on the CIFAR-10 cats-dogs simulation remained near 50% throughout training. This was not the case when we tested easier classes, such as planes-versus-cars, where large features (e.g., blue sky) can be used to discriminate. We therefore only include this model in the MNIST and Fashion-MNIST simulations, and use the training procedure directly from the aforementioned paper's codebase.

**$\alpha, \beta$-CROWN.** As $\alpha, \beta$-CROWN certification time scales exponentially with the network size, we keep the certified networks small in order to improve the certification performance of the baseline. For all datasets, we train and certify a one-hidden-layer network with 200 hidden units and ReLU activations. All networks are adversarially trained for a $\ell_\infty$-perturbation radius starting at 0.001 and linearly scaling to the desired $\epsilon$ over the first 20 epochs, as described in Kayed, Anter, and Mohamed [74], which trained the models used in Wang et al. [143]. The desired final $\epsilon$ is set to 0.3 for MNIST, 0.1 for Fashion-MNIST and Malimg, and 2/255 for CIFAR-10. The adversarial training uses a standard PGD attack with 50 steps and step size $2\epsilon/50$. Other optimizer training details are identical to Wang et al. [143]. The branch-and-bound timeout is set to 30 seconds to maintain comparability to other methods, and robustness is evaluated over a dataset-dependent range of discrete radii for each adversarial norm.

**Feature-Convex Architecture and Training**

In this section, we provide more details of the feature-convex architectures used in our numerical simulations of Section 5.4. For the more challenging datasets (Fashion-MNIST shirts and CIFAR-10 cats-dogs), we use various instantiations of a convex ConvNet (described below) where successive layers have a constant number of channels and image size. This allows for the addition of identity residual connections to each convolution and lets us remove the passthrough connections altogether. Convexity is enforced by projecting relevant weights onto the nonnegative orthant after each epoch and similarly constraining BatchNorm $\gamma$ parameters to be positive. We initialize positive weight matrices to be drawn uniformly from the interval $[0, \epsilon]$, where $\epsilon = 0.003$ for linear weights and $\epsilon = 0.005$ for convolutional weights. Jacobian regularization is also used to improve our certified radii [67].

The convex ConvNet architecture consists of a sequence of convolutional layers, BatchNorms, and ReLU nonlinearities. The first convolutional layer is unconstrained, as the composition of a convex function with an affine function is still convex [3]. All subsequent convolutions and the final linear readout layer are uniformly initialized from some small positive weight interval ($[0, 0.003]$ for linear weights, $[0, 0.005]$ for convolutional weights) and projected to have nonnegative weights after each gradient step. We found this heuristic initialization choice helps to stabilize network training, as standard Kaiming initialization assumptions are violated when weights are constrained to be nonnegative instead of normally distributed with mean zero. More principled weight initialization strategies for this architecture would form an exciting area of future research. Before any further processing, inputs into the network are fed into an initial BatchNorm—this enables flexibility with different feature augmentation maps.

Since the first convolutional layer is permitted negative weights, we generally attain better performance by enlarging the first convolution kernel size (see Table 5.3). For subsequent convolutions, we set the stride to 1, the input and output channel counts to the output channel count from the first convolution, and the padding to half the kernel size, rounded down. This ensures that the output of each of these deeper convolutions has equivalent dimension to its input, allowing for an identity residual connection across each convolution. If $C_i(z)$ is a convolutional operation on a hidden feature $z$, this corresponds to evaluating $C_i(z)+z$ instead of just $C_i(z)$. The final part of the classifier applies MaxPool and BatchNorm layers before a linear readout layer with output dimension 1. See Figure 5.8 for a diagram depicting an exemplar convex ConvNet instantiation.

For training, we use a standard binary cross entropy loss, optionally augmented with a Jacobian regularizer on the Frobenius norm of the network Jacobian scaled by $\lambda > 0$ [67]. As our certified radii in Theorem 12 vary inversely to the norm of the Jacobian, this regularization helps boost our certificates at a minimal loss in clean accuracy. We choose $\lambda = 0.0075$ for CIFAR-10, $\lambda = 0.075$ for Malimg and $\lambda = 0.01$ for MNIST and Fashion-MNIST. Further ablation tests studying the impact of regularization are reported below. All feature-convex networks are trained using SGD with a learning rate of 0.001, momentum 0.9, and exponential learning rate decay with $\gamma = 0.99$.

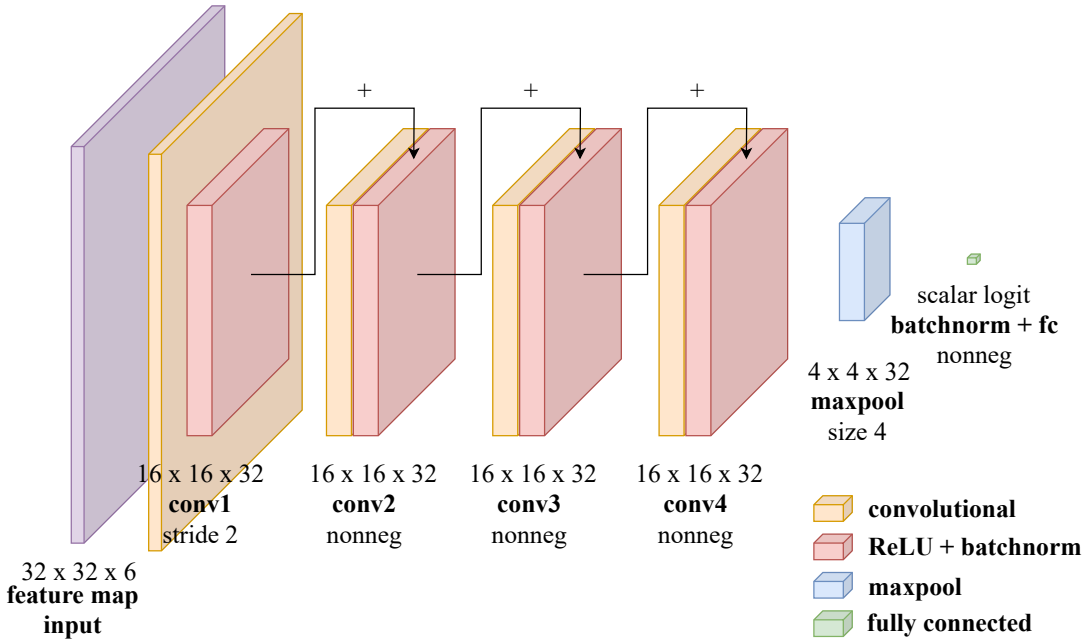



Figure 5.8: An example convex ConvNet of depth 4 with a $C_1$ stride of 2, pool size of 4, and $32 \times 32$ RGB images. There are 6 input channels from the output of the feature map $\varphi\colon x \mapsto (x - \mu, |x - \mu|)$.

Table 5.3: Convex ConvNet architecture parameters. $C_1$ denotes the first convolution, with $C_{2,\dots}$ denoting all subsequent convolutions. The "Features" column denotes the number of output features of $C_1$, which is held fixed across $C_{2,\dots}$. The "Pool" column refers to the size of the final MaxPool window before the linear readout layer. The MNIST and Malimg architectures are simple multilayer perceptrons and are therefore not listed here.

| Dataset | Features | Depth | $C_1$ size | $C_1$ stride | $C_1$ dilation | $C_{2,\dots}$ size | Pool |
|---|---|---|---|---|---|---|---|
| Fashion-MNIST | 4 | 3 | 5 | 1 | 1 | 3 | 1 |
| CIFAR-10 | 16 | 5 | 11 | 1 | 1 | 3 | 1 |

### Class Accuracy Balancing

As discussed in Section 5.1, a balanced class 1 and class 2 test accuracy is essential for a fair comparison of different methods. For methods where the output logits can be directly balanced, this is easily accomplished by computing the ROC curve and choosing the threshold that minimizes $|\text{TPR} - (1 - \text{FPR})|$. This includes both our feature-convex classifiers with one output logit and the Cayley orthogonalization and $\ell_\infty$-Net architectures with two output logits.

Randomized smoothing classifiers are more challenging as the relationship between the base classifier threshold and the smoothed classifier prediction is indirect. We address this using a binary search balancing procedure. Namely, on each iteration, the classifier's prediction routine is executed over the test dataset and the "error" between the class 1 accuracy and the class 2 accuracy is computed. The sign of the error then provides the binary signal for whether the threshold should be shifted higher or lower in the standard binary search implementation. This procedure is continued until the error drops below 1%.

## $\ell_2$- and $\ell_\infty$-Certified Radii

This section reports the counterpart to Figure 5.2 for the $\ell_2$- and $\ell_\infty$-norms. Across all simulations, we attain substantial $\ell_2$- and $\ell_\infty$-radii without relying on computationally expensive sampling schemes or nondeterminism. Methods that certify to another norm $\|\cdot\|_p$ are converted to $\ell_q$-radii at a factor of 1 if $p > q$ or $d^{1/p-1/q}$ otherwise.

Certified $\ell_2$-radii are reported in Figure 5.9. Our $\ell_2$-radii are moderate, generally slightly smaller than those produced by Gaussian randomized smoothing.

Certified $\ell_\infty$-radii are reported in Figure 5.10. For the MNIST 3-8 simulation, the $\ell_\infty$-distance nets produce exceptional certified radii. Likewise, the $\ell_\infty$-distance net certificates are dominant for the Fashion-MNIST dataset, despite achieving slightly inferior clean accuracy. We note however that the applicability of $\ell_\infty$-distance nets for sophisticated vision tasks is uncertain as the method is unable to achieve better-than-random performance for CIFAR-10 cats-dogs. Our method is comparable to randomized-smoothing and $\alpha, \beta$-CROWN in all $\ell_\infty$ simulations.

## Ablation Tests

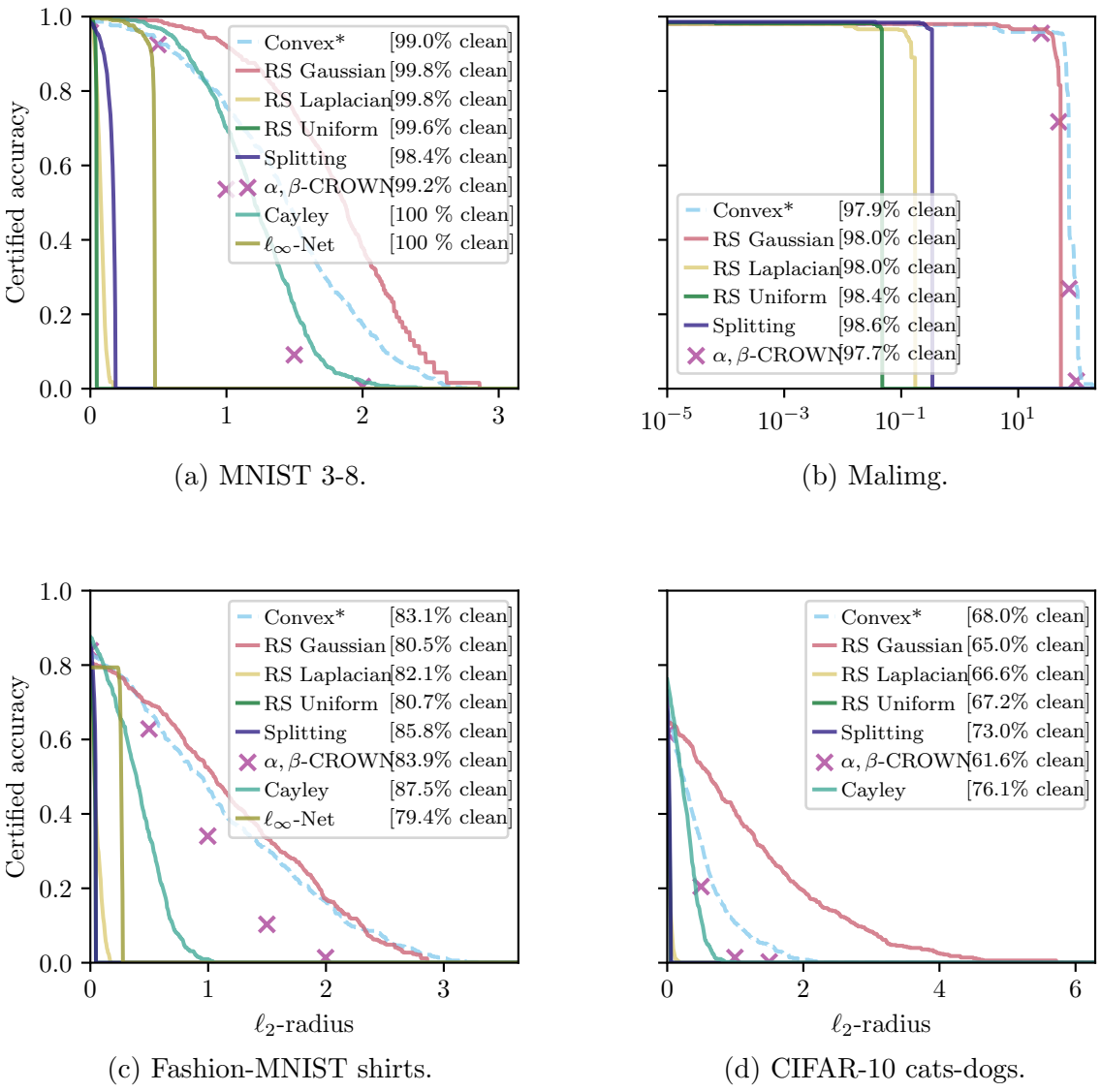
We conduct a series of ablation tests on the CIFAR-10 cats-dogs dataset, examining the impact of regularization, feature maps, and data augmentation.

### Regularization

Figure 5.11 examines the impact of Jacobian regularization over a range of regularization scaling factors $\lambda$, with $\lambda = 0$ corresponding to no regularization. As is typical, we see a tradeoff between clean accuracy and certified radii. Further increases in $\lambda$ yield minimal additional benefit.

### Feature Map

In this section, we investigate the importance of the feature map $\varphi$. Figure 5.12 compares our standard feature-convex classifier with $\varphi(x) = (x - \mu, |x - \mu|)$ against an equivalent architecture with $\varphi = \mathrm{Id}$. Note that the initial layer in the convex ConvNet is a BatchNorm, so even with $\varphi = \mathrm{Id}$, features still get normalized before being passed into the convolutional




(a) MNIST 3-8.

(b) Malimg.

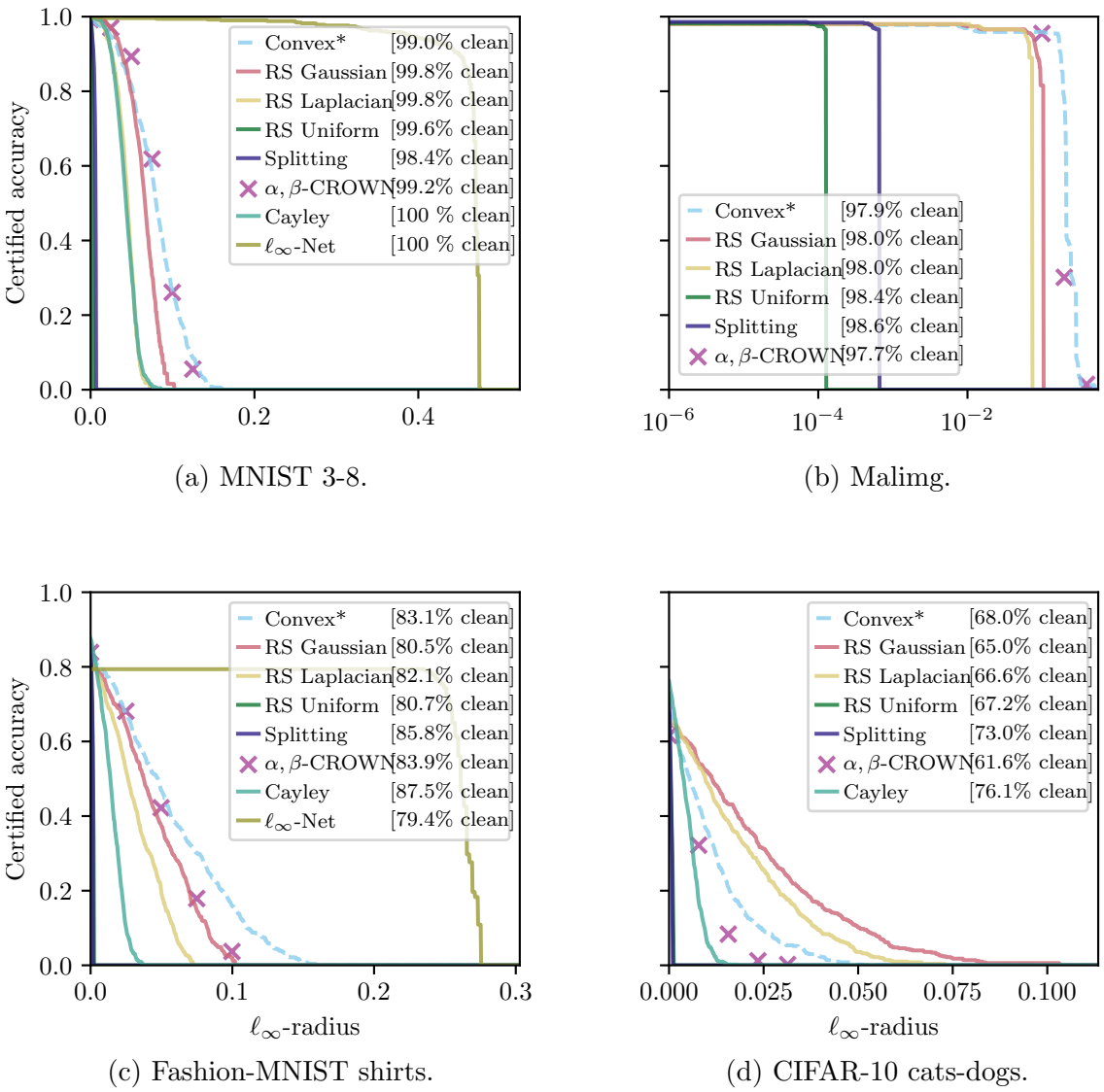
(c) Fashion-MNIST shirts.

(d) CIFAR-10 cats-dogs.

Figure 5.9: Class 1 certified radii curves for the $\ell_2$-norm.

architecture. We perform this experiment across both the standard cats-dogs experiment (cats are certified) in the main text and the reverse dogs-cats experiment (dogs are certified).

As expected, the clean accuracies for both datasets are lower for $\varphi = \mathrm{Id}$, while the certified radii are generally larger due to the Lipschitz scaling factor in Theorem 12. Interestingly, while the standard $\varphi$ produces comparable performance in both experiments, the identity feature map classifier is more effective in the dogs-cats experiment, achieving around 7% greater clean accuracy. This reflects the observation that convex separability is




(a) MNIST 3-8.

(b) Malimg.

(c) Fashion-MNIST shirts.

(d) CIFAR-10 cats-dogs.

Figure 5.10: Class 1 certified radii curves for the $\ell_\infty$-norm.

an asymmetric condition and suggests that feature maps can mitigate this concern.

### Unaugmented Accuracies

Table 5.4 summarizes the experimental counterpart to the representation power characterization in Section 5.3. Namely, Fact 1 proves that there exists an input-convex classifier ($\varphi = \mathrm{Id}$) that achieves perfect training accuracy on the CIFAR-10 cats-dogs dataset with no dataset augmentations (random crops, flips, etc.). Our practical experiments are far from

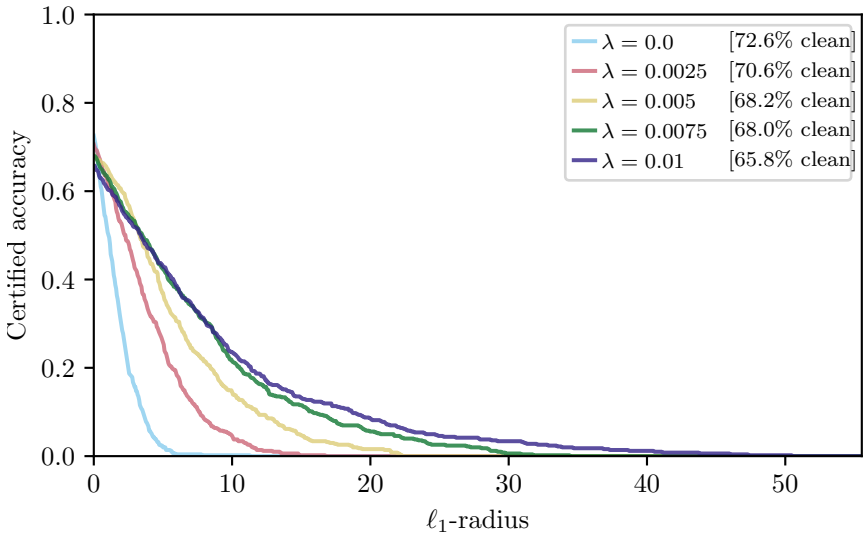



Figure 5.11: Impact of the Jacobian regularization parameter $\lambda$ on CIFAR-10 cats-dogs classification.

achieving this theoretical guarantee, with just 73.4% accuracy for cats-dogs and 77.2% for dogs-cats. Improving the practical performance of input-convex classifiers to match their theoretical capacity is an exciting area of future research.

Table 5.4: CIFAR-10 accuracies with no feature augmentation ($\varphi = \mathrm{Id}$) and no input augmentation.

| Class 1-class 2 data | Training accuracy | Test accuracy (balanced) |
|---|---|---|
| Cats-dogs | 73.4% | 57.3% |
| Dogs-cats | 77.2% | 63.9% |

## MNIST Classes Sweep

For our comparison experiments, we select a specific challenging MNIST class pair (3 versus 8). For completeness, this section includes certification results for our method over all combinations of class pairs in MNIST. As this involves training models over 90 combinations, we lower the number of epochs from 60 to 10, maintaining all other architectural details described in the experimental setup above.

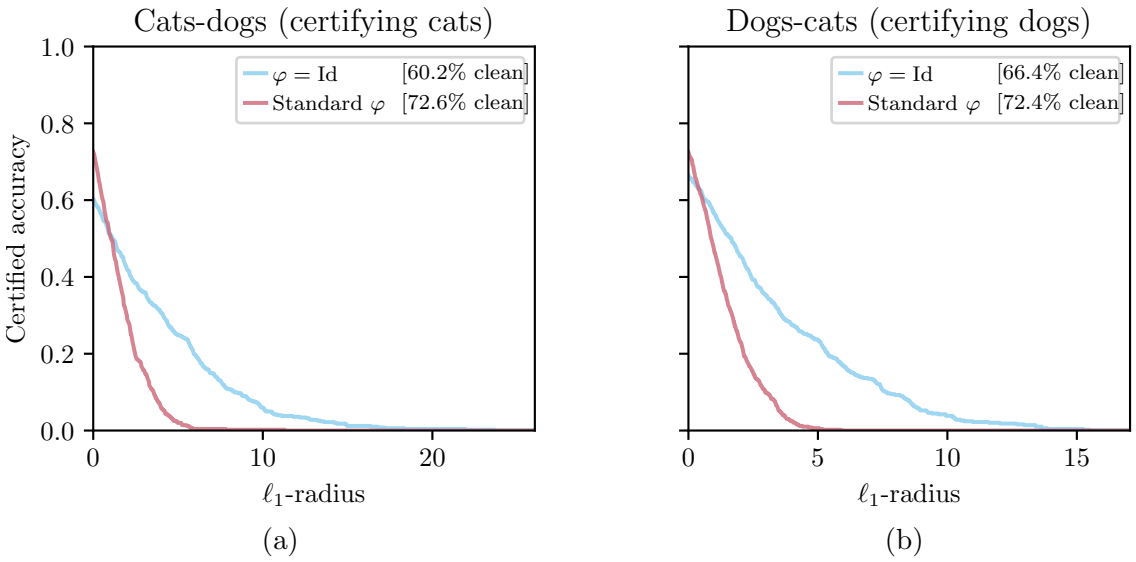


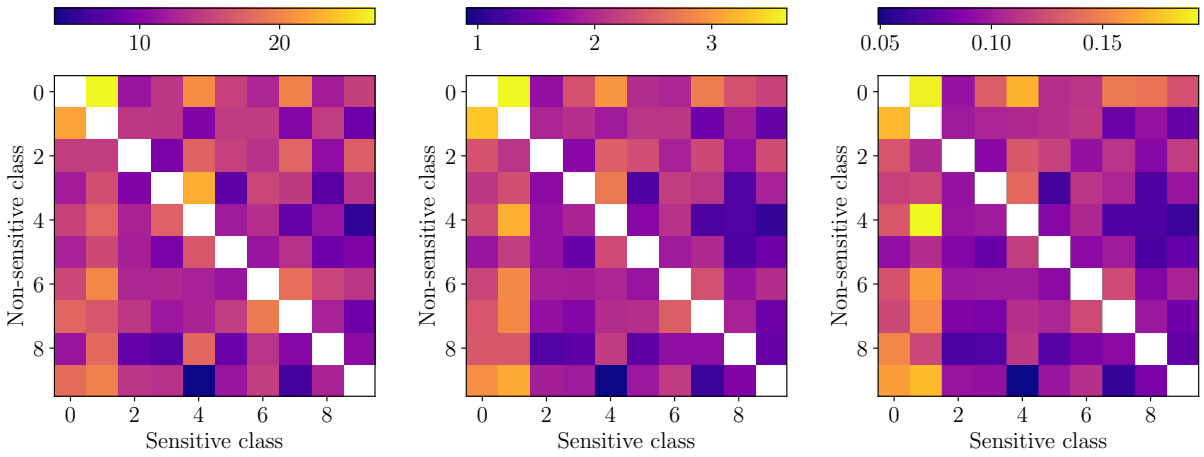

Figure 5.12: (a) Certification performance with cats as class 1 and dogs as class 2. (b) Certification performance with dogs as class 1 and cats as class 2.

Our certified radii naturally scale with the complexity of the classification problem. As expected, 3 and 8 are among the most challenging digits to distinguish, along with 2-8, 5-8, 4-9, and 7-9. Particularly easy combinations to classify typically include 0 or 1.

The certification performance is remarkably symmetric across the diagonal despite the asymmetry in our convex architectures. In other words, when classifying between digits $i$ and $j$, if a convex classifier exists which generates strong certificates for $i$, then we can generally train an asymmetric classifier that generates strong certificates for $j$. A few exceptions to this can be seen in Figure 5.13; the most notable are the 1-9 versus 9-1 pairs and the 4-8 versus 8-4 pairs. A deeper understanding of how class characteristics affect asymmetric certification is an exciting avenue of future research.

## Randomized Smoothing Noise Level Sweeps

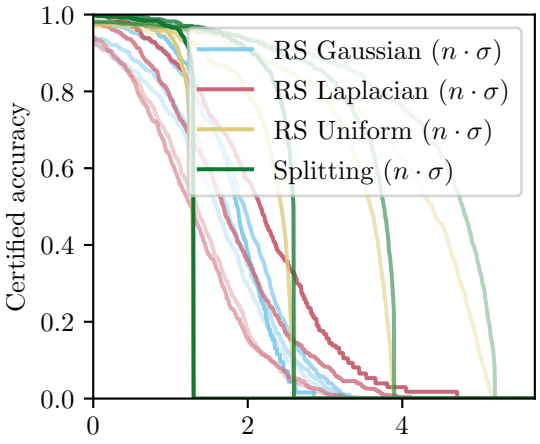
In this section, we reproduce the performance randomized smoothing classifiers under different noise distributions for a range of noise parameters $\sigma$. Namely, we sweep over multiples of base values of $\sigma$ reported in the subcaptions of Figure 5.14, Figure 5.15, and Figure 5.16. The base values of $\sigma$ were set to $\sigma = 0.75$ for the MNIST 3-8, Fashion-MNIST, and CIFAR-10 cats-dogs simulations. For the higher-resolution Malimg simulation, we increase the base noise to $\sigma = 3.5$, matching the highest noise level examined in Levine and Feizi [87]. The epochs used for training were similarly scaled by $n$, starting from the base values provided in the experimental setup above, with the exception of the CIFAR-10 base epochs being



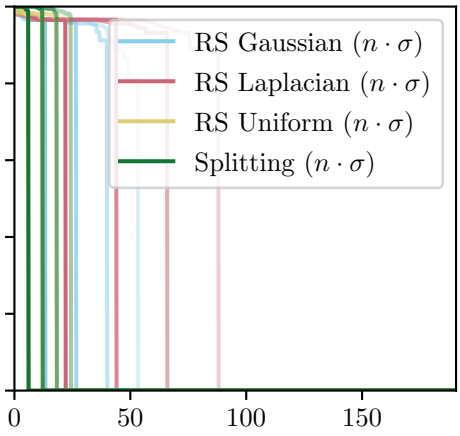

(a) $\ell_1$-certificates. (b) $\ell_2$-certificates. (c) $\ell_\infty$-certificates.

Figure 5.13: Plotting the median certified radii for the MNIST feature-convex architecture over a range of class combinations. The horizontal axis is the class being certified. The MNIST 3-8 experiment considered throughout therefore corresponds to the cell $(3, 8)$ in each plot.

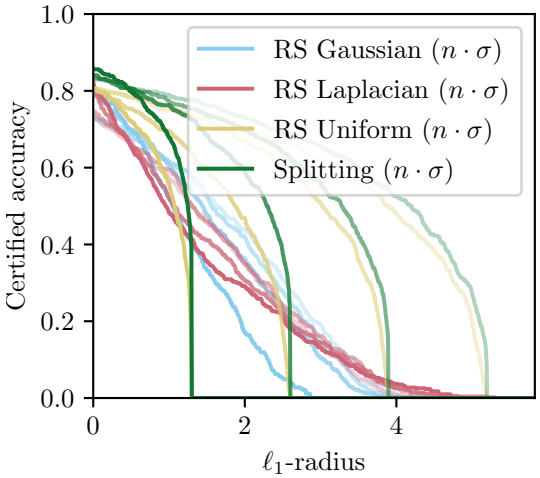
increased to 600 epochs.



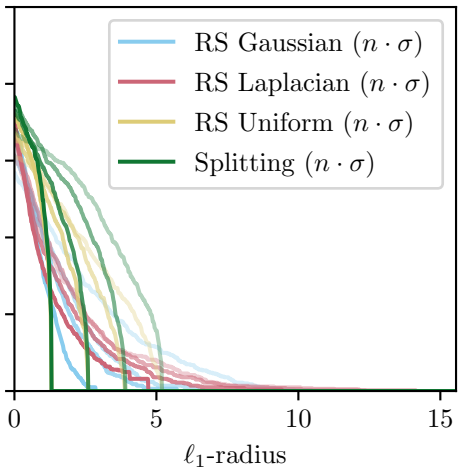

(a) MNIST 3-8, $\sigma = 0.75$.




(b) Malimg, $\sigma = 3.5$.

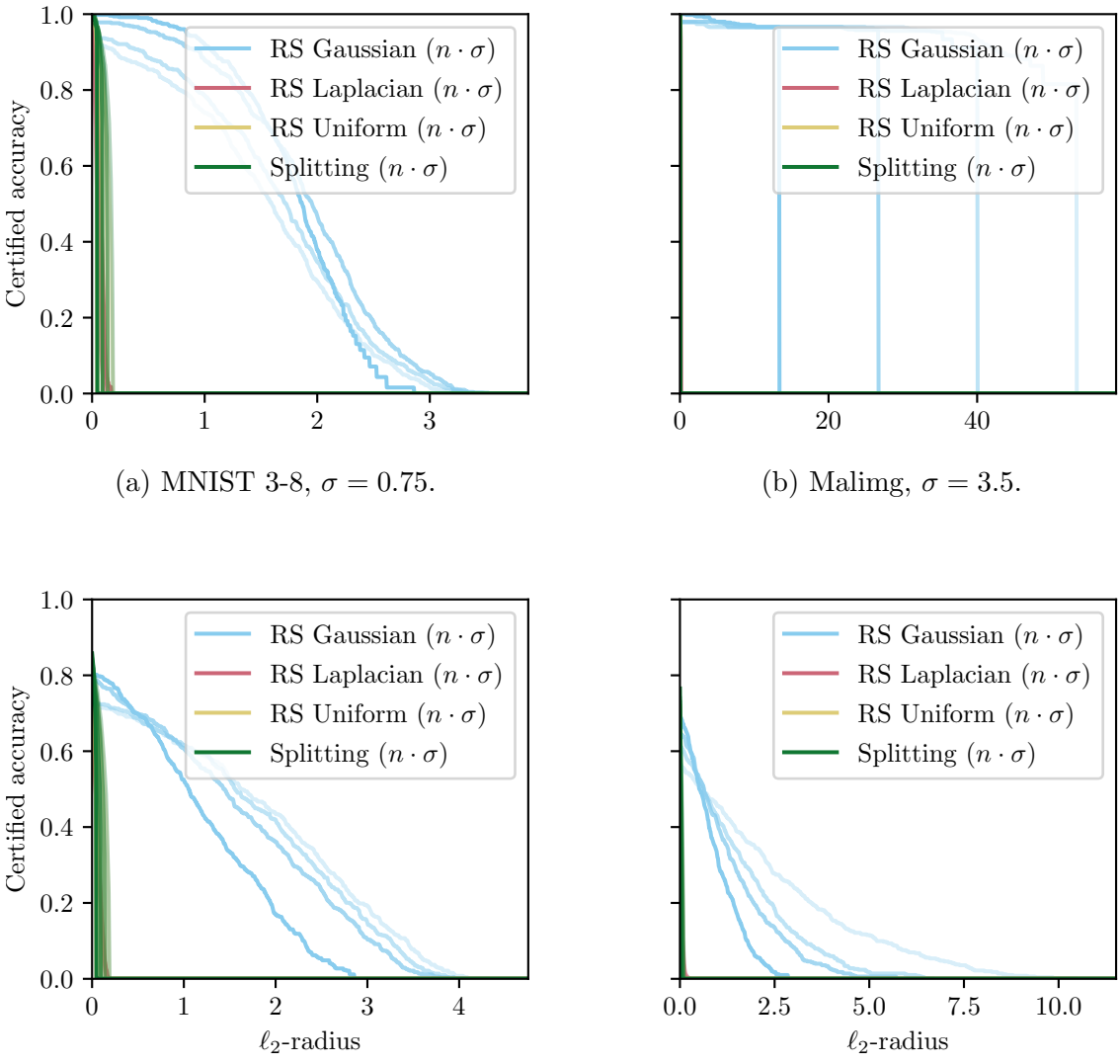



(c) Fashion-MNIST shirts, $\sigma = 0.75$.




(d) CIFAR-10 cats-dogs, $\sigma = 0.75$.

Figure 5.14: Randomized smoothing certified radii sweeps for the $\ell_1$-norm. Line shade indicates value of the integer noise multiplier $n$, with $n$ ranging from 1 (darkest line) to 4 (lightest line).



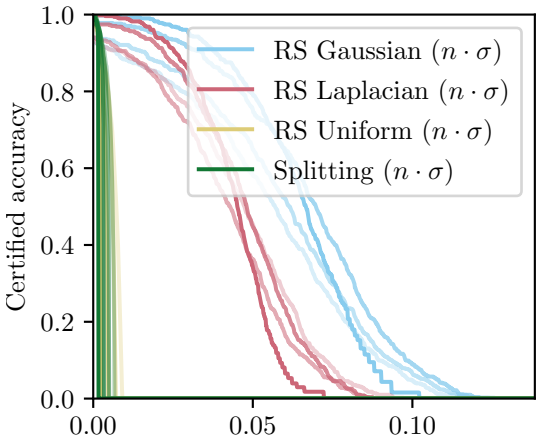

(a) MNIST 3-8, $\sigma = 0.75$.

(b) Malimg, $\sigma = 3.5$.

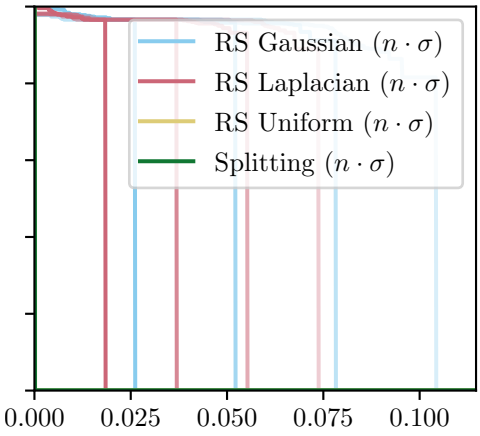
(c) Fashion-MNIST shirts, $\sigma = 0.75$.

(d) CIFAR-10 cats-dogs, $\sigma = 0.75$.

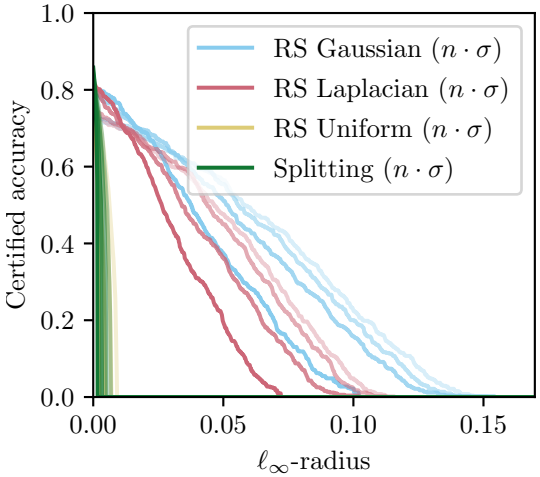
Figure 5.15: Randomized smoothing certified radii sweeps for the $\ell_2$-norm. Line shade indicates value of the integer noise multiplier $n$, with $n$ ranging from 1 (darkest line) to 4 (lightest line). For higher-dimensional inputs (Malimg and CIFAR-10) methods which certify to a different norm and convert are uncompetitive.



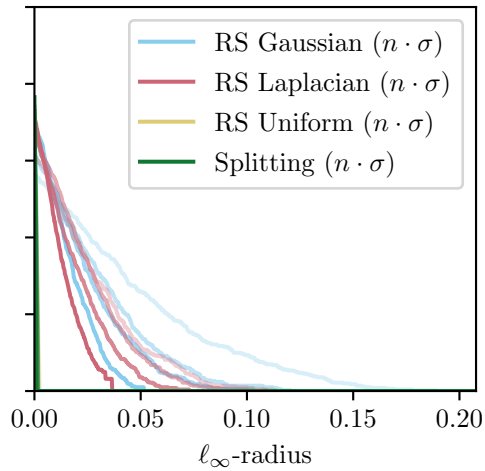

(a) MNIST 3-8, $\sigma = 0.75$.




(b) Malimg, $\sigma = 3.5$.




(c) Fashion-MNIST shirts, $\sigma = 0.75$.




(d) CIFAR-10 cats-dogs, $\sigma = 0.75$.

Figure 5.16: Randomized smoothing certified radii sweeps for the $\ell_\infty$-norm. Line shade indicates value of the integer noise multiplier $n$, with $n$ ranging from 1 (darkest line) to 4 (lightest line).

# Chapter 6

# Locally Biased Randomized Smoothing

Randomized smoothing remains one of the state-of-the-art methods for robustification of a machine learning model with theoretical guarantees. In this chapter, we show that using uniform and unbiased smoothing measures, as is standard in the randomized smoothing literature, relies on the underlying assumption that smooth decision boundaries yield good robustness, which manifests into a robustness-accuracy tradeoff. We generalize the smoothing framework to remove this assumption and learn a locally optimal robustification of the decision boundary based on training data, a method we term *locally biased randomized smoothing*. We prove nontrivial closed-form certified robust radii for the resulting model, avoiding Monte Carlo certifications as used by other smoothing methods. Numerical simulations on synthetic, MNIST, and CIFAR-10 data show a notable increase in the certified radii and accuracy over conventional smoothing.

This chapter is based on the following previously published work:

[8] Brendon G. Anderson and Somayeh Sojoudi, "Certified robustness via locally biased randomized smoothing," *Learning for Dynamics and Control (L4DC)*, 2022.

Related papers:

[16] Yatong Bai, Brendon G. Anderson, Aerin Kim, and Somayeh Sojoudi, "Improving the accuracy-robustness trade-off of classifiers via adaptive smoothing," *SIAM Journal on Mathematics of Data Science*, 2024.

[17] Yatong Bai, Brendon G. Anderson, and Somayeh Sojoudi, "Mixing classifiers to alleviate the accuracy-robustness trade-off," *Learning for Dynamics and Control (L4DC)*, 2024.

[111] Samuel Pfrommer, Brendon G. Anderson, and Somayeh Sojoudi, "Projected random-

ized smoothing for certified adversarial robustness," *Transactions on Machine Learning Research*, 2023.

[7] Brendon G. Anderson, Samuel Pfrommer, and Somayeh Sojoudi, "Towards optimal randomized smoothing: A semi-infinite linear programming approach," *ICML Workshop on Formal Verification of Machine Learning (WFVML)*, 2022.

## 6.1 Introduction

Randomized smoothing, popularized by Lecuyer et al. [85], Li et al. [88], and Cohen, Rosenfeld, and Kolter [35], is commonly accepted as one of the state-of-the-art methods for robustifying large-scale models with rigorous robustness guarantees. Instead of relying on the model's baseline prediction, randomized smoothing assigns the most probable prediction when considering random perturbations of the input. Intuitively, this ensemble approach averages out any outlier inputs that may have drastically changed the prediction, such as adversarially attacked inputs. By using specific probability distributions, e.g., normal or Laplacian, researchers have proven the non-existence of adversarial inputs within balls corresponding to some norm or metric, e.g., $\ell_2$- or $\ell_1$-norm, or Wasserstein metrics [35, 133, 86].

Despite the popularity of randomized smoothing, the method still presents a handful of limitations and open questions, many of which have only recently been considered or remain under investigation. For example, Salman et al. [122] blends randomized smoothing with adversarial training to significantly improve the resulting model's certified robustness. The paper Yang et al. [158] determines the geometry of optimal smoothing distributions for $\ell_1$-, $\ell_2$-, and $\ell_\infty$-norm bounded attacks. Contrarily, Zhang et al. [166] considers optimizing the base classifier to maximize the robust radius for a fixed distribution. The work Dvijotham et al. [44] develops a measure-theoretic approach for robustness certification of models smoothed using arbitrary distributions. Many negative results have also been shown, e.g., Mohapatra et al. [100] shows that smoothed classifiers suffer from a "shrinking phenomenon": decision regions shrink and eventually vanish as the variance of the smoothing distribution increases. Many works have also identified a robustness-accuracy tradeoff in relation to the smoothness of models [137, 79, 160, 60], a limitation we discuss in Section 6.2 and address in our proposed approach. Finally, some recent works have considered more general formulations of randomized smoothing in an attempt to increase certified radii—we discuss these works in-depth in Section 6.2.

Randomized smoothing is usually considered in a static classification setting, and this is the setting we study. Nonetheless, such works are actively being incorporated into dynamic settings with more general outputs, e.g., smoothing of neural network policies in reinforcement learning [81, 152]. Consequently, the results of this chapter may be of interest in more general dynamic learning problems than the static classification setting that we present.

### Contributions

We show that standard randomized smoothing methods possess the informal assumption that making models smoother is a good surrogate for making them more robust. This manifests into a robustness-accuracy tradeoff, and we show that to eradicate the assumption it is necessary to generalize to biased and input-dependent distributions. Accordingly, we propose *locally biased randomized smoothing*, which uses training data to directly learn model robustification without relying on the assumption that smoothness yields robustness. For the binary classification setting, we obtain a closed-form smoothed model with closed-form certified radii for arbitrary norms, overcoming the Monte Carlo estimations used by most current methods. Our numerical simulations demonstrate an increased accuracy both on clean and adversarially attacked data, as well as increased certified radii.

### Outline

In Section 6.2, we review the mathematical foundations of randomized smoothing, the limitations of the method, and discuss recent works that attempt to generalize past some of these limitations. In Section 6.3, we develop our smoothing scheme, entitled locally biased randomized smoothing, for binary linear base classifiers, and derive a certified robust radius. We generalize the method and certified radius to nonlinear classifiers in Section 6.4. Numerical simulations are presented in Section 6.5 and conclusions are drawn in Section 6.6.

## 6.2 Randomized Smoothing: Review, Limitations, and Generalizations

### Preliminaries

We denote by $\mathcal{P}(\mathbb{R}^d)$ the set of probability measures on $\mathbb{R}^d$ equipped with the Borel $\sigma$-algebra. If $\mu \in \mathcal{P}(\mathbb{R}^d)$ and $g\colon \mathbb{R}^d \to \mathbb{R}^n$ has $\mu$-integrable components $g_i\colon \mathbb{R}^d \to \mathbb{R}$, $i \in \{1, 2, \dots, n\}$, we define the expectation $\mathbb{E}_{x\sim\mu} g(x) \coloneqq \int_{\mathbb{R}^d} g(x) d\mu(x) = \big(\int_{\mathbb{R}^d} g_1(x) d\mu(x), \dots, \int_{\mathbb{R}^d} g_n(x) d\mu(x)\big)$. We assume $\mu$-integrability whenever we write $\mathbb{E}_{x\sim\mu} g(x)$ or $\int_{\mathbb{R}^d} g(x) d\mu(x)$. The normal distribution on $\mathbb{R}^d$ with mean $\overline{x}$ and covariance $\Sigma$ is denoted by $\mathcal{N}(\overline{x}, \Sigma)$. The distribution function of $\mathcal{N}(0, 1)$ on $\mathbb{R}$ is denoted by $\Phi$, which we recall has a well-defined inverse. The dual norm of a norm $\|\cdot\|\colon \mathbb{R}^d \to [0, \infty)$ is denoted by $\|\cdot\|_*$, and is given by $\|y\|_* = \sup\{x^\top y : \|x\| \leq 1\}$ for all $y \in \mathbb{R}^d$. Throughout, we allow $\|\cdot\|$ to denote an arbitrary norm, the domain of which will be clear from context. We let $\rho\colon \mathbb{R}^d \times \mathbb{R}^d \to [0, \infty)$ denote the metric defined by $\rho(x, y) = \|x - y\|_2$. For ease of exposition, we assume that all $\arg\max$ and $\arg\min$ yield singleton sets.[1]

[1]This assumption is violated in some cases, e.g., when considering inputs on decision boundaries. In practice, however, we are not concerned with these pathological cases, as they correspond to sets of zero Lebesgue measure.

Consider a classifier $f\colon \mathbb{R}^d \to \{1, 2, \ldots, n\}$ defined by $f(x) \in \arg\max_{i\in\{1,2,\ldots,n\}} g_i(x)$, where $g\colon \mathbb{R}^d \to \mathbb{R}^n$. In this chapter, we consider robustifying $f$ using the randomized smoothing framework.

## Review of Randomized Smoothing

Instead of assigning the class $f(x)$ to an input $x \in \mathbb{R}^d$, randomized smoothing assign the expected class under $f$ of random perturbations of $x$. This amounts to choosing a smoothing measure $\mu \in \mathcal{P}(\mathbb{R}^d)$ and replacing $f$ with the smoothed classifier $f^\mu\colon \mathbb{R}^d \to \{1, 2, \ldots, n\}$ defined by $f^\mu(x) \in \arg\max_{i\in\{1,2,\ldots,n\}} g_i^\mu(x)$, with $g^\mu\colon \mathbb{R}^d \to \mathbb{R}^n$ given by $g^\mu(x) = \mathbb{E}_{\epsilon\sim\mu} g(x+\epsilon)$.

Some works consider directly manipulating the hard classifier $f$ without regard to the soft classifier $g$ [35, 133]. In contrast, we smooth the soft classifier $g$ before the arg max is taken, as is done in many other works [122, 164, 86, 82]. Smoothing $g$, which generalizes smoothing $f$ [122], takes into account the confidence of the base classifier, whereas hard smoothing does not [82]. Consequently, we concern ourselves only with soft smoothing.

Intuitively, randomized smoothing flattens jagged regions of the decision boundary, where adversarial inputs are conjectured to exist [48]. This intuition can be formalized in the framework of convolution. If $\mu$ has density $\phi\colon \mathbb{R}^d \to [0, \infty)$ (with respect to Lebesgue measure) that is symmetric (i.e., $\phi(-x) = \phi(x)$), then randomized smoothing is the convolution

$$g^\mu(x) = \int_{\mathbb{R}^d} \phi(\epsilon) g(x - \epsilon) d\epsilon =: \phi * g(x).$$

In general, the convolution $g^\mu = \phi * g$ is smoother than the functions $\phi$ and $g$ being convolved [55]. From the control and signal processing perspective, this convolutional representation shows that randomized smoothing acts as a low-pass filter on $g$. Upon attenuating the high-frequency behavior in $g$ via smoothing, the radius of robustness around clean inputs has been found to increase, with certified robust radii given for special cases of the smoothing measure $\mu$.

The most popular form of randomized smoothing, introduced in Cohen, Rosenfeld, and Kolter [35], takes the smoothing measure $\mu$ to be that of the normal distribution $\mathcal{N}(0, \sigma^2 I)$. We refer to this scheme as *normal smoothing*. In this case, $g^\mu$ becomes the Weierstrass transform of $g$, which is well known to attenuate high-frequency components in $g$. Since evaluating $g^\mu(x)$ in this case requires computing an integral that has no closed-form formula in general, implementing normal smoothing typically requires Monte Carlo estimation. The authors of Cohen, Rosenfeld, and Kolter [35] proved a certified robust $\ell_2$-radius for normal smoothing, which must also be estimated via Monte Carlo methods. We recall the result below in terms of soft classifier smoothing—see Zhai et al. [164] for this generalization.

**Theorem 16** ([35, 164])**.** *Assume that $g\colon \mathbb{R}^d \to [0, 1]^n$. Let $\sigma^2 > 0$, and let $\mu$ be the probability measure of the normal distribution $\mathcal{N}(0, \sigma^2 I)$. Consider a point $x \in \mathbb{R}^d$ and let $y = f^\mu(x) \in \arg\max_{i\in\{1,2,\ldots,n\}} g_i^\mu(x)$ and $y' \in \arg\max_{i\in\{1,2,\ldots,n\}\setminus\{y\}} g_i^\mu(x)$. Then $f^\mu(x+\delta) = y$*

*for all* $\delta \in \mathbb{R}^d$ *such that*

$$\|\delta\|_2 \leq r_\sigma(x) \coloneqq \frac{\sigma}{2}\left(\Phi^{-1}(g_y^\mu(x)) - \Phi^{-1}(g_{y'}^\mu(x))\right).$$

## Limitations of Randomized Smoothing

We remark two important restrictions on the measure $\mu$ that are common to most randomized smoothing methods in the literature: 1) $\mu$ is uniform with respect to the input $x$, and 2) $\mu$ is centered at $0 \in \mathbb{R}^d$. In this section, we formalize these restrictions and show why they should be relaxed.

We begin with Proposition 19 below, which, as a direct consequence of the uniform smoothing measure, shows that $g^\mu$ is necessarily "more constant" than $g$. This forces classification to remain constant over larger regions of the input space, but when these regions become too large, the accuracy of the predictions degrades [79, 160]. The proposition has an obvious generalization to the case of local Lipschitzness.

**Definition 14.** Let $L \in \mathbb{R}$, and let $\|\cdot\|$ and $\|\cdot\|'$ be norms on $\mathbb{R}^d$ and $\mathbb{R}^n$, respectively. A function $h\colon \mathbb{R}^d \to \mathbb{R}^n$ is called *$L$-Lipschitz in norms* $(\|\cdot\|, \|\cdot\|')$ if for all $x, x' \in \mathbb{R}^d$ it holds that $\|h(x) - h(x')\|' \leq L\|x - x'\|$.

**Proposition 19.** *If $g$ is $L$-Lipschitz in norms $(\|\cdot\|, \|\cdot\|')$, then $g^\mu$ is $L$-Lipschitz in norms $(\|\cdot\|, \|\cdot\|')$.*

*Proof.* Suppose that $g$ is $L$-Lipschitz in norms $(\|\cdot\|, \|\cdot\|')$ and let $x, x' \in \mathbb{R}^d$. Then $\|g^\mu(x) - g^\mu(x')\|' \leq \int_{\mathbb{R}^d} \|g(x+\epsilon) - g(x'+\epsilon)\|' d\mu(\epsilon) \leq \int_{\mathbb{R}^d} L\|x - x'\| d\mu(\epsilon) = L\|x - x'\|$. □

We next show that smoothing measures centered at the origin $0 \in \mathbb{R}^d$ cannot change a linear decision boundary, even if they are allowed to depend on the input $x$ (which we denote by $\mu_x$). This is true even when doing so would increase robustness with respect to the data distribution at hand. Thus, unbiased smoothing distributions cannot robustify linear classifiers.

**Definition 15.** A measure $\mu \in \mathcal{P}(\mathbb{R}^d)$ is called *unbiased* if $\mathbb{E}_{\epsilon\sim\mu}\epsilon = 0$. The measure $\mu$ is called *biased* if it is not unbiased.

**Proposition 20.** *Suppose that $g$ is affine, namely $g(x) = Ax + b$ for some $A \in \mathbb{R}^{n\times d}$ and $b \in \mathbb{R}^n$. Consider $g^\mu$ with input-dependent smoothing measure $\mu_x$, so that $g^\mu(x) = \mathbb{E}_{\epsilon\sim\mu_x} g(x+\epsilon)$. If $\mu_x$ is unbiased for all $x \in \mathbb{R}^d$, then $f^\mu = f$.*

*Proof.* Suppose that $\mu_x$ is unbiased for all $x \in \mathbb{R}^d$. Then $g^\mu(x) = \mathbb{E}_{\epsilon\sim\mu_x}(A(x+\epsilon) + b) = Ax + b + A\mathbb{E}_{\epsilon\sim\mu_x}\epsilon = g(x)$. Hence, $g^\mu = g$, and consequently $f^\mu = f$. □

When the smoothing measure is unbiased and uniform with respect to the input, we refer to the scheme as *standard smoothing*. Together, the two limitations in Proposition 19 and

Proposition 20 point to a fundamental informal assumption that underlies standard smoothing: *making classifiers smoother, as characterized by their Lipschitz constant or the linearity of their decision boundaries, is a good surrogate for increasing robustness.* Although standard smoothing has been shown to work well in many settings, the assumption that smoothness yields robustness is fundamentally flawed, since minimizing the Lipschitz constant degrades accuracy [79, 160]. If the assumption were to hold, then a constant classifier, obtained, e.g., by letting $\sigma^2 \to \infty$ in Theorem 16, would be the most robust classifier, which is nonsensical when we take accuracy into account. The work Madry et al. [95] corroborates this conclusion, arguing that simultaneous accuracy and robustness often requires a complicated decision boundary. Thus, our goal should be to directly increase robustness with respect to the data distribution, without resorting to surrogate notions such as smoothness. To do so, Proposition 19 and Proposition 20 show that we must generalize the smoothing framework to allow for input-dependent and biased smoothing measures.

## Generalizing Randomized Smoothing and Related Works

Henceforth, we consider $\mu = \{\mu_x \in \mathcal{P}(\mathbb{R}^d) : x \in \mathbb{R}^d\}$ with all $\mu_x$ possibly biased, and define

$$g^{\mu}(x) = \mathbb{E}_{\epsilon \sim \mu_x} g(x + \epsilon). \tag{6.1}$$

We refer to this scheme as *generalized smoothing*.

A handful of recent works have considered generalized smoothing (although mostly in a blind attempt to increase robust radii, not due to recognition of the flawed informal assumption previously discussed). For example, Wang et al. [141] uses normal distributions $\mathcal{N}(0, \sigma_i^2 I)$ to maximize $\ell_2$-robust regions around every training point $x_i$. If a test input $x$ is not contained in any such certified region, they optimize a new variance $\sigma^2(x)$ to allocate a certified region around $x$, which is then used for future classification. Not only is this restricted to $\ell_2$-adversaries and computationally heavy due to two-stage training, but also the resulting classifier depends on the order of incoming inputs, introducing new performance and robustness concerns. The works Alfarra et al. [1] and Eiras et al. [46] take a similar memory-based approach, with the latter allowing for specific anisotropic certified regions, and hence also yield order-dependent classifiers that change at test time.

Chen et al. [30] uses $\mu_x$ being the measure associated with $\mathcal{N}(0, \sigma^2(x)I)$, where the variance maximizes the certified $\ell_2$-radius of normal smoothing; $\sigma^2(x) \in \arg\max_{\sigma^2 > 0} r_\sigma(x)$. The authors of Súkeník, Kuvshinov, and Günnemann [129] show that, in addition to suffering from the curse of dimensionality, the robustness certificate issued by this work is actually invalid in practice. To see this, consider a fixed input $x \in \mathbb{R}^d$ and its chosen smoothing measure $\mu_x \in \mathcal{P}(\mathbb{R}^d)$. This work certifies that, for $\delta \in \mathbb{R}^d$ within a specified robust radius, $x + \delta$ is classified the same as $x$ under the smoothed classifier using $\mu_x$. However, the classifier uses the measure $\mu_{x+\delta} \neq \mu_x$ when classifying $x + \delta$ (since the measure is optimized per-input), and therefore, the robustness certificate does not apply to the actual classifier used at test time. To overcome this, Súkeník, Kuvshinov, and Günnemann [129] proposes

a specific parameterization of $\sigma^2(x)$ for generalized smoothing with $\mathcal{N}(0, \sigma^2(x))$ that leads to valid robust $\ell_2$-radii, but they find that the certified radii do not notably increase over normal smoothing in practice.

These works showcase the importance and timeliness of generalized smoothing, and highlight its difficulties in deriving robust radii. In the sequel, we use generalized smoothing to learn a closed-form manipulation of the decision boundary from data, granting robust radii for arbitrary norms that are mathematically rigorous and practically valid.

## 6.3 Robustifying Binary Linear Classifiers

Since standard smoothing is unable to robustify linear classifiers, we start from the basics: we assume a binary linear setting, with $g\colon \mathbb{R}^d \to \mathbb{R}$, $g(x) = a^\top x + b$, and $f(x) = \operatorname{sign}(g(x))$.

### Optimal Robustification Under the Direction Oracle

Consider a point $x \in \mathbb{R}^d$. We start by assuming that we know that the true class of $x$ is 1. Formally, we assume that there exists an oracle function $y\colon \mathbb{R}^d \to \{-1, 1\}$ that gives the true class of $x$, and for this point $x$ it holds that $y(x) = 1$. With this in mind, we remark that

$$g^\mu(x) = \mathbb{E}_{\epsilon\sim\mu_x}(a^\top(x+\epsilon) + b) = g(x) + a^\top \mathbb{E}_{\epsilon\sim\mu_x}\epsilon. \tag{6.2}$$

Since $x$ has true class 1, robustification at $x$ is equivalent to $g^\mu(x) > g(x)$, so that the neighborhood around $x$ classified into class 1 increases in size. Hence, our goal amounts to maximizing $a^\top \mathbb{E}_{\epsilon\sim\mu_x}\epsilon$. Without constraints on $\mu_x$, this optimization would be unbounded. Therefore, we consider measures with bounded expectation $\mathbb{E}_{\epsilon\sim\mu_x}\epsilon$, and we find that an optimal $\mu_x$ is one attaining[2] $a^\top \mathbb{E}_{\epsilon\sim\mu_x}\epsilon = \sup\left\{a^\top \mathbb{E}_{\epsilon\sim\nu_x}\epsilon : \|\mathbb{E}_{\epsilon\sim\nu_x}\epsilon\| \le \alpha,\ \nu_x \in \mathcal{P}(\mathbb{R}^d)\right\} = \alpha\|a\|_*$.

If, on the other hand, the true class of $x$ is $-1$, then an optimal $\mu_x$ is one attaining $a^\top \mathbb{E}_{\epsilon\sim\mu_x}\epsilon = -\alpha\|a\|_*$. Therefore, for general $x \in \mathbb{R}^d$, we find that the optimal smoothed classifier is given by

$$g^\mu(x) = g(x) + \alpha y(x)\|a\|_*, \tag{6.3}$$

where $y(x) \in \{-1, 1\}$ is the oracle class assigned to $x$. We call $y$ the *direction oracle*, since its value at $x$ determines which direction to push the decision boundary (either in the $a$ or $-a$ direction).

### Approximating the Direction Oracle

Suppose that we have a subset of training data $\{(x_1, y_1), \ldots, (x_N, y_N)\} \subseteq \mathbb{R}^d \times \{-1, 1\}$. We have the true classes $y(x_i) = y_i$ for these data points, and therefore the optimal robustified classification is given by $g^\mu(x_i) = g(x_i) + \alpha y_i\|a\|_*$. However, for a general point $x$, we do not have access to $y(x)$ (if we did, we would not need to learn anything). Thus,

[2]This optimization is always attained by an appropriately chosen Dirac measure.

for $x \notin \{x_1, x_2, \ldots, x_N\}$, we propose to approximate $y(x)$ based on the given data. This approximation of the direction oracle will be denoted by $\hat{y}\colon \mathbb{R}^d \to \{-1, 1\}$, and will be used in place of $y$ in our smoothed classifier (6.3).

It is insightful to note that $g^\mu$ does not use the oracle value $y(x)$ to directly classify $x$. Rather, it is used to encode which direction to push the decision boundary for robustification. Thus, a "good" approximation of the direction oracle is one that encodes a "good" manipulation of the decision boundary to achieve robustification, not necessarily those that accurately predict the true label. With this insight in mind, we propose the approximate direction oracle to be the 1-nearest neighbor $\hat{y}(x) = y_{i^\star(x)}$, where $i^\star\colon \mathbb{R}^d \to \{1, 2, \ldots, N\}$ is defined[3] by $i^\star(x) \in \arg\min_{i \in \{1,2,\ldots,N\}} \rho(x, x_i)$. This choice is natural since robustness is a local property and most classifiers are continuous. Note that $\hat{y}(x_i)$ recovers $y_i$ for the data $x_i$. In Theorem 17 and Theorem 18, we will see that this approximate direction oracle yields closed-form certified robust radii. Using other approximate direction oracles could present an interesting direction for future research (for example, $k$-nearest neighbors or learning a neural network to output labels that optimize the induced robustness).

## Locally Biased Randomized Smoothing

With our smoothing scheme now finalized, the classifier becomes

$$g^\mu(x) = g(x) + \alpha y_{i^\star(x)} \|a\|_*. \tag{6.4}$$

We remark the two underlying features that distinguish our scheme from standard smoothing: the direction oracle encodes an informed manipulation of the decision boundary that is determined *locally* based on data, and this manipulation optimized for robustness using *biased* smoothing measures. For this reason, we term our framework *locally biased randomized smoothing*.

In contrast to standard smoothing, $g^\mu$ may be nonlinear when the data informs us that nonlinearity is required to increase robustness. We will continue to refer to $f^\mu$ (and $g^\mu$) as the smoothed classifier, despite the fact that it may be less smooth than the base classifier. Unlike normal smoothing, our classifier requires no Monte Carlo estimation, since the smoothing distribution has a closed-form expectation. As $\alpha \to \infty$, the classifier $f^\mu$ converges pointwise to the 1-nearest neighbor classifier. On the other hand, normal smoothing converges pointwise to a constant function as $\sigma^2 \to \infty$. Thus, we may view both methods as interpolating between the base classifier, typically optimized for clean accuracy, and a limiting classifier. With this perspective, a good limiting classifier is one that is optimized for robust accuracy, and we posit that our data-informed 1-nearest neighbor better serves this purpose than the constant function. Indeed, it has been shown that 1-nearest neighbor classifiers are accurate and certifiably robust when the data follows mild separation properties [145], which justifies our use of the 1-nearest neighbor approximate direction oracle.

[3]This is well-defined under our assumption that the arg min yields a singleton set.

The choice of norm $\|\cdot\|$ and bias level $\alpha$ are left to the user. As will soon be seen in Theorem 17 and Theorem 18, the certified radii are in terms of $\|\cdot\|$, so the norm should be chosen according to the threat model at hand. For example, it is common to use $\|\cdot\|_\infty \colon \delta \mapsto \max_{i\in\{1,2,\dots,d\}} |\delta_i|$ in image classification settings. The effects of the bias level $\alpha$ are explored experimentally in Section 6.5.

We now provide closed-form certified radii for linear base classifiers.

**Theorem 17.** *Consider $x \in \mathbb{R}^d$ and fix $i = i^\star(x)$. Then $f^\mu(x+\delta) = f^\mu(x)$ for all $\delta \in \mathbb{R}^d$ such that*

$$\|\delta\| < r^\mu_{\text{linear}}(x) \coloneqq \min\left\{ \frac{|g^\mu(x)|}{\|a\|_*}, \min\left\{ \frac{\rho(x,x_j)^2 - \rho(x,x_i)^2}{2\|x_i - x_j\|_*} : y_j \neq y_i,\ j \in \{1,2,\dots,N\} \right\} \right\}.$$

*Proof.* Let $\delta \in \mathbb{R}^d$ be such that $\|\delta\| < r^\mu_{\text{linear}}(x)$. Since $\|\delta\| \leq \frac{\rho(x,x_j)^2 - \rho(x,x_i)^2}{2\|x_i - x_j\|_*}$ for all $j \in \{1,2,\dots,N\}$ such that $y_j \neq y_i$, it holds that

$$\begin{aligned}
\rho(x+\delta,x_j)^2 - \rho(x+\delta,x_i)^2 &= (x+\delta-x_j)^\top(x+\delta-x_j) - (x+\delta-x_i)^\top(x+\delta-x_i) \\
&= 2(x_i - x_j)^\top(x+\delta) + \|x_j\|_2^2 - \|x_i\|_2^2 \\
&= 2x_i^\top x - 2x_j^\top x + \|x_j\|_2^2 - \|x_i\|_2^2 + 2(x_i - x_j)^\top\delta \\
&= (x-x_j)^\top(x-x_j) - (x-x_i)^\top(x-x_i) + 2(x_i-x_j)^\top\delta \\
&= \rho(x,x_j)^2 - \rho(x,x_i)^2 + 2(x_i-x_j)^\top\delta \\
&\geq \rho(x,x_j)^2 - \rho(x,x_i)^2 - 2|(x_i-x_j)^\top\delta| \\
&\geq \rho(x,x_j)^2 - \rho(x,x_i)^2 - 2\|x_i-x_j\|_*\|\delta\| \\
&\geq 0
\end{aligned}$$

for all such $j$. Hence, $\rho(x+\delta,x_i) \leq \rho(x+\delta,x_j)$ for all $j$ such that $y_j \neq y_i$. Therefore, it must be that $i^\star(x+\delta) \in \{j \in \{1,2,\dots,N\} : y_j = y_i\}$, and hence $y_{i^\star(x+\delta)} = y_i$. Therefore,

$$\begin{aligned}
g^\mu(x+\delta) &= g(x+\delta) + \alpha y_{i^\star(x+\delta)}\|a\|_* \\
&= a^\top(x+\delta) + b + \alpha y_i\|a\|_* \\
&= g(x) + \alpha y_i\|a\|_* + a^\top\delta \\
&= g^\mu(x) + a^\top\delta.
\end{aligned}$$

This gives that

$$|g^\mu(x+\delta) - g^\mu(x)| = |a^\top\delta| \leq \|a\|_*\|\delta\| < |g^\mu(x)|,$$

where the last inequality follows from the fact that $\|\delta\| < \frac{|g^\mu(x)|}{\|a\|_*}$. Therefore,

$$-|g^\mu(x)| < g^\mu(x+\delta) - g^\mu(x) < |g^\mu(x)|, \tag{6.5}$$

so

$$g^\mu(x) - |g^\mu(x)| < g^\mu(x+\delta) < g^\mu(x) + |g^\mu(x)|.$$

If $g^\mu(x) \geq 0$, then the left-hand inequality gives that $0 = g^\mu(x) - |g^\mu(x)| < g^\mu(x+\delta)$, whereas if $g^\mu(x) < 0$, then the right-hand inequality gives that $g^\mu(x+\delta) < g^\mu(x) + |g^\mu(x)| = 0$. In both cases, $\operatorname{sign}(g^\mu(x+\delta)) = \operatorname{sign}(g^\mu(x))$, which proves that $f^\mu(x+\delta) = f^\mu(x)$, as desired. □

Note that $r^\mu_{\text{linear}}(x) \geq 0$ for all $x \in \mathbb{R}^d$. Furthermore, the term $\frac{|g^\mu(x)|}{\|a\|_*}$ is the distance (in norm $\|\cdot\|$) from $x$ to the hyperplane $\{x' \in \mathbb{R}^d : g(x') + \alpha y_{i^\star(x)}\|a\|_* = 0\}$ [98], which is the decision boundary of $f$ offset by $\alpha y_{i^\star(x)}\|a\|_*$. Thus, when $y_{i^\star(x)} = f(x)$, meaning that the base classifier and the approximate direction oracle agree at $x$, we have $\frac{|g^\mu(x)|}{\|a\|_*} = \frac{|g(x)|}{\|a\|_*} + \alpha$. Therefore, this term of the certified radius is strictly larger than the robust radius $r(x) \coloneqq \frac{|g(x)|}{\|a\|_*}$ under $f$. The term $\frac{\rho(x,x_j)^2 - \rho(x,x_i)^2}{2\|x_i - x_j\|_*}$ with $y_j \neq y_i$ quantifies how close $x$ is to a data point of class different from that assigned by the approximate direction oracle. If $x$ is sufficiently far from such data points and $y_{i^\star(x)} = f(x)$, then $r^\mu_{\text{linear}}(x) = r(x) + \alpha$, and as a result the robust radius increases by $\alpha$. This need not happen in general. If $x$ is relatively close to a data point of class $y_j \neq y_i$ or if $y_{i^\star(x)} \neq f(x)$, then $r^\mu_{\text{linear}}(x)$ may be less than $r(x)$. This is expected, since in these cases the nearby data informs us that $x$ may not belong to class $f(x)$ predicted by the base classifier. These are the sacrificial points that may move closer to the resulting decision boundary in the name of robustifying where the data says to. Such points must exist since it is not possible to robustify everywhere simultaneously.

## 6.4 Extension to Nonlinear Classifiers

We now extend the approach to binary nonlinear classifiers. Assume that $g\colon \mathbb{R}^d \to \mathbb{R}$ is continuously differentiable and possibly nonlinear. For $x, \epsilon \in \mathbb{R}^d$, the mean value theorem gives that $g(x+\epsilon) = g(x) + \nabla g(x')^\top \epsilon$ for some $x'$ on the line segment between $x$ and $x+\epsilon$. By continuity of $\nabla g$, we have that $\lim_{x' \to x} \nabla g(x') = \nabla g(x)$. Therefore, informally, $g(x+\epsilon) \approx g(x) + \nabla g(x)^\top \epsilon$ for all $\epsilon$ with small norm. Consequently, instead of using the expectation of $g(x+\epsilon)$ to define the smoothed classifier, we propose to use the expectation of $g(x) + \nabla g(x)^\top \epsilon$. In doing so, we define $g^\mu$ by $g^\mu(x) = g(x) + \nabla g(x)^\top \mathbb{E}_{\epsilon \sim \mu_x} \epsilon$. Unlike the linear case, $g^\mu(x)$ may not equal $\mathbb{E}_{\epsilon \sim \mu_x} g(x+\epsilon)$. This modification enables us to prove certified radii while maintaining notable increases in robust accuracy in practice. When the base classifier is linear, $g^\mu$ reduces to the prior formulation (6.2).

Performing the same analysis as in the linear case, the smoothed classifier becomes

$$g^\mu(x) = g(x) + \alpha y_{i^\star(x)} \|\nabla g(x)\|_*. \tag{6.6}$$

We emphasize that (6.6) shows that the convergence of $f^\mu$ to the 1-nearest neighbor classifier as $\alpha \to \infty$ is nonlinear and nontrivial. In particular, (6.6) is a data-informed nonlinear manipulation of the decision boundary. The smoothed classifier (6.6) cannot be justified directly without relying on our methodology; $g^\mu$ is not simply a naive linear interpolation between $g$ and the 1-nearest neighbor.

Interestingly, when $\|\cdot\| = \|\cdot\|_\infty$, the value $g^\mu(x)$ approximates the soft classification (under $g$) of the adversarial attack $x_{\text{FGSM}} \coloneqq x + \alpha \operatorname{sign}(\nabla \ell(x))$ generated by the well-known

fast gradient sign method (FGSM) with loss $\ell(\cdot) = y_{i^\star(\cdot)} g(\cdot)$ to be maximized [62]. High values of this loss are actually beneficial with respect to the given data, and therefore an alternative interpretation of our method is as a preemptive "anti-attack" everywhere in the input space.

Theorem 18 below generalizes the certified radii of Theorem 17 to nonlinear base classifiers. The result uses a global Lipschitz constant of the gradient, which is easily modified to use local constants if desired (e.g., the local Lipschitz constant over a $\|\cdot\|$-norm ball at $x$ of radius $r^\mu_{\text{data}}(x)$). In general, local constants give stronger bounds but are difficult to compute. See related works on estimating and upper-bounding Lipschitz constants, e.g., Weng et al. [149] and Fazlyab et al. [51].

**Assumption 8.** The gradient $\nabla g \colon \mathbb{R}^d \to \mathbb{R}^d$ is $L$-Lipschitz in norms $(\|\cdot\|, \|\cdot\|_*)$ for some $L > 0$.

**Theorem 18.** *Suppose that Assumption 8 holds. Consider $x \in \mathbb{R}^d$ and fix $i = i^\star(x)$. Then $f^\mu(x+\delta) = f^\mu(x)$ for all $\delta \in \mathbb{R}^d$ such that $\|\delta\| < r^\mu(x) \coloneqq \min\{r^\mu_{\text{base}}(x), r^\mu_{\text{data}}(x)\}$, where*

$$r^\mu_{\text{base}}(x) = \frac{\sqrt{(\alpha L + \|\nabla g(x)\|_*)^2 + 4L|g^\mu(x)|} - (\alpha L + \|\nabla g(x)\|_*)}{2L},$$
$$r^\mu_{\text{data}}(x) = \min\left\{ \frac{\rho(x, x_j)^2 - \rho(x, x_i)^2}{2\|x_i - x_j\|_*} : y_j \neq y_i,\ j \in \{1, 2, \ldots, N\} \right\}.$$

*Proof.* Let $\delta \in \mathbb{R}^d$ be such that $\|\delta\| < r^\mu(x)$. Since $\|\delta\| \leq \frac{\rho(x,x_i)^2 - \rho(x,x_j)^2}{2\|x_i - x_j\|_*}$ for all $j \in \{1, 2, \ldots, N\}$ such that $y_j \neq y_i$, it holds that

$$\begin{aligned}
\rho(x+\delta, x_j)^2 - \rho(x+\delta, x_i)^2 &= (x+\delta-x_j)^\top (x+\delta-x_j) - (x+\delta-x_i)^\top (x+\delta-x_i) \\
&= 2(x_i - x_j)^\top (x+\delta) + \|x_j\|_2^2 - \|x_i\|_2^2 \\
&= 2x_i^\top x - 2x_j^\top x + \|x_j\|_2^2 - \|x_i\|_2^2 + 2(x_i - x_j)^\top \delta \\
&= (x - x_j)^\top (x - x_j) - (x - x_i)^\top (x - x_i) + 2(x_i - x_j)^\top \delta \\
&= \rho(x, x_j)^2 - \rho(x, x_i)^2 + 2(x_i - x_j)^\top \delta \\
&\geq \rho(x, x_j)^2 - \rho(x, x_i)^2 - 2|(x_i - x_j)^\top \delta| \\
&\geq \rho(x, x_j)^2 - \rho(x, x_i)^2 - 2\|x_i - x_j\|_* \|\delta\| \\
&\geq 0
\end{aligned}$$

for all such $j$. Hence, $\rho(x+\delta, x_i) \leq \rho(x+\delta, x_j)$ for all $j$ such that $y_j \neq y_i$. Therefore, it must be that $i^\star(x+\delta) \in \{j \in \{1, 2, \ldots, N\} : y_j = y_i\}$, and hence $y_{i^\star(x+\delta)} = y_i$. Therefore,

$$\begin{aligned}
g^\mu(x+\delta) &= g(x+\delta) + \alpha y_{i^\star(x+\delta)} \|\nabla g(x+\delta)\|_* \\
&= g(x) + \nabla g(x')^\top \delta + \alpha y_i \|\nabla g(x+\delta)\|_*
\end{aligned}$$

for some $x'$ on the line segment between $x$ and $x+\delta$. This gives that

$$\begin{aligned}
|g^\mu(x+\delta) - g^\mu(x)| &= |g^\mu(x+\delta) - g(x) - \alpha y_i \|\nabla g(x)\|_*| \\
&= \left|\nabla g(x')^\top \delta + \alpha y_i \|\nabla g(x+\delta)\|_* - \alpha y_i \|\nabla g(x)\|_*\right| \\
&= \left|\nabla g(x)^\top \delta + \left(\nabla g(x') - \nabla g(x)\right)^\top \delta + \alpha y_i \left(\|\nabla g(x+\delta)\|_* - \|\nabla g(x)\|_*\right)\right| \\
&\leq \left|\nabla g(x)^\top \delta\right| + \left|\left(\nabla g(x') - \nabla g(x)\right)^\top \delta\right| \\
&\quad + \alpha \left|\|\nabla g(x+\delta)\|_* - \|\nabla g(x)\|_*\right| \\
&\leq \|\nabla g(x)\|_* \|\delta\| + \|\nabla g(x') - \nabla g(x)\|_* \|\delta\| \\
&\quad + \alpha \left|\|\nabla g(x+\delta)\|_* - \|\nabla g(x)\|_*\right| \\
&\leq \|\nabla g(x)\|_* \|\delta\| + L\|x' - x\| \|\delta\| \\
&\quad + \alpha \left|\|\nabla g(x+\delta)\|_* - \|\nabla g(x)\|_*\right| \\
&\leq \|\nabla g(x)\|_* \|\delta\| + L\|\delta\|^2 \\
&\quad + \alpha \left|\|\nabla g(x+\delta)\|_* - \|\nabla g(x)\|_*\right|.
\end{aligned}$$

To bound the last term, note that $-\|\nabla g(x+\delta) - \nabla g(x)\|_* \leq \|\nabla g(x+\delta)\|_* - \|\nabla g(x)\|_* \leq \|\nabla g(x+\delta) - \nabla g(x)\|_*$, so $|\|\nabla g(x+\delta)\|_* - \|\nabla g(x)\|_*| \leq \|\nabla g(x+\delta) - \nabla g(x)\|_* \leq L\|\delta\|$. Thus,

$$|g^\mu(x+\delta) - g^\mu(x)| \leq (\alpha L + \|\nabla g(x)\|_*) \|\delta\| + L\|\delta\|^2.$$

Since $\|\delta\| < r^\mu_{\text{base}}(x)$, this gives that

$$|g^\mu(x+\delta) - g^\mu(x)| \leq |g^\mu(x)|.$$

The remainder of the proof is identical to that of Theorem 17 from (6.5) onwards. □

As in the linear case, the certified radius depends on two terms ($r^\mu_{\text{base}}(x)$ and $r^\mu_{\text{data}}(x)$) that, informally, characterize the local geometry of the base classifier and quantify the distance to the nearest data point of a differing class, respectively. Again, $r^\mu(x) \geq 0$. When $g$ is linear, then $\nabla g$ is constant and is therefore 0-Lipschitz. In this case, Theorem 18 holds for all $L > 0$, and therefore $\lim_{L \downarrow 0} r^\mu_{\text{base}}(x) = \frac{|g^\mu(x)|}{\|\nabla g(x)\|_*}$ implies that $r^\mu(x) = r^\mu_{\text{linear}}(x)$, i.e., the certified radius recovers that of Theorem 17, despite the proof for the nonlinear case involving more bounding steps.

## 6.5 Numerical Simulations

### Illustrative Example

Consider the spiral dataset with test data shown in Figure 6.1 and a support vector machine (SVM) learned on isolated training data. Using the SVM as the base classifier, we apply locally biased randomized smoothing (with an unused subset of training data) with $\alpha \in$

$[0, 10]$, denoted $\alpha$-LBRS. The certified radius from Theorem 17 is computed at every test point using both $\|\cdot\| = \|\cdot\|_2$ and $\|\cdot\| = \|\cdot\|_\infty$. The averages of these radii are denoted by $\ell_2$-avg$(r^\mu(x))$ and $\ell_\infty$-avg$(r^\mu(x))$, respectively. We also compute the average true certified radii, $\ell_2$- and $\ell_\infty$-avg$_{\text{true}}(r^\mu(x))$, which are found by setting the certified radius to zero for test points that are classified incorrectly by $f^\mu$. From the decision region plots in Figure 6.1, we see for $\alpha > 0$ that $\alpha$-LBRS learns to increase the nonlinearity of the base classifier in order to enhance robustness. In contrast, standard smoothing leaves the base SVM classifier unchanged, failing to increase robustness. The average certified radii, the average true certified radii, and the clean accuracy all simultaneously increase upon applying $\alpha$-LBRS (see Figure 6.2). We see that $\alpha$-LBRS converges pointswise to the 1-nearest neighbor (1-NN) as $\alpha \to \infty$, which we recall is a nontrivial consequence of our method. We will see in the next section that this is beneficial even on larger non-synthetic examples.

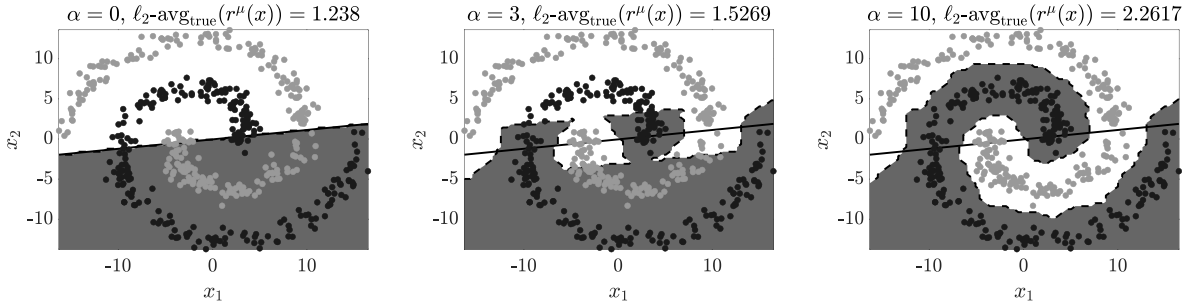



Figure 6.1: Test data, SVM decision boundary (bold line), and $f^\mu$ decision regions (shaded).

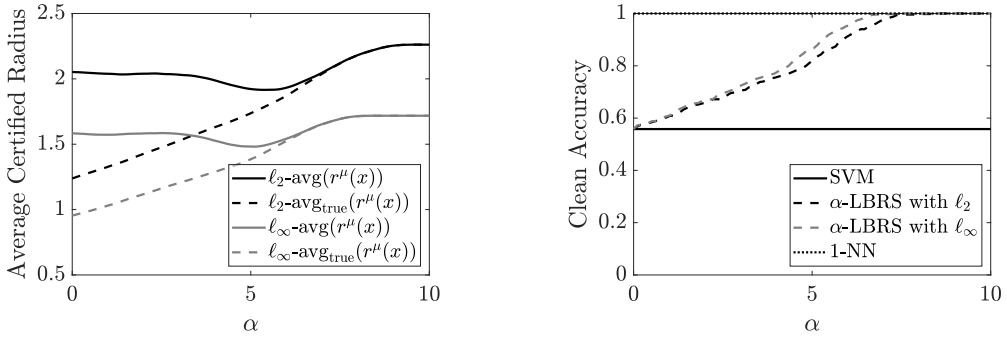



Figure 6.2: Average certified radius and clean accuracy for $\alpha$-LBRS versus $\alpha$.

## Evaluating Clean and Robust Accuracy

The MNIST dataset [84] is considered in a binary setting, where images with digit eight are labeled 1 and the rest are labeled $-1$. The training and testing data are randomly

selected so that the number of data points in class $-1$ equals the number in class 1. Of the training data, $N = 10$ points are reserved for locally biased randomized smoothing. We train a convolutional neural network (CNN) containing three convolutional layers with ReLU activations and one fully connected layer. Using the CNN as the base classifier, we apply normal smoothing [35] with $\sigma \in [0, 0.5]$, denoted $\sigma$-NS, and locally biased randomized smoothing with $\alpha \in [0, 1000]$, denoted $\alpha$-LBRS. We also consider the 1-nearest neighbor (1-NN) using the $N$ reserved training data points.

The accuracy of each model is computed on the test set as well as on an adversarially attacked version of the test set using a 10-step $\ell_2$-PGD attack [95] with attack radius $\epsilon \in \{0.5, 1\}$. The results are shown in Figure 6.3. Although $\sigma$-NS achieves good robustification for small $\sigma$, the accuracy rapidly degrades to that of a constant function (0.5 for this binary problem) as $\sigma$ increases. On the other hand, $\alpha$-LBRS converges (nonlinearly and nontrivially) to the accuracy of the 1-NN as $\alpha \to \infty$. The 1-NN is seen to be robust against the attacks (which may in part be due to the fact that the attacks are designed for the base CNN classifier—a benefit to the defender from using smoothing at test time), and therefore $\alpha$-LBRS inherits this robustness for large enough $\alpha$.

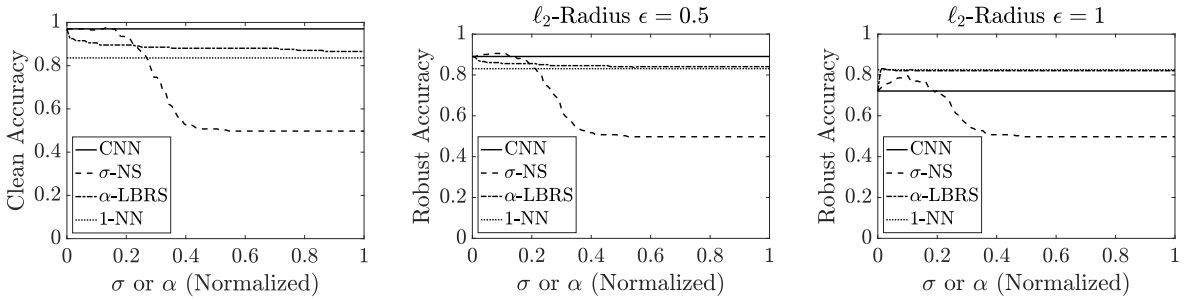



Figure 6.3: Clean and robust accuracy versus smoothing parameter $\sigma$ or $\alpha$.

Next, we fix the parameters $\sigma = 0.05$ and $\alpha = 10$ near the "corners" in Figure 6.3 (recall that the abscissa was normalized) that yield both good clean accuracy and good robust accuracy for $\epsilon \in \{0.5, 1\}$. We attack these models with the wider range of $\ell_2$-radii $\epsilon \in [0, 3]$ and find that $\alpha$-LBRS maintains its resistance to larger attacks for longer than $\sigma$-NS does—the accuracy of $\sigma$-NS degrades at a faster rate—see Figure 6.4. We also demonstrate the generality of our method by considering the same experiment using $\|\cdot\| = \|\cdot\|_\infty$ along with $\ell_\infty$-PGD attacks. Normal smoothing is not catered towards $\ell_\infty$-attacks, which explains the performance increase of $\alpha$-LBRS over $\sigma$-NS relative to the CNN for this attack when compared to the $\ell_2$-attack.

We repeat the simulations on CIFAR-10 [80] and arrive at the same conclusions, albeit with generally lower accuracies and higher sensitivities to attacks. See Appendix 6.A for the quantitative results.

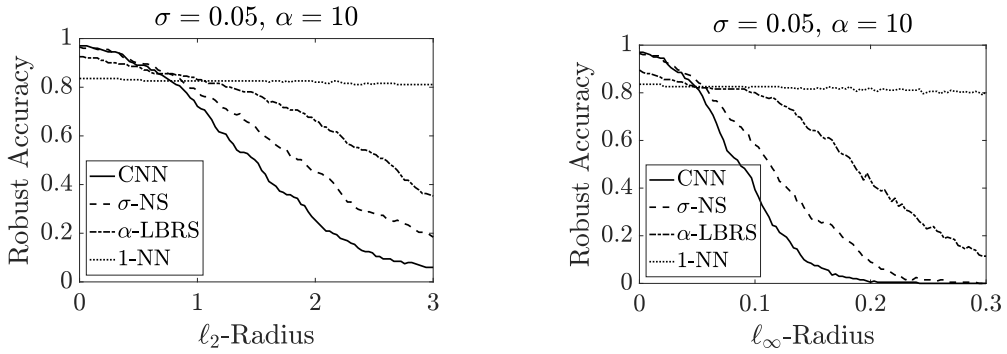



Figure 6.4: Robust accuracy versus attack radius.

## 6.6 Conclusions

In this chapter, conventional randomized smoothing is shown to rely on the idea that smooth decision boundaries are robust, an assumption that manifests into a robustness-accuracy tradeoff. To combat this limitation, locally biased randomized smoothing is introduced to learn locally optimal robustification of a classifier's decision boundary from data. The method directly induces robustness without relying on the surrogate notion of smoothness. Certified robust radii are proved for the binary setting, and simulations show an increased certified, clean, and robust accuracy over conventional smoothing. Possible future directions include a multiclass extension, studying alternate approximate direction oracles, and reducing the memory requirement of storing data at test time.

# Appendices

## 6.A Additional Numerical Simulations

We re-run the simulations of Section 6.5 to evaluate the clean and robust accuracy of our locally biased randomized smoothing method on the CIFAR-10 dataset [80]. The results are shown in Figure 6.5 and Figure 6.6 below. All of the conclusions remain the same as in Section 6.5.

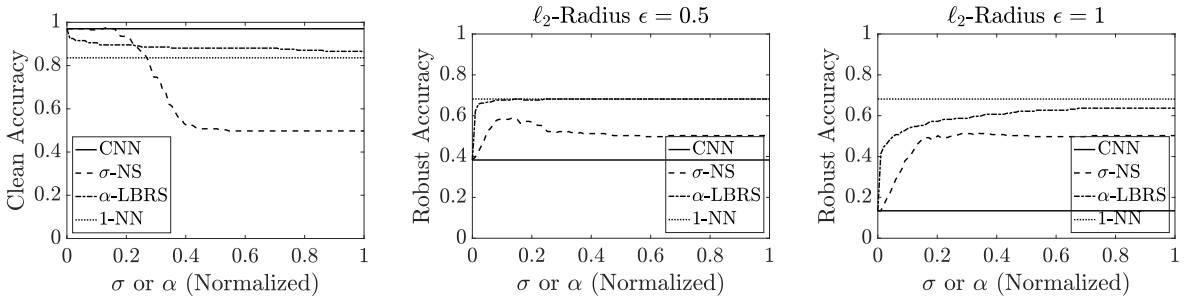



Figure 6.5: Clean and robust accuracy versus smoothing parameter $\sigma$ or $\alpha$.

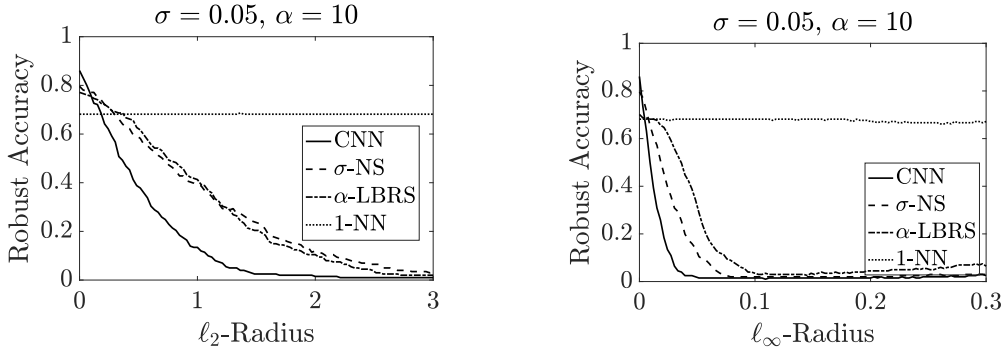



Figure 6.6: Robust accuracy versus attack radius.

# Bibliography


[1] Motasem Alfarra, Adel Bibi, Philip H.S. Torr, and Bernard Ghanem. "Data dependent randomized smoothing". In: *arXiv preprint arXiv:2012.04351* (2020).

[2] Charalambos D. Aliprantis, David Harris, and Rabee Tourky. "Riesz estimators". In: *Journal of Econometrics* (2007).

[3] Brandon Amos, Lei Xu, and Zico Kolter. "Input convex neural networks". In: *International Conference on Machine Learning*. 2017.

[4] Brendon G. Anderson, Ziye Ma, Jingqi Li, and Somayeh Sojoudi. "Tightened convex relaxations for neural network robustness certification". In: *IEEE Conference on Decision and Control*. 2020.

[5] Brendon G. Anderson, Ziye Ma, Jingqi Li, and Somayeh Sojoudi. "Towards optimal branching of linear and semidefinite relaxations for neural network robustness certification". In: *arXiv preprint arXiv:2101.09306* (2023).

[6] Brendon G. Anderson, Samuel Pfrommer, and Somayeh Sojoudi. "Tight certified robustness via min-max representations of ReLU neural networks". In: *IEEE Conference on Decision and Control (CDC)*. 2023.

[7] Brendon G. Anderson, Samuel Pfrommer, and Somayeh Sojoudi. "Towards optimal randomized smoothing: A semi-infinite linear programming approach". In: *ICML Workshop on Formal Verification of Machine Learning*. 2022.

[8] Brendon G. Anderson and Somayeh Sojoudi. "Certified robustness via locally biased randomized smoothing". In: *Learning for Dynamics and Control*. 2022.

[9] Brendon G. Anderson and Somayeh Sojoudi. "Data-driven certification of neural networks with random input noise". In: *IEEE Transactions on Control of Network Systems* (2022).

[10] MOSEK ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 9.0.* 2019. URL: `http://docs.mosek.com/9.0/toolbox/index.html`.

[11] Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. "Understanding deep neural networks with rectified linear units". In: *International Conference on Learning Representations*. 2018.

[12] Karl J. Åström. *Introduction to Stochastic Control Theory*. Courier Corporation, 2012.

[13] Pranjal Awasthi, Abhratanu Dutta, and Aravindan Vijayaraghavan. "On robustness to adversarial examples and polynomial optimization". In: *Advances in Neural Information Processing Systems*. 2019.

[14] Adil M. Bagirov. "Max-min separability". In: *Optimization Methods and Software* (2005).

[15] Adil M. Bagirov and Julien Ugon. "Supervised data classification via max-min separability". In: *Continuous Optimization: Current Trends and Modern Applications* (2005).

[16] Yatong Bai, Brendon G. Anderson, Aerin Kim, and Somayeh Sojoudi. "Improving the accuracy-robustness trade-off of classifiers via adaptive smoothing". In: *SIAM Journal on Mathematics of Data Science* (2024).

[17] Yatong Bai, Brendon G. Anderson, and Somayeh Sojoudi. "Mixing classifiers to alleviate the accuracy-robustness trade-off". In: *Learning for Dynamics and Control*. 2024.

[18] Stanley Bak, Changliu Liu, and Taylor Johnson. "The second international verification of neural networks competition (VNN-COMP 2021): Summary and results". In: *arXiv preprint arXiv:2109.00498* (2021).

[19] Randall Balestriero, Jerome Pesenti, and Yann LeCun. "Learning in high dimension always amounts to extrapolation". In: *arXiv preprint arXiv:2110.09485* (2021).

[20] Mislav Balunovic, Maximilian Baader, Gagandeep Singh, Timon Gehr, and Martin Vechev. "Certifying geometric robustness of neural networks". In: *Advances in Neural Information Processing Systems*. 2019.

[21] Ben Batten, Panagiotis Kouvaros, Alessio Lomuscio, and Yang Zheng. "Efficient neural network verification via layer-based semidefinite relaxations and linear cuts". In: *International Joint Conference on Artificial Intelligence*. 2021.

[22] Dimitri P. Bertsekas. *Nonlinear Programming*. 3rd. Athena Scientific, 2016.

[23] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. "End to end learning for self-driving cars". In: *arXiv preprint arXiv:1604.07316* (2016).

[24] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[25] Rudy Bunel, P. Mudigonda, Ilker Turkaslan, P Torr, Jingyue Lu, and Pushmeet Kohli. "Branch and bound for piecewise linear neural network verification". In: *Journal of Machine Learning Research* (2020).

[26] Marco C. Campi, Simone Garatti, and Maria Prandini. "The scenario approach for systems and control design". In: *Annual Reviews in Control* (2009).

[27] Marco Claudio Campi, Simone Garatti, and Federico Alessandro Ramponi. "A general scenario theory for nonconvex optimization and decision making". In: *IEEE Transactions on Automatic Control* (2018).

[28] Nicholas Carlini and David Wagner. "Towards evaluating the robustness of neural networks". In: *IEEE Symposium on Security and Privacy*. 2017.

[29] Francesco Cartella, Orlando Anunciacao, Yuki Funabiki, Daisuke Yamaguchi, Toru Akishita, and Olivier Elshocht. "Adversarial attacks for tabular data: Application to fraud detection and imbalanced data". In: *arXiv preprint arXiv:2101.08030* (2021).

[30] Chen Chen, Kezhi Kong, Peihong Yu, Juan Luque, Tom Goldstein, and Furong Huang. "Insta-RS: Instance-wise randomized smoothing for improved robustness and accuracy". In: *arXiv preprint arXiv:2103.04436* (2021).

[31] Sitan Chen, Adam R. Klivans, and Raghu Meka. "Learning deep ReLU networks is fixed-parameter tractable". In: *arXiv preprint arXiv:2009.13512* (2020).

[32] Tong Chen, Jean B. Lasserre, Victor Magron, and Edouard Pauwels. "Semialgebraic optimization for Lipschitz constants of relu networks". In: *Advances in Neural Information Processing Systems* (2020).

[33] Yize Chen, Yuanyuan Shi, and Baosen Zhang. "Data-driven optimal voltage regulation using input convex neural networks". In: *Electric Power Systems Research* (2020).

[34] Yize Chen, Yuanyuan Shi, and Baosen Zhang. "Optimal control via neural networks: A convex approach". In: *International Conference on Learning Representations*. 2019.

[35] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. "Certified adversarial robustness via randomized smoothing". In: *International Conference on Machine Learning*. 2019.

[36] Nicolas Couellan. "Probabilistic robustness estimates for feed-forward neural networks". In: *Neural Networks* (2021).

[37] Stefano Cresci, Marinella Petrocchi, Angelo Spognardi, and Stefano Tognazzi. "Adversarial machine learning for protecting against online manipulation". In: *Internet Computing* (2021).

[38] Nilesh Dalvi, Pedro Domingos, Sumit Sanghai, and Deepak Verma. "Adversarial classification". In: *10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2004.

[39] Alessandro De Palma, Rudy Bunel, Alban Desmaison, Krishnamurthy Dvijotham, Pushmeet Kohli, Philip H.S. Torr, and M. Pawan Kumar. "Improved branch and bound for neural network verification via lagrangian decomposition". In: *arXiv preprint arXiv:2104.06718* (2021).

[40] Alex Devonport and Murat Arcak. "Estimating reachable sets with scenario optimization". In: *Learning for Dynamics and Control*. 2020.

[41] Steven Diamond and Stephen Boyd. "CVXPY: A Python-embedded modeling language for convex optimization". In: *Journal of Machine Learning Research* (2016).

[42] Dheeru Dua and Casey Graff. *UCI Machine Learning Repository*. 2017. URL: `https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%5C%28Diagnostic%5C%29`.

[43] Krishnamurthy Dvijotham, Marta Garnelo, Alhussein Fawzi, and Pushmeet Kohli. "Verification of deep probabilistic models". In: *Advances in Neural Information Processing Systems, SecML workshop*. 2018.

[44] Krishnamurthy Dvijotham, Jamie Hayes, Borja Balle, Zico Kolter, Chongli Qin, Andras Gyorgy, Kai Xiao, Sven Gowal, and Pushmeet Kohli. "A framework for robustness certification of smoothed classifiers using $f$-divergences". In: *International Conference on Learning Representations*. 2020.

[45] Ruediger Ehlers. "Formal verification of piece-wise linear feed-forward neural networks". In: *International Symposium on Automated Technology for Verification and Analysis*. Springer. 2017.

[46] Francisco Eiras, Motasem Alfarra, M. Pawan Kumar, Philip H.S. Torr, Puneet K. Dokania, Bernard Ghanem, and Adel Bibi. "ANCER: Anisotropic certification via sample-wise volume maximization". In: *arXiv preprint arXiv:2107.04570* (2021).

[47] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, and Pascal Frossard. "Robustness of classifiers: From adversarial to random noise". In: *Advances in Neural Information Processing Systems*. 2016.

[48] Alhussein Fawzi, Seyed-Mohsen Moosavi-Dezfooli, Pascal Frossard, and Stefano Soatto. "Empirical study of the topology and geometry of deep networks". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2018.

[49] Mahyar Fazlyab, Manfred Morari, and George J. Pappas. "Probabilistic verification and reachability analysis of neural networks via semidefinite programming". In: *IEEE Conference on Decision and Control (CDC)*. 2019.

[50] Mahyar Fazlyab, Manfred Morari, and George J. Pappas. "Safety verification and robustness analysis of neural networks via quadratic constraints and semidefinite programming". In: *IEEE Transactions on Automatic Control* (2020).

[51] Mahyar Fazlyab, Alexander Robey, Hamed Hassani, Manfred Morari, and George Pappas. "Efficient and accurate estimation of Lipschitz constants for deep neural networks". In: *Advances in Neural Information Processing Systems*. 2019.

[52] Anthony V. Fiacco and Jerzy Kyparisis. "Convexity and concavity properties of the optimal value function in parametric nonlinear programming". In: *Journal of Optimization Theory and Applications* (1986).

[53] Samuel G. Finlayson, John D. Bowers, Joichi Ito, Jonathan L. Zittrain, Andrew L. Beam, and Isaac S. Kohane. "Adversarial attacks on medical machine learning". In: *Science* (2019).

[54] William Fleshman, Edward Raff, Jared Sylvester, Steven Forsyth, and Mark McLean. "Non-negative networks against adversarial attacks". In: *arXiv preprint arXiv: 1806.06108* (2018).

[55] Gerald B. Folland. *Real Analysis: Modern Techniques and Their Applications*. Vol. 40. John Wiley & Sons, 1999.

[56] Hans Föllmer and Alexander Schied. *Stochastic Finance*. de Gruyter, 2016.

[57] Jean-Yves Franceschi, Alhussein Fawzi, and Omar Fawzi. "Robustness of classifiers to uniform $\ell_p$ and Gaussian noise". In: *21st International Conference on Artificial Intelligence and Statistics*. 2018.

[58] Mario Luca Fravolini, Tansel Yucelen, Antonio Ficola, and Marcello Rosario Napolitano. "Probabilistic estimation of the reachable set of model reference adaptive controllers using the scenario approach". In: *International Journal of Control* (2017).

[59] Fernando Gama and Somayeh Sojoudi. "Graph neural networks for distributed linear-quadratic control". In: *Learning for Dynamics and Control*. 2021.

[60] Yue Gao, Harrison Rosenberg, Kassem Fawaz, Somesh Jha, and Justin Hsu. "Analyzing accuracy loss in randomized smoothing defenses". In: *arXiv preprint arXiv: 2003.01595* (2020).

[61] Felipe O. Giuste and Juan C. Vizcarra. "CIFAR-10 image classification using feature ensembles". In: *arXiv preprint arXiv:2002.03846* (2020).

[62] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. "Explaining and harnessing adversarial examples". In: *International Conference on Learning Representations*. 2015.

[63] Edita Grolman, Hodaya Binyamini, Asaf Shabtai, Yuval Elovici, Ikuya Morikawa, and Toshiya Shimizu. "HateVersarial: Adversarial attack against hate speech detection algorithms on Twitter". In: *30th ACM Conference on User Modeling, Adaptation and Personalization*. 2022.

[64] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. "Adversarial examples for malware detection". In: *European Symposium on Research in Computer Security*. Springer. 2017.

[65] Matthias Hein and Maksym Andriushchenko. "Formal guarantees on the robustness of a classifier against adversarial manipulation". In: *Advances in Neural Information Processing Systems*. 2017.

[66] Dorit S. Hochbaum. "Approximating covering and packing problems: Set cover, vertex cover, independent set, and related problems". In: *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Co., 1996, pp. 94–143.

[67] Judy Hoffman, Daniel A. Roberts, and Sho Yaida. "Robust learning with Jacobian regularization". In: *arXiv preprint arXiv:1908.02729* (2019).

[68] Chengqiang Huang, Zheng Hu, Xiaowei Huang, and Ke Pei. "Statistical certification of acceptable robustness for neural networks". In: *International Conference on Artificial Neural Networks*. Springer. 2021.

[69] Íñigo Íncer Romeo, Michael Theodorides, Sadia Afroz, and David Wagner. "Adversarially robust malware detection using monotonic classification". In: *4th ACM International Workshop on Security and Privacy Analytics*. 2018.

[70] ISO/IEC. "Guide 51: Safety Aspects—Guidelines for their Inclusion in Standards". In: (1999).

[71] Ming Jin, Javad Lavaei, Somayeh Sojoudi, and Ross Baldick. "Boundary defense against cyber threat for power system state estimation". In: *Transactions on Information Forensics and Security* (2020).

[72] Matt Jordan and Alexandros G. Dimakis. "Exactly computing the local Lipschitz constant of ReLU networks". In: *Advances in Neural Information Processing Systems*. 2020.

[73] Guy Katz, Clark Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. "Reluplex: An efficient SMT solver for verifying deep neural networks". In: *Computer Aided Verification: 29th International Conference*. 2017.

[74] Mohammed Kayed, Ahmed Anter, and Hadeer Mohamed. "Classification of garments from fashion MNIST dataset using CNN LeNet-5 architecture". In: *International Conference on Innovative Trends in Communication and Computer Engineering*. 2020.

[75] Leonid Khachiyan, Endre Boros, Konrad Borys, Vladimir Gurvich, and Khaled Elbassioni. "Generating all vertices of a polyhedron is hard". In: *Discrete & Computational Geometry* (2009).

[76] Jinrae Kim and Youdan Kim. "Parameterized convex universal approximators for decision-making problems". In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[77] Diederik P. Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *International Conference on Learning Representations*. 2015.

[78] Weicong Kong, Zhao Yang Dong, Youwei Jia, David J. Hill, Yan Xu, and Yuan Zhang. "Short-term residential load forecasting based on LSTM recurrent neural network". In: *IEEE Transactions on Smart Grid* (2017).

[79] Vishaal Krishnan, Abed AlRahman Al Makdah, and Fabio Pasqualetti. "Lipschitz bounds and provably robust training by Laplacian smoothing". In: *Advances in Neural Information Processing Systems*. 2020.

[80] Alex Krizhevsky and Geoffrey Hinton. "Learning multiple layers of features from tiny images". In: *Technical report* (2009).

[81] Aounon Kumar, Alexander Levine, and Soheil Feizi. "Policy smoothing for provably robust reinforcement learning". In: *arXiv preprint arXiv:2106.11420* (2021).

[82] Aounon Kumar, Alexander Levine, Soheil Feizi, and Tom Goldstein. "Certifying confidence via randomized smoothing". In: *Advances in Neural Information Processing Systems*. 2020.

[83] K. Naveen Kumar, C. Vishnu, Reshmi Mitra, and C. Krishna Mohan. "Black-box adversarial attacks in autonomous vehicle technology". In: *IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*. 2020.

[84] Yann LeCun. *The MNIST database of handwritten digits*. `http://yann.lecun.com/exdb/mnist/`. 1998.

[85] Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. "Certified robustness to adversarial examples with differential privacy". In: *Symposium on Security and Privacy*. IEEE. 2019.

[86] Alexander Levine and Soheil Feizi. "Wasserstein smoothing: Certified robustness against Wasserstein adversarial attacks". In: *International Conference on Artificial Intelligence and Statistics*. 2020.

[87] Alexander J. Levine and Soheil Feizi. "Improved, deterministic smoothing for $\ell_1$ certified robustness". In: *International Conference on Machine Learning*. 2021.

[88] Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. "Certified adversarial robustness with additive noise". In: *Advances in Neural Information Processing Systems*. 2019.

[89] Renjue Li, Pengfei Yang, Cheng-Chao Huang, Bai Xue, and Lijun Zhang. "Probabilistic robustness analysis for DNNs based on PAC Learning". In: *arXiv preprint arXiv:2101.10102* (2021).

[90] Qun Liu and Supratik Mukhopadhyay. "Unsupervised learning using pretrained CNN and associative memory bank". In: *International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2018.

[91] Daniel Lowd and Christopher Meek. "Good word attacks on statistical spam filters". In: *CEAS*. 2005.

[92] Jingyue Lu and M. Pawan Kumar. "Neural network branching for neural network verification". In: *arXiv preprint arXiv:1912.01329* (2019).

[93] James Luedtke and Shabbir Ahmed. "A sample approximation approach for optimization with probabilistic constraints". In: *SIAM Journal on Optimization* (2008).

[94] Ziye Ma and Somayeh Sojoudi. "A sequential framework towards an exact SDP verification of neural networks". In: *8th International Conference on Data Science and Advanced Analytics (DSAA)*. IEEE. 2021.

[95] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. "Towards deep learning models resistant to adversarial attacks". In: *International Conference on Learning Representations*. 2018.

[96] Ashok Makkuva, Amirhossein Taghvaei, Sewoong Oh, and Jason Lee. "Optimal transport mapping via input convex neural networks". In: *International Conference on Machine Learning*. 2020.

[97] Ravi Mangal, Aditya V. Nori, and Alessandro Orso. "Robustness of neural networks: A probabilistic and practical approach". In: *IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*. 2019.

[98] Olvi L Mangasarian. "Arbitrary-norm separating plane". In: *Operations Research Letters* (1999).

[99] Jean-Luc Marichal. "Weighted lattice polynomials". In: *Discrete Mathematics* (2009).

[100] Jeet Mohapatra, Ching-Yun Ko, Lily Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. "Hidden cost of randomized smoothing". In: *International Conference on Artificial Intelligence and Statistics*. 2021.

[101] Guido F. Montufar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. "On the number of linear regions of deep neural networks". In: *Advances in Neural Information Processing Systems*. 2014.

[102] Mark Niklas Müller, Christopher Brix, Stanley Bak, Changliu Liu, and Taylor T Johnson. "The Third International Verification of Neural Networks Competition (VNN-COMP 2022): Summary and Results". In: *arXiv preprint arXiv:2212.10376* (2022).

[103] Mark Niklas Müller, Gleb Makarchuk, Gagandeep Singh, Markus Püschel, and Martin Vechev. "PRIMA: Precise and general neural network certification via multi-neuron convex relaxations". In: *arXiv preprint arXiv:2103.03638* (2021).

[104] K. Muralitharan, Rathinasamy Sakthivel, and R. Vishnuvarthan. "Neural network based optimization approach for energy demand prediction in smart grid". In: *Neurocomputing* (2018).

[105] Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. "Deep double descent: Where bigger models and more data hurt". In: *Journal of Statistical Mechanics: Theory and Experiment* (2021).

[106] Lakshmanan Nataraj, Sreejith Karthikeyan, Gregoire Jacob, and Bangalore S. Manjunath. "Malware images: Visualization and automatic classification". In: *8th International Symposium on Visualization for Cyber Security*. 2011.

[107] Arkadi Nemirovski and Alexander Shapiro. "Convex approximations of chance constrained programs". In: *SIAM Journal on Optimization* (2007).

[108] Vitali Nesterov, Fabricio Arend Torres, Monika Nagy-Huber, Maxim Samarin, and Volker Roth. "Learning invariances with generalised input-convex neural networks". In: *arXiv preprint arXiv:2204.07009* (2022).

[109] Sergei Ovchinnikov. "Max-min representation of piecewise linear functions". In: *Contributions to Algebra and Geometry* (2002).

[110] Xiang Pan, Tianyu Zhao, and Minghua Chen. "DeepOPF: Deep neural network for DC optimal power flow". In: *IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. 2019.

[111] Samuel Pfrommer, Brendon G. Anderson, and Somayeh Sojoudi. "Projected randomized smoothing for certified adversarial robustness". In: *Transactions on Machine Learning Research* (2023).

[112] Samuel Pfrommer, Brendon G. Anderson, and Somayeh Sojoudi. "Projected randomized smoothing for certified adversarial robustness". In: *Transactions on Machine Learning Research* (2023).

[113] Samuel Pfrommer*, Brendon G. Anderson*, Julien Piet, and Somayeh Sojoudi. "Asymmetric certified robustness via feature-convex neural networks". In: *Advances in Neural Information Processing Systems*. *Co-first author and equal contribution. 2023.

[114] Tien Ho-Phuoc. "CIFAR-10 to compare visual recognition performance between deep neural networks and humans". In: *arXiv preprint arXiv:1811.07270* (2018).

[115] Allan Pinkus. "Approximation theory of the MLP model in neural networks". In: *Acta Numerica* (1999).

[116] Aditi Raghunathan, Jacob Steinhardt, and Percy S Liang. "Semidefinite relaxations for certifying robustness to adversarial examples". In: *Advances in Neural Information Processing Systems*. 2018.

[117] Kai Ren, Heejin Ahn, and Maryam Kamgarpour. "Chance-constrained trajectory planning with multimodal environmental uncertainty". In: *IEEE Control Systems Letters* (2022).

[118] Blaine Rister and Daniel L. Rubin. "Piecewise convexity of artificial neural networks". In: *Neural Networks* (2017).

[119] R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[120] Vicenc Rubies Royo, Roberto Calandra, Dusan M. Stipanovic, and Claire Tomlin. "Fast neural network verification via shadow prices". In: *arXiv preprint arXiv: 1902.07247* (2019).

[121] Amir Mahdi Sadeghzadeh, Saeed Shiravi, and Rasool Jalili. "Adversarial network traffic: Towards evaluating the robustness of deep-learning-based network traffic classification". In: *IEEE Transactions on Network and Service Management* (2021).

[122] Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. "Provably robust deep learning via adversarially trained smoothed classifiers". In: *Advances in Neural Information Processing Systems*. 2019.

[123] Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. "A convex relaxation barrier to tight robustness verification of neural networks". In: *Advances in Neural Information Processing Systems* (2019).

[124] Suman Sapkota and Binod Bhattarai. "Input invex neural network". In: *arXiv preprint arXiv:2106.08748* (2021).

[125] Hossein Sartipizadeh, Abraham P. Vinod, Behçet Açikmeşe, and Meeko Oishi. "Voronoi partition-based scenario reduction for fast sampling-based stochastic reachability computation of linear systems". In: *American Control Conference (ACC)*. 2019.

[126] Ali Siahkamari, Durmus Alp Emre Acar, Christopher Liao, Kelly L. Geyer, Venkatesh Saligrama, and Brian Kulis. "Faster algorithms for learning convex functions". In: *International Conference on Machine Learning*. 2022.

[127] Sarath Sivaprasad, Ankur Singh, Naresh Manwani, and Vineet Gandhi. "The curious case of convex neural networks". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2021.

[128] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. "One pixel attack for fooling deep neural networks". In: *Transactions on Evolutionary Computation* (2019).

[129] Peter Súkeník, Aleksei Kuvshinov, and Stephan Günnemann. "Intriguing properties of input-dependent randomized smoothing". In: *arXiv preprint arXiv:2110.05365* (2021).

[130] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. "Intriguing properties of neural networks". In: *International Conference on Learning Representations*. 2014.

[131] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. "Intriguing properties of neural networks". In: *International Conference on Learning Representations*. 2014.

[132] Roberto Tempo, Giuseppe Calafiore, and Fabrizio Dabbene. *Randomized Algorithms for Analysis and Control of Uncertain Systems*. Springer Science & Business Media, 2012.

[133] Jiaye Teng, Guang-He Lee, and Yang Yuan. "$\ell_1$ adversarial robustness certificates: A randomized smoothing approach". In: *Preprint* (2020). URL: `https://openreview.net/forum?id=H1lQIgrFDS`.

[134] Christian Tjandraatmadja, Ross Anderson, Joey Huchette, Will Ma, Krunal Kishor Patel, and Juan Pablo Vielma. "The convex relaxation barrier, revisited: Tightened single-neuron relaxations for neural network verification". In: *Advances in Neural Information Processing Systems* (2020).

[135] Vincent Tjeng, Kai Xiao, and Russ Tedrake. "Evaluating robustness of neural networks with mixed integer programming". In: *International Conference on Learning Representations*. 2019.

[136] Asher Trockman and Zico Kolter. "Orthogonalizing convolutional layers with the Cayley transform". In: *arXiv preprint arXiv:2104.07167* (2021).

[137] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. "Robustness May Be at Odds with Accuracy". In: *International Conference on Learning Representations*. 2019.

[138] Santiago Velasco-Forero and Jesús Angulo. "MorphoActivation: Generalizing ReLU activation function by mathematical morphology". In: *Discrete Geometry and Mathematical Morphology*. 2022.

[139] Aladin Virmaux and Kevin Scaman. "Lipschitz regularity of deep neural networks: Analysis and efficient estimation". In: *Advances in Neural Information Processing Systems*. 2018.

[140] Allen Wang, Ashkan Jasour, and Brian Williams. "Non-Gaussian chance-constrained trajectory planning for autonomous vehicles under agent uncertainty". In: *IEEE Robotics and Automation Letters* (2020).

[141] Lei Wang, Runtian Zhai, Di He, Liwei Wang, and Li Jian. "Pretrain-to-finetune adversarial training via sample-wise randomized smoothing". In: *Preprint* (2021). URL: `https://openreview.net/pdf?id=Te1aZ2myPIu`.

[142] Shiqi Wang, Kexin Pei, Justin Whitehouse, Junfeng Yang, and Suman Jana. "Efficient formal safety analysis of neural networks". In: *Advances in Neural Information Processing Systems* (2018).

[143] Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and Zico Kolter. "Beta-CROWN: Efficient bound propagation with per-neuron split constraints for neural network robustness verification". In: *Advances in Neural Information Processing Systems*. 2021.

[144] Shuning Wang and Kumpati S. Narendra. "Nonlinear system identification with lattice piecewise-linear functions". In: *American Control Conference*. 2002.

[145] Yizhen Wang, Somesh Jha, and Kamalika Chaudhuri. "Analyzing the robustness of nearest neighbors to adversarial examples". In: *International Conference on Machine Learning*. 2018.

[146] Stefan Webb, Tom Rainforth, Yee Whye Teh, and M. Pawan Kumar. "A statistical approach to assessing neural network robustness". In: *International Conference on Learning Representations*. 2019.

[147] Lily Weng, Pin-Yu Chen, Lam Nguyen, Mark Squillante, Akhilan Boopathy, Ivan Oseledets, and Luca Daniel. "PROVEN: Verifying robustness of neural networks with a probabilistic approach". In: *International Conference on Machine Learning*. 2019.

[148] Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. "Towards fast computation of certified robustness for ReLU networks". In: *International Conference on Machine Learning*. 2018.

[149] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. "Evaluating the robustness of neural networks: An extreme value theory approach". In: *International Conference on Learning Representations*. 2018.

[150] Eric Wong and Zico Kolter. "Provable defenses against adversarial examples via the convex outer adversarial polytope". In: *International Conference on Machine Learning*. 2018.

[151] Bichen Wu, Forrest Iandola, Peter H. Jin, and Kurt Keutzer. "SqueezeDet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

[152] Fan Wu, Linyi Li, Zijian Huang, Yevgeniy Vorobeychik, Ding Zhao, and Bo Li. "CROP: Certifying robust policies for reinforcement learning through functional smoothing". In: *arXiv preprint arXiv:2106.09292* (2021).

[153] Haoze Wu, Alex Ozdemir, Aleksandar Zeljic, Kyle Julian, Ahmed Irfan, Divya Gopinath, Sadjad Fouladi, Guy Katz, Corina Pasareanu, and Clark Barrett. "Parallelization techniques for verifying neural networks". In: *20th International Conference on Formal Methods In Computer-Aided Design*. 2020.

[154] Weiming Xiang and Taylor T. Johnson. "Reachability analysis and safety verification for neural network control systems". In: *arXiv preprint arXiv:1805.09944* (2018).

[155] Weiming Xiang, Hoang-Dung Tran, and Taylor T. Johnson. "Output reachable set estimation and verification for multilayer neural networks". In: *IEEE Transactions on Neural Networks and Learning Systems* (2018).

[156] Han Xiao, Kashif Rasul, and Roland Vollgraf. "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms". In: *arXiv preprint arXiv:1708.07747* (2017).

[157] Kaidi Xu, Huan Zhang, Shiqi Wang, Yihan Wang, Suman Jana, Xue Lin, and Cho-Jui Hsieh. "Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers". In: *arXiv preprint arXiv:2011.13824* (2020).

[158] Greg Yang, Tony Duan, J. Edward Hu, Hadi Salman, Ilya Razenshteyn, and Jerry Li. "Randomized smoothing of all shapes and sizes". In: *International Conference on Machine Learning*. 2020.

[159] Yang Yang, Jun Zhang, Kai-Quan Cai, and Maria Prandini. "A stochastic reachability analysis approach to aircraft conflict detection and resolution". In: *IEEE Conference on Control Applications (CCA)*. 2014.

[160] Yao-Yuan Yang, Cyrus Rashtchian, Hongyang Zhang, Russ R. Salakhutdinov, and Kamalika Chaudhuri. "A closer look at accuracy vs. robustness". In: *Advances in Neural Information Processing Systems* (2020).

[161] Roozbeh Yousefzadeh. "Deep learning generalization and the convex hull of training sets". In: *NeurIPS Deep Learning through Information Workshop*. 2020.

[162] Radoslaw R. Zakrzewski. "Randomized approach to verification of neural networks". In: *IEEE International Joint Conference on Neural Networks*. 2004.

[163] Fancheng Zeng, Guanqiu Qi, Zhiqin Zhu, Jian Sun, Gang Hu, and Matthew Haner. "Convex neural networks based reinforcement learning for load frequency control under denial of service attacks". In: *Algorithms* (2022).

[164] Runtian Zhai, Chen Dan, Di He, Huan Zhang, Boqing Gong, Pradeep Ravikumar, Cho-Jui Hsieh, and Liwei Wang. "MACER: Attack-free and scalable robust training via maximizing certified radius". In: *International Conference on Learning Representations*. 2020.

[165] Bohang Zhang, Du Jiang, Di He, and Liwei Wang. "Boosting the certified robustness of l-infinity distance nets". In: *arXiv preprint arXiv:2110.06850* (2021).

[166] Dinghuai Zhang, Mao Ye, Chengyue Gong, Zhanxing Zhu, and Qiang Liu. "Black-box certification with randomized smoothing: A functional optimization based framework". In: *Advances in Neural Information Processing Systems*. 2020.

[167] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. "Theoretically principled trade-off between robustness and accuracy". In: *International Conference on Machine Learning*. 2019.

[168] Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. "Efficient neural network robustness certification with general activation functions". In: *Advances in Neural Information Processing Systems*. 2018.

[169] Ling Zhang, Yize Chen, and Baosen Zhang. "A convex neural network solver for DCOPF with generalization guarantees". In: *IEEE Transactions on Control of Network Systems* (2021).

[170] Liwen Zhang, Gregory Naitzat, and Lek-Heng Lim. "Tropical geometry of deep neural networks". In: *International Conference on Machine Learning*. 2018.

[171] Richard Zhang. "On the tightness of semidefinite relaxations for certifying robustness to adversarial examples". In: *Advances in Neural Information Processing Systems*. 2020.

[172] Sen Zhao, Erez Louidor, and Maya Gupta. "Global optimization networks". In: *International Conference on Machine Learning*. 2022.

[173] Jianzhe Zhen, Daniel Kuhn, and Wolfram Wiesemann. "Mathematical foundations of robust and distributionally robust optimization". In: *arXiv preprint arXiv:2105.00760* (2021).

[174] Andy Zou, Zifan Wang, Zico Kolter, and Matt Fredrikson. "Universal and transferable adversarial attacks on aligned language models". In: *arXiv preprint arXiv:2307.15043* (2023).