Certified Adversarial Robustness With Domain Constraints

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

Ing. Václav Voráček

aus Brno / Tschechien

Tübingen

2025

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	24.01.2025
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter/-in:	Prof. Dr. Matthias Hein
2. Berichterstatter/-in:	Prof. Dr. Ulrike von Luxburg
3. Berichterstatter/-in:	Assoc. Prof. Dr. Francesco Orabona

Contents

1 Summary **1**

2 Introduction **5**
2.1 Adversarial robustness . . . 6
2.1.1 Empirical robustness . . . 7
2.1.2 Certified robustness . . . 8
2.2 Objectives and Outcomes . . . 10
2.2.1 List of publications . . . 11
2.3 Thesis organization . . . 11
2.4 Notation . . . 12

3 Provably Adversarially Robust Nearest Prototype Classifiers **13**
3.1 Backstory . . . 13
3.2 Introduction . . . 13
3.3 Overview of results . . . 14
3.4 Conclusion . . . 18

4 Sound Randomized Smoothing in Floating-Point Accuracy **19**
4.1 Backstory . . . 19
4.2 Introduction . . . 19
4.3 Overview of results . . . 20
4.4 Conclusion . . . 22

5 Improving ℓ_1 Certified Robustness via Randomized Smoothing By Leveraging Box Constraints **23**
5.1 Backstory . . . 23
5.2 Introduction . . . 23
5.3 Overview of results . . . 24
5.4 Conclusion . . . 26

6 Treatment of Statistical Estimation Problems in Randomized Smoothing **27**
6.1 Backstory . . . 27
6.2 Introduction . . . 27
6.3 Overview of results . . . 28
6.4 Conclusion . . . 32

7 Full papers **37**
7.1 Provably Adversarially Robust Nearest Prototype Classifiers 37
7.2 Sound Randomized Smoothing in Floating-Point Accuracy 61
7.3 Improving ℓ_1 Certified Robustness via Randomized Smoothing By Leveraging Box Constraints 84
7.4 Treatment of Statistical Estimation Problems in Randomized Smoothing 110

1 Summary

EN: Deep learning has become a dominant technique in machine learning and with only a little exaggeration it became a synonym to machine learning itself. Despite its great performance on tasks ranging from image classification to text generation, the underlying mechanism remains largely not understood and the deep-learning systems are a black-box, yielding some undesirable properties, such as the presence of adversarial examples. An adversarial example is a tiny modification of an input (usually demonstrated for images) that is imperceivable to humans and does not change the semantics of the input, however, the classifier is fooled and changes output to some absurd value. This is of a major concern in safety-critical applications, such as for autonomous driving where the consequence of such adversarial manipulations are potentially catastrophic. In this thesis, we continue in the effort to mitigate the problem.

- In the first publication, we observe that the problem is not only present for deep learning, but also for simpler classifiers, such as nearest prototype classifiers. In that case, we derive rigorous mathematical guarantees about the robustness and provide tractable lower-bounds for the robustness. Despite using simpler models, this allowed us to establish state-of-the-art results on a popular benchmark.
- Later, we focus on randomized smoothing, which is a method certifying the robustness of a classifier to adversarial perturbations. In simple terms, in randomized smoothing we add noise to the input many times and output the majority vote over the outputs of the classifier for the noisy inputs. We present three works regarding this topic.
  - First, we show that the mathematical guarantees do not transfer to standard computer implementations of randomized smoothing. Then we develop a fix so that the guarantees hold not only in math, but also in the computer implementation.
  - Second, we have shown how to incorporate the domain constraints of the input to improve the robustness certificates. Namely, that the images are represented by pixel intensities between 0 and 1.
  - Third, we have revised the statistical estimation tasks, commonly appearing in the certification procedure. Concretely, we need to decide if the mean of a Bernoulli distribution (equivalently, the head probability in a coin toss) is larger than a certain constant and we have provided an optimal (up to constants) way to solve the problem in terms of samples required.

DE: Deep Learning ist zu einer dominierenden Technik im Bereich des maschinellen Lernens geworden und ist, mit nur wenig Übertreibung, zu einem Synonym für maschinelles Lernen selbst geworden. Trotz seiner großartigen Leistung bei Aufgaben, die von der Bildklassifikation bis zur Textgenerierung reichen, bleibt der zugrunde liegende Mechanismus weitgehend unverstanden, und die Deep-Learning-Systeme sind eine Black-Box, was zu einigen unerwünschten Eigenschaften führt, wie der Präsenz von adversarialen Beispielen. Ein adversariales Beispiel ist eine winzige Modifikation eines Inputs (in der Regel bei Bildern demonstriert), die für Menschen nicht wahrnehmbar ist und die Semantik des Inputs nicht verändert. Dennoch wird der Klassifikator getäuscht und gibt einen absurden Wert aus. Dies ist von großer Bedeutung in sicherheitskritischen Anwendungen, wie z. B. beim autonomen Fahren, wo die Konsequenzen solcher adversarialen Manipulationen potenziell katastrophal sind. In dieser Arbeit setzen wir die Bemühungen fort, das Problem zu mildern.

- In der ersten Veröffentlichung beobachten wir, dass das Problem nicht nur beim Deep Learning auftritt, sondern auch bei einfacheren Klassifikatoren, wie z. B. bei den nächstgelegenen Prototyp-Klassifikatoren. In diesem Fall leiten wir strenge mathematische Garantien zur Robustheit ab und liefern eine berechenbare untere Schranke für die Robustheit. Trotz der Verwendung einfacher Modelle konnten wir damit auf einem beliebten Benchmark Ergebnisse erzielen, die dem Stand der Technik entsprechen.

- Später konzentrieren wir uns auf Randomized Smoothing, eine Methode zur Zertifizierung der Robustheit eines Klassifikators gegenüber adversarialen Störungen. Einfach ausgedrückt fügen wir beim Randomized Smoothing dem Input mehrmals Rauschen hinzu und geben die Mehrheitsentscheidung basierend auf den Ausgaben des Klassifikators für die verrauschten Inputs aus. Wir präsentieren drei Arbeiten zu diesem Thema.
  - Zunächst zeigen wir, dass die mathematischen Garantien nicht auf Standard-Computerimplementierungen von Randomized Smoothing übertragbar sind. Dann entwickeln wir eine Lösung, damit die Garantien nicht nur mathematisch, sondern auch in der Computerimplementierung gelten.
  - Zweitens haben wir gezeigt, wie die Domänenbeschränkungen des Inputs integriert werden können, um die Robustheitszertifikate zu verbessern. Konkret geht es darum, dass Bilder durch Pixelintensitäten zwischen 0 und 1 dargestellt werden.
  - Drittens haben wir die statistischen Schätzaufgaben, die häufig im Zertifizierungsverfahren auftreten, überarbeitet. Konkret müssen wir entscheiden, ob der Mittelwert einer Bernoulli-Verteilung (äquivalent zur Wahrscheinlichkeit für "Kopf" beim Münzwurf) größer als eine bestimmte Konstante ist. Wir haben eine optimale (bis auf Konstanten) Methode zur Lösung des Problems in Bezug auf die erforderlichen Stichproben entwickelt.

translated with the help of chatgpt.

Acknowledgments

I thank my family, friends, coauthors, and the members of the groups from our corridor (Hein and von Luxburg) for many engaging discussions over the years during (and not limited to) the reading groups.

2 Introduction

Let us begin with a brief introduction to supervised statistical learning. We have a set of examples $\{(x_i, y_i)\}_{i=1}^n$, where x_i represents an input based on which we make a decision and y_i is the corresponding decision we want to learn. For example, $x_i \in \mathbb{R}^2$ might be an output of a sensor measuring the speed of the wind and the intensity of sunlight and y_i be a binary variable deciding if the blinds should be up or down, represented by 0 and 1 respectively. The goal is to find (*learn*) a function $F : \mathbb{R}^2 \to \{0, 1\}$ minimizing

$$\sum_{i=1}^{n} (F(x_i) - y_i)^2, \tag{2.1}$$

corresponding to the number of errors it makes on our set of examples. Then, we hope that on a fresh sample x, the output $F(x)$ will be an accurate predictor of the correct y and we can deploy F to control blinds. This can of course go wrong[1] and it is for another discussion when it happens.

To find a classifier F that is good on the task, we choose a set $\mathcal{F}$ of candidate functions beforehand from which we pick F. Usually, the candidate set is parameterized by some real numbers so that there is an almost everywhere differentiable function f, mapping the parameters and inputs to a real number. The corresponding classifier is then obtained by thresholding: $F_w(x) = [\![f(w, x) \geq 0.5]\!]$, or equivalently:

$$F_w(x) = \begin{cases} 0, & \text{if } f(w, x) < 0.5, \\ 1, & \text{otherwise.} \end{cases} \tag{2.2}$$

For example, one such set of functions is the set of linear classifiers. In our case, they are parameterized by weights $w \in \mathbb{R}^2$, $f(w, x) = \langle w, x \rangle$, and the corresponding set of functions is $\mathcal{F} = \{F_w \mid w \in \mathbb{R}^2\}$. Unfortunately, even for such a simple set of functions, the following optimization problem of selecting the best classifier is hard[2]:

$$\min_{F \in \mathcal{F}} \sum_{i=1}^{n} (F(x_i) - y_i)^2. \tag{2.3}$$

Thus, we instead solve the following minimization problem by gradient-based methods[3] and use the minimizer w and classifier F_w; we do not always find the actual minimizer,

[1] We possibly resort to curtains after several years of waiting.
[2] Arguably not hard in 2 dimensions, but NP-hard in general.
[3] We iteratively perform tiny updates of w, each of which decreases the objective value a bit.

but in practice, we can find good enough solutions this way.

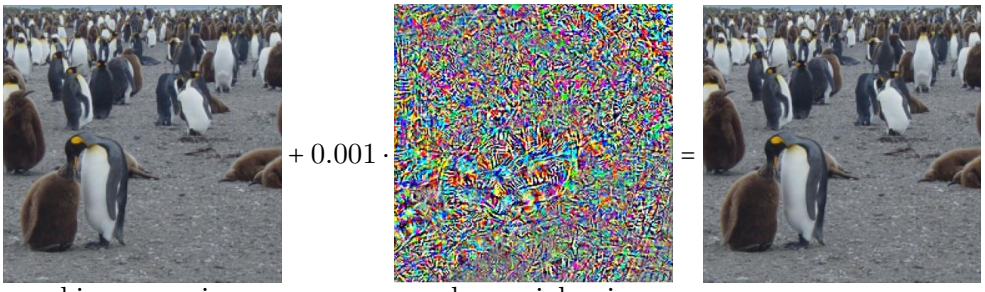
$$\min_{w \in \mathbb{R}^2} \sum_{i=1}^{n} \ell(f(w, x_i), y_i) \tag{2.4}$$

Here, ℓ is a suitable *loss* function such that $\ell(f(w, x_i), y_i) \geq (F_w(x_i) - y_i)^2$, in which case the minimization of (2.4) also leads to small values of (2.3). Simple examples are the squared loss $\ell(u, v) = 4(u - v)^2$ and the hinge loss $\ell(u, v) = v \max\{0, 1 - 2u\} + (1 - v) \max\{0, 2u - 1\}$, but more complex loss functions are used in practice. If we have more than two labels, things get slightly more complicated and we do not review it here since the spirit stays the same.

2.1 Adversarial robustness

In modern machine learning, we work with a very complicated set of functions - neural networks and also with very high dimensional inputs. For instance, RGB images with resolution 224×224 are represented by $\mathbb{R}^{150\,528}$, since every pixel is represented by 3 intensities of red, green and blue channel; $3 \cdot 224^2 = 150\,528$. It follows from the learning procedure described above that we do not have any actual control over how the classifier behaves outside of the set of presented examples and we only hope that it will be useful even for the unseen inputs. Luckily, this is often the case.

However, it was observed in Szegedy et al. (2014); Biggio et al. (2013) that there exist adversarial examples, i.e., small imperceptible modifications of the input which change the decision of the classifier. This property is of major concern in application areas where safety and security are critical such as medical diagnosis or autonomous driving. We illustrate this phenomenon in the following example, where a strong off-the-shelf classifier is fooled by imperceptible noise and mistakes a penguin for a carnivorous mammal. The image is taken from ImageNet dataset (Russakovsky et al., 2015).



king penguin | adversarial noise | mongoose

This thesis deals mainly with problems related to adversarial robustness. We start with the definitions to make things a little formal. Usually, we are interested only in the correct predictions being robust, but for the interest of simplicity, we do not require this in the definition. We also usually let the distance to be implicit from the context.

Definition 2.1. Let $F : \mathbb{R}^d \to \{0,1\}$ be a classifier, $d : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ be a distance function, and $x \in \mathbb{R}^d$ be an input.

- We say that $x' \in \mathbb{R}^d$ is an *ϵ-adversarial example* if $F(x) \neq F(x')$ and $d(x, x') < \epsilon$.
- We say that F is robust at x at radius ϵ if there is no $\epsilon-$adversarial example.

In the literature, it is common to consider the distance to be induced by a $p-$norm[4] with $p \in \{1, 2, \infty\}$; e.g., $d(x, x') = \|x - x'\|_2$ and we follow this custom. We note that there are also more exotic settings, for example where the distance measure approximates semantic dissimilarity of images (Wong et al., 2019; Levine and Feizi, 2020; Laidlaw et al., 2021; Voráček and Hein, 2022).

Usually, it is infeasible (co-NP hard for ReLU networks (Katz et al., 2017)) to decide if a classifier is $\epsilon-$robust at an input. There are two lines of research bypassing this hardness result:

- **Empirical robustness:** We search for an adversarial example in the ϵ-neighborhood of the input. If we fail to find such an adversarial example, we conclude that the classifier is $\epsilon-$robust here. This approach overestimates the actual robustness.
- **Certified robustness:** We claim that the classifier is $\epsilon-$robust only when we can prove it. This way, we generally underestimate the actual robustness.

In the thesis, we focus mainly on the second line - certified robustness.

2.1.1 Empirical robustness

We have already described that in this sub-field we rely on algorithms searching for adversarial examples - we call them adversarial *attacks*. We consider targeted attacks, they search for adversarial examples belonging to a certain class y. The optimization problem for checking $\epsilon-$robustness is very similar to the problem during learning as in (2.4), the difference is that the parameters w are fixed and we optimize over the input instead:

$$\begin{aligned} &\min_{x' \in [0,1]^d} \ell(f(w, x'), y), \\ &\text{s.t.} \quad \left\|x - x'\right\| \leq \epsilon. \end{aligned} \tag{2.5}$$

We have already argued that it is infeasible to solve Problem (2.5) exactly, so we might end up not finding the adversarial example due to optimization problems. To present a brief historical overview, problems of the form (2.5) were first attempted by L-BFGS (Szegedy et al., 2014). Later, Goodfellow et al. (2015) proposed to replace the objective of Problem (2.5) by its first-order Taylor approximation and dropped the box constraint on x'. The problem is then a maximization of a linear function in a norm-ball. For $p-$norms, these problems have closed-form solutions that pop out from the standard

[4] $\|x\|_p = \sqrt[p]{\sum_{i=1}^d |x_i|^p}$ for $1 \leq p < \infty$ and $\|x\|_\infty = \max_{i=1}^d |x_i|$.

proof of Hölder's inequality. Later, this approach was extended to perform multiple such steps and use projected gradient descent (resp. ascent, called PGD) Madry et al. (2018), and finally to use an ensemble of attacks containing PGD and other attacks not described here Croce and Hein (2020).

On the defending side, adversarial training Kurakin et al. (2017); Madry et al. (2018) is used to make networks resistant to adversarial attacks. The idea is that the training is performed on the adversarially attacked (usually a few steps of PGD) examples, highlighting the differences to standard training (2.4) in blue:

$$\min_{w\in\mathbb{R}^n} \max_{x_i' \mid \|x_i - x_i'\| \le \epsilon} \sum_{i=1}^{n} \ell(f(w, x_i'), y_i). \tag{2.6}$$

With this approach to adversarial robustness, it happened many times that defenses that were considered effective later turned out to be much weaker than originally claimed (Athalye et al., 2018; Croce and Hein, 2020; Tramer et al., 2020; Carlini et al., 2019); specifically, this happened to the majority of defenses based on different ideas than adversarial training. The classifiers are made robust to the known attack, so it might happen that a new attack will show that the networks currently considered robust in-fact are not.

2.1.2 Certified robustness

An alternative approach is to prove that a classifier is ϵ-robust at a point, but possibly be conservative. We split the discussion here into the deterministic and stochastic solutions.

Deterministic: Recalling our notation, we have a function $f(w, x)$ and the induced classifier is $F_w(x) = [\![f(w, x) \ge 0.5]\!]$ as in Equation (2.2). One common direction is to control the Lipschitz constant of the underlying function. If $f(w, x)$ is far from the decision boundary (i.e., $|f(w, x) - 0.5|$ is large), and the function $f(w, \cdot)$ has a small Lipschitz constant, then we might conclude F_w is robust at x. We make this argument more formal.

Definition 2.2. Let $h : \mathbb{R}^d \to \mathbb{R}$. Its Lipschitz constant w.r.t. norm $\|\cdot\|$ is at most L if for all distinct $x, y \in \mathbb{R}^d$ it holds that

$$L \ge \frac{|h(x) - h(y)|}{\|x - y\|}.$$

Theorem 2.3. *If $f(w, \cdot)$ has Lipschitz constant at most $L > 0$, then F_w is $|f(w, x) - 0.5| / L$ robust at x.*

Proof. If $|x' - x| < |f(w, x) - 0.5| / L$, then $|f(w, x) - f(w, x')| < |f(w, x) - 0.5|$ and so $F_w(x) = F_w(x')$. □

Neural network is usually a composition of simple operators; e.g., matrix multiplications, non-linearities, or additions; the Lipschitz constant of a network is then bounded per layer. For every operator, the Lipschitz constant is computed and their product is the

upper bound on the Lipschitz constant of the network. For examples of this direction, see Hein and Andriushchenko (2017); Li et al. (2019); Trockman and Kolter (2021); Leino et al. (2021); Singla et al. (2022) for the ℓ_2 threat model and Zhang et al. (2022) for ℓ_∞.

The alternative deterministic approaches usually consider ℓ_∞-norm bounded adversarial examples and exploit the layer-wise structure of neural networks. They propagate through the network a convex set (for example a product of intervals) that contains images of all the possible inputs from the ϵ-neighborhood mapped through the part of the network. possibly combined with mixed-integer linear programs or SMT; see, e.g., Katz et al. (2017); Gowal et al. (2018); Wong et al. (2018); Balunovic and Vechev (2020).

Stochastic: A popular way to get robustness certificates is randomized smoothing (Lecuyer et al., 2019; Cohen et al., 2019; Salman et al., 2019). Here, the intuition is that when we add a strong, random (for example Gaussian) noise to the input, we cannot distinguish the image from another image that differs from the original one only subtly.

We first describe smoothing with normal distribution for ℓ_2 robustness. For simplicity, we only certify class 1, the other case is symmetric. Let $F : \mathbb{R}^d \to \{0, 1\}$ be a *base classifier*. Its smoothed version is $h(x) = \mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2 I)} F(x + \varepsilon)$, and the resulting hard classifier is $H(x) = [\![h(x) > 0.5]\!]$. Using the Neyman-Pearson lemma the following (canonical) result has been shown:

Theorem 2.4 ((Cohen et al., 2019))**.** *Let $F : \mathbb{R}^d \to \{0, 1\}$ and $h(x) = \mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2 I)} F(x+\varepsilon)$, then $H(x') = 1$ for all x' with $\|x - x'\|_2 < \sigma\, \Phi^{-1}(h(x))$, where Φ^{-1} is the Gaussian quantile function.*

It is generally infeasible to evaluate $h(x)$, and in practice, we only approximate it by Monte-Carlo sampling; here every ε_i is a fresh sample from $\mathcal{N}(0, \sigma^2 I)$.

$$h(x) \approx \frac{1}{n} \sum_{i=1}^{n} F(x + \varepsilon_i), \tag{2.7}$$

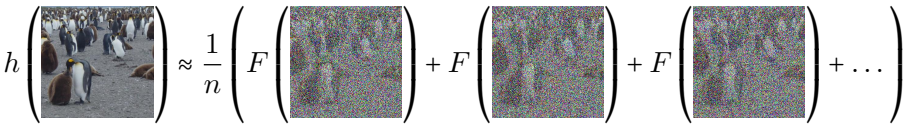


Figure 2.1: Illustration of Equation (2.7); penguin is from ImageNet (Russakovsky et al., 2015).

The estimation procedure is stochastic, so we cannot always correctly estimate the true underlying probability. To obtain rigorous guarantees, we need to control the estimation error in (2.7). If we underestimate $h(x)$, we only provide a weaker certificate than possible, so this type of error is not crucial. On the other hand, the overestimation of $h(x)$ leads to a stronger robustness claim than we can prove and this we need to control.

Usually, we allow this probability of overestimation to be $\alpha = 0.001$ and since in the RHS of (2.7) we have a sum of Bernoulli random variables, we can use Clopper-Pearson bounds to get the confidence interval at the desired level.

We obtain F in the similar way as we described earlier in Equation (2.4). The difference is that we add noise to the training examples x, as shown below. Again, highlighting differences in blue:

$$\min_{w \in \mathbb{R}^d} \sum_{i=1}^{n} \ell(f(w, x_i + \varepsilon_i), y_i), \tag{2.8}$$

where ε_i are independent samples of $\mathcal{N}(0, \sigma^2 I)$.

While randomized smoothing in Theorem 2.4 only considers additive Gaussian noise and provides guarantees against ℓ_2 perturbations, it is possible to extend this framework to virtually any reasonable distribution and distance metric (Yang et al., 2020). In Chapter 5, we describe the ℓ_1 case.

Despite many differences in different randomized smoothing algorithms, they almost always share the same statistical estimation task of estimating the mean of a Bernoulli distribution and we look at this more in detail in Chapter 6.

2.2 Objectives and Outcomes

The objective of this thesis is to advance the safety of machine learning systems; namely, to mitigate the problems caused by the presence of adversarial examples. The aim is to advance the rigorous approaches dealing with them and we (often) do so by exploiting the domain constraints of the images in computer representation.

The outcomes of this research are 4 papers accepted at major machine learning conferences (ICML, ICLR and NeurIPS).

1. In Voráček and Hein (2022) (outlined in Chapter 3), we study the adversarial robustness of nearest prototype classifiers. For example, we provide computational complexities of the problems related to adversarial robustness.

2. In Voráček and Hein (2023b) (outlined in Chapter 4), we demonstrate that the standard randomized smoothing certificates are no-longer valid in floating-point arithmetic and develop a sound implementation fixing the problem.

3. In Voráček and Hein (2023a) (outlined in Chapter 5), we improve the randomized smoothing certification procedure for ℓ_1-norm bounded adversarial robustness.

4. In Voráček (2024) (outlined in Chapter 6), we study the problem of statistical estimation deciding if the mean of a Bernoulli distribution is smaller or larger than a certain constant, and we provide an algorithms that draws the optimal amount (up to a constant) of samples in order to decide the task at high confidence.

Domain constraints

An overarching topic of the presented Chapters 3, 4, 5 is the consideration of domain constraints. We know that the images are represented by a vector of intensities $x \in [0,1]^d$. Therefore, we are not worried about potential adversarial inputs outside of the domain, since it is trivial to detect them and they would cause no harm. This idea is used in Chapters 3 and 5. More specifically, we know that the images are in fact not represented by real numbers but are quantized to 256 quantization levels, and this helped in Chapter 4 to efficiently implement certain procedures, which would take prohibitive time otherwise.

2.2.1 List of publications

This thesis contains a discussion of the following papers focusing on certifiable adversarial robustness:

- **Václav Voráček** and Matthias Hein. Provably Adversarially Robust Nearest Prototype Classifiers. In *ICML*, 2022
- **Václav Voráček** and Matthias Hein. Sound Randomized Smoothing in Floating-Point Arithmetics. In *ICLR*, 2023
- **Václav Voráček** and Matthias Hein. Improving ℓ_1-Certified Robustness via Randomized Smoothing by Leveraging Box Constraints. In *ICML*, 2023
- **Václav Voráček**. Treatment of Statistical Estimation Problems in Randomized Smoothing. *NeurIPS*, 2024

Individual contributions

Ideas often resulted from discussion and it is almost impossible (and subjective and meaningless) to provide a concrete credit assignment. Despite this, I tried to capture the research and writing process of every chapter in the respective backstories.

2.3 Thesis organization

We discuss 4 research papers in this thesis. We start the Chapters 3, 4, 5, 6 with a short story from behind the scenes. Next, we introduce the studied problems and outline the results with brief sketches of the relevant theory. These chapters are vague and imprecise in places for the sake of simplicity and contain adapted text from the papers; in other words, the actual papers were used as the bases of the chapters and were simplified. The full papers follow by the end of the thesis.

2.4 Notation

The notation is usually introduced at relevant places and its meaning is clear from the context. Here, we highlight that we use capital letters (X) for random variables and lower-case (x) for their realizations. If a is a vector or a sequence, then a_i is its i-th element. If instead, a_i is a vector, then $a_i^{(j)}$ is its j-th element. Iverson brackets $[\![P]\!]$ evaluate to 1 if P is true and to 0 otherwise.

3 Provably Adversarially Robust Nearest Prototype Classifiers

Václav Voráček and Matthias Hein. Provably Adversarially Robust Nearest Prototype Classifiers. In *ICML*, 2022

3.1 Backstory

In the summer of 2021, Matthias showed me a paper (Saralajew et al., 2020) achieving promising results despite using generic tools while specialized ones were readily available. He also made a MATLAB implementation of initial experiments for the $\ell_2 - \ell_2$ case and derived Theorem 2.4 and Proposition 3.1. I later derived the optimal solutions in other cases and was done with the paper since I thought that we needed either proofs or experiments. Matthias did not share this point of view with me, so we had to run some last-minute experiments. He also wrote the majority of the main paper.

3.2 Introduction

The nearest neighbor classifier is a classic machine learning algorithm that assigns a new sample to the same class as the closest element in the training set. The nearest prototype classifier (NPC) generalizes the nearest neighbor classifier by also learning the set of prototypes which replaces the training set during inference. The nearest neighbor classifier can also be seen as NPC where one uses the training set as prototypes and thus they are not learned. However, by training prototypes one can achieve better classification performance, and also robustness. In this work, we derive robustness guarantees of nearest neighbor-like classifiers and propose a training scheme yielding state-of-the-art performance on some benchmarks, outperforming neural-net-based approaches. In this chapter, we present some of the main ideas from the paper.

Problem Setting

We present the results here in a simplified notation compared to the actual paper which will be sufficient to describe the ideas. We have two classes A, B respectively. For both of them, there is a set of prototypes associated with them, $\{a_i\}_{i=1}^n \subset \mathbb{R}^d$ and $\{b_i\}_{i=1}^n \subset \mathbb{R}^d$

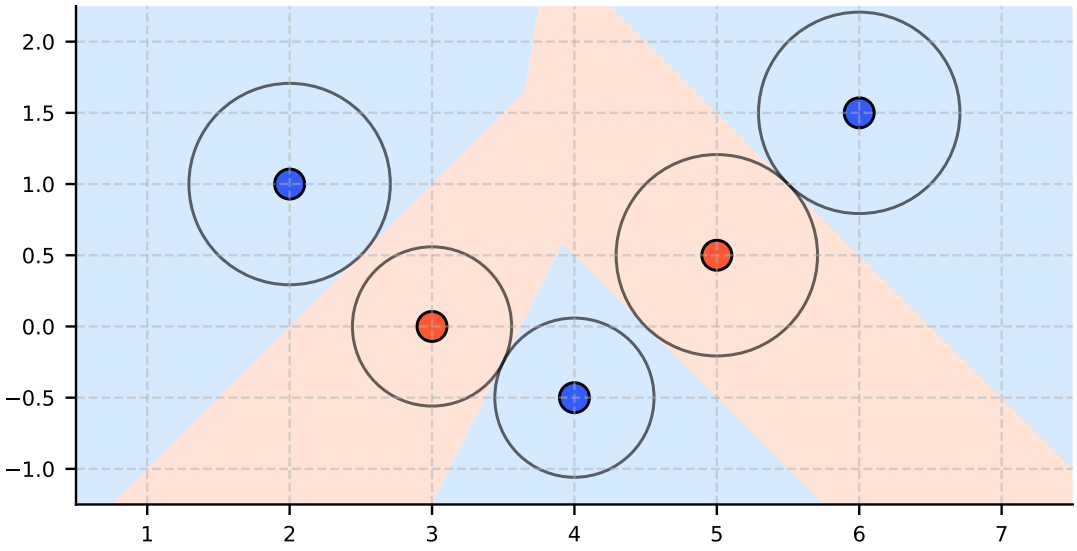


Figure 3.1: Illustration of an NPC. There are 5 prototypes in total corresponding to the red class and the blue class. The background color corresponds to the prediction of the classifier, or equivalently to the color of the closest prototype in Euclidean distance. We draw the largest possible circles around the prototypes such that the decision within the circle is constant.

respectively. Additionally, $d_{\text{classify}}(\cdot,\cdot)$ measures distances and we denote the classifier $F : \mathbb{R}^d \to \{A, B\}$. An example $x \in \mathbb{R}^d$ is classified as the same class as the closest prototype to it. Formally, we write

$$F(x) = \begin{cases} A, & \text{if } \min_{i=1}^{n} d_{\text{classify}}(x, a_i) < \min_{i=1}^{n} d_{\text{classify}}(x, b_i), \\ B, & \text{otherwise.} \end{cases}$$

See Figure 3.1 for an example of an NPC.

3.3 Overview of results

We are interested in certifying the robustness of F against adversarial perturbations bounded by $d_{\text{adversarial}}(\cdot,\cdot)$ distance. First, if $d_{\text{classify}}(\cdot,\cdot) = d_{\text{adversarial}}(\cdot,\cdot)$, then we call them both d and by triangle inequality, we have the following simple bound that cannot be improved in general.

Theorem 3.1. *Let $(\mathcal{X}, d)$ be a semi-metric space, F is ϵ-robust at x with*

$$\epsilon = \left| \frac{\min_{i=1}^{n} d(x, a_i) - \min_{i=1}^{n} d(x, b_i)}{2} \right|.$$

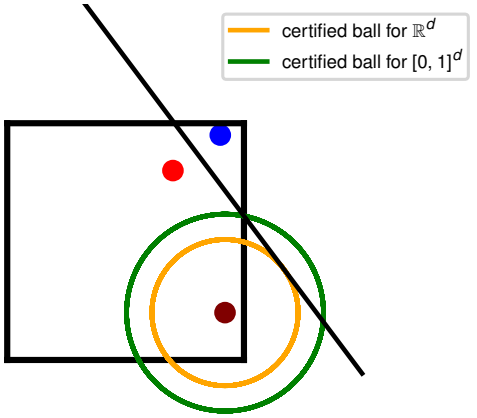


Figure 3.2: Illustration of NPC for two prototypes (red and blue): when considering that the data lies in $[0,1]^d$, we can certify a larger ball without adversarial examples around the input (wine) than in $\mathbb{R}^d$. We let $d_{\text{classify}}(x,y) = d_{\text{adversarial}}(x,y) = \|x-y\|_2$

This is also the result of Saralajew et al. (2020), although they provided a slightly less general version with a semi-norm instead of a semi-metric.

We further focus on a standard setting where both $d_{\text{classify}}(\cdot,\cdot)$ and $d_{\text{adversarial}}(\cdot,\cdot)$ are induced by (possibly different) p-norms. Our primary interest is in the cases where $p \in \{1,2,\infty\}$, though other cases are occasionally covered as a byproduct. Specially, let $d_{\text{classify}}(x,y) = \|x-y\|_p$ and $d_{\text{adversarial}}(x,y) = \|x-y\|_q$. To avoid ambiguity, we refer to ϵ-robustenss as ϵ_p^q-robustnes.

Next, we describe how to compute the maximum ϵ_p^q such that F is ϵ_p^q– robust. We focus only on robustness within the $[0,1]^d$ box, as this is the domain for images, and points outside it are easily detected. We demonstrate in Figure 3.2 the benefits of this constraint. Without loss of generality, assume $F(x) = A$. For every prototype b_i, we find the closest point x' to x (in terms of q-norm) such that b_i is closer to x' (in terms of p-norm) than any prototype of class A. The smallest such distance is ϵ_p^q, then F is ϵ_p^q–robust at x. More formally, we have to solve the following optimization problem:

Proposition 3.2. *Let $F(x) = A$ and $\{a_i\}_{i=1}^n, \{b_i\}_{i=1}^n$ be the prototypes corresponding to classes A, B respectively. Let*

$$\begin{aligned} r_p^q(x,b) = \min_{x' \in [0,1]^d} \quad & \left\|x - x'\right\|_q, \\ \text{sbj. to:} \quad & \left\|x' - b\right\|_p \le \left\|x' - a_i\right\|_p, \quad \forall i \in \{1,\dots,n\}. \end{aligned} \tag{3.1}$$

then F is ϵ_p^q-robust at x where

$$\epsilon_p^q(x) = \min_{i=1}^{n} r_p^q(x, b_i).$$

ℓ_p-distance	ℓ_q-threat model		
	ℓ_1	ℓ_2	ℓ_∞
ℓ_1	NP-hard	NP-hard	Poly
ℓ_2	Poly	Poly	Poly
ℓ_∞	NP-hard	NP-hard	NP-hard

Table 3.1: Computational Complexity of $r_p^q(x, b)$.

ℓ_p-distance	ℓ_q-threat model		
	ℓ_1	ℓ_2	ℓ_∞
ℓ_1	NP-hard	NP-hard	$O(d\log(d))$
ℓ_2	$\Theta(d)$	$\Theta(d)$	$\Theta(d)$
ℓ_∞	$\Theta(d)$	$O(d\log(d))$	$\Theta(d)$

Table 3.2: Computational complexity of $\rho_p^q(x, a, b)$.

Before we inspect the complexity of optimization Problems (3.1) more in detail, we note that direct maximization of ϵ_p^q during training is infeasible, particularly since the problems are non-convex whenever $p \neq 2$. To address this, we propose the following relaxed optimization problem:

$$\rho_p^q(x, a, b) = \min_{x' \in \mathbb{R}^d} \quad \left\|x - x'\right\|_q \tag{3.2}$$

$$\text{sbj. to:} \quad \left\|x' - b\right\|_p \leq \left\|x' - a\right\|_p \tag{3.3}$$

and it holds that

$$r_p^q(x, b) \geq \max_{i=1}^{n} \rho_p^q(x, a_i, b),$$

since we effectively drop all but (the strongest) constraint from Problem (3.1); and thus we have that

$$\epsilon_p^q(z) \geq \min_{j=1}^{n} \max_{i=1}^{n} \rho_p^q(x, a_i, b_j),$$

where the RHS will be maximized during training.

We now present the complexities of the Problems (3.1) and (3.2). The log factors can be removed by techniques analogous to linear-time median computations.

Theorem 3.3. *Computational complexities of optimization problems $\epsilon_p^q(x, b)$ for $p, q \in \{1, 2, \infty\}$ and are summarized in Table 3.1.*

Theorem 3.4. *Computational complexities of optimization problems $\rho_p^q(x, a, b)$ for $p, q \in \{1, 2, \infty\}$ are summarized in Table 3.2.*

We sketch proof ideas of several entries of the tables from Theorems 3.3 and 3.4. The other entries required individual and rather cumbersome treatment.

- In the row when $p = 2$, the constraint in Problem (3.2) is a linear constraint and we need to find an ℓ_q projection on a hyperplane. Here, the solution follows directly from Hölder's inequality and we only need to compute the dual norm of a certain vector. Similarly, Problem (3.1), reduces to projecting a point to a polytope. This is a linear program when $q \in \{1, \infty\}$ and a quadratic program when $q = 2$.

- In the row for $p = 1$ and $q \neq \infty$, the knapsack problem can be reduced to Problem (3.2) implying the hardness result. For simplicity, we only present an example of the reduction for $q = 1$ from which we believe the general case easily follows. Consider an instance of a knapsack problem, having two items with prices $5, 10$ and values $3, 4$ respectively. The goal is to minimize the price for which we can buy value 6. We construct an instance of Problem (3.2) that is equivalent to the described knapsack problem. Let
  - $x = (0, 0, 0)$,
  - $a = (5, 10, 100)$.
  - $b = (5.03, 10.04, 100.05)$,

  In this configuration, we observe that for $x' = (0, 0, 0)$, the constraint $\|x' - b\|_1 \leq \|x' - a\|_1$ is violated by 0.12 (we call the quantity *constraint gap*), corresponding to two times our desired value. We also note that the third coordinate is a "dummy" coordinate and in the optimal solution we will have $x'^{(3)} = 0$. Both the first and second coordinates correspond to the respective items. We interpret the quantity $\left|x'^{(1)}\right|$ as the price we pay for the first item. If we pay less than 5, we do not decrease the constraint gap at all (we do not buy the item in knapsack problem). If we pay 5.03 or more. We decrease it by 0.6, corresponding to twice the value of the first item (we buy the item in the knapsack problem) We note that by construction, this difference between paying cost 5 and 5.03 is negligible and the reduction is complete.

- For $p = \infty$, we reduce 3-SAT to the feasibility version Problem (3.1); again, on an example. We start with a CNF formula $(A \wedge \neg B \wedge C) \vee (A \wedge C \wedge \neg D)$ and the corresponding Problem (3.1) is given by:
  - $b = (0, 0, 0, 0, 3)$
  - $a_1 = (-1, 2, -1, 0, 0)$
  - $a_2 = (-1, 0, -1, 2, 0)$

  That is, for every clause, we have a prototype a_i. The number of dimensions is by one larger than the number of logical variables. Prototype a_i contains at $j - th$ position one of the numbers $-1, 0, 2$, corresponding to j-th variable occurring in i-th clause without negation/not occurring/occurring with negation (with lexicographic ordering of variables). The last dimension is a dummy dimension ensuring that

$\|x' - b\|_\infty \geq 2$ for any x', where equality can easily be attained. Now, in order for the problem to be feasible, we have to find an x' such that $\|x' - a_i\| \geq 2$. For $i = 1$ this means either $x'^{(1)} = 1$, or $x'^{(2)} = 0$, or $x'^{(3)} = 1$ - recalling $x' \in [0, 1]^d$ - which directly corresponds to the satisfying assignment of the clause $A \vee \neg B \vee C$.

3.4 Conclusion

We presented a selection of theoretical results from the paper. The paper additionally contains details on training, experimental results, extension to the perceptual-metric-based robustness, and some implementation tricks. For example, we do not want to solve thousands of linear/quadratic programs per one certified point as a naive application of Proposition 3.2 would suggest. Instead, we solve roughly 2 linear/quadratic programs on average.

The method yielded state-of-the-art robustness results on MNIST with ℓ_2 threat model and also on CIFAR10 with LPIPS threat model. This still holds.

4 Sound Randomized Smoothing in Floating-Point Accuracy

Václav Voráček and Matthias Hein. Sound Randomized Smoothing in Floating-Point Accuracy. In *ICLR*, 2023

4.1 Backstory

In the spring of 2022, I reviewed a paper about randomized smoothing containing a numerical integration subroutine without a convincing error analysis, which in turn made the resulting certificates unreliable in my eyes. Then I asked myself a question, if even the standard randomized smoothing machinery should be trusted, or is it just a theoretical construct that cannot be reliably implemented to have the guarantees? But the theoretical guarantees are the reason why we use randomized smoothing in the first place! Soon after I constructed an example where the standard randomized smoothing implementation fails completely. At that point, I was satisfied with the (negative) result and considered it the end of the story. A week or so before the submission Matthias asked if it could be fixed. It sounded too optimistic to realistically consider that possibility before, but now the question was asked, I had to think about it. In the end, it could. I wrote the majority of the paper.

4.2 Introduction

Randomized smoothing is a popular technique to certify the robustness of classifiers. It is however not clear if the certificates also hold when they are implemented in finite precision. We first show that it is not the case and present a simple example where randomized smoothing certifies a radius of 1.26 around a point, even though there is an adversarial example in the distance 0.8 and describe how this can be abused to give false certificates for CIFAR10. In order to overcome this problem, we propose a sound approach to randomized smoothing when using floating-point precision with essentially equal speed for quantized inputs. It yields sound certificates for image classifiers which are virtually equal to the unsound practice of randomized smoothing. Our only assumption is that we have access to a fair coin. Before diving into the problem, we provide a brief overview of floating-point representations and arithmetic.

Floating Point representation

Single-precision floating-point[1] numbers are represented in memory as sequences of bits $x_1x_2\dots x_{32}$. The first bit is a sign bit, the next 8 bits determine the exponent, and the last 23 numbers determine the mantissa. The conversion in the normalized form reminds of the scientific notation: $(-1)^{x_1}\cdot 2^{x_2\dots x_9-127}\cdot 1.x_{10}\dots x_{32}$, or more formally:

$$(-1)^{x_1}\cdot 2^{\left(\Sigma_{i=2}^{9}x_i\cdot 2^{9-i}\right)-127}\cdot\left(1+\sum_{i=10}^{32}x_i\cdot 2^{9-i}\right).$$

We write the floating-point operations in circles; e.g., $\oplus,\ominus$ instead of $+,-$ to distinguish them from the mathematical ones which do not suffer from rounding errors. The addition (or analogically subtraction) of two floating-point numbers is performed in three steps: (1) The number with the lower exponent is transformed to the higher exponent and the superfluous mantissa bits are rounded. (2) The addition is performed, with - to simplify presentation - infinite precision. (3) The result is rounded to fit into the floating-point representation. Thus, the operations are commutative, but it happens that $x\oplus y = x\oplus z$ for any $x\neq 0$ and some $y\neq z$. Consequently, there will exist some w such that there is no v for which $x\oplus v = w$. This is the main observation that we will exploit in the sequel.

4.3 Overview of results

In the standard randomized smoothing implementation, there are three main places where the math and computer implementation might differ; before presenting them, we refresh the relevant randomized smoothing mechanism for smoothing with normally distributed smoothing distribution[2]. We need to evaluate the following quantity for an arbitrary $F:\mathbb{R}^d\to\{0,1\}$:

$$h(x) = \mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2 I)}F(x+\varepsilon).$$

Due to intractability issues, $h(x)$ is approximated by the Monte-Carlo sampling and a high-probability lower bound (for certifying class 1) is computed. The critical computations[3] in the standard randomized smoothing machinery are the following ones:

1. Sample noise $\varepsilon\sim\mathcal{N}(0,\sigma^2 I)$.
2. For an input x, compute $x' = x+\varepsilon$.
3. Evaluate $F(x')$.

Evaluating $F(x')$ correctly is not critical. It might happen due to floating point errors that the output differs in the computer evaluation and in the "actual math evaluation". However, as long as we are only interested in certifying the computer-represented function, we do not need to worry as the method is assumption-free on the underlying classifier F.

[1]The other precisions only differ in the counts of mantissa and exponent bits.
[2]The problems and solutions for other smoothing distributions considered in the literature are analogical.
[3]The others can be easily performed correctly.

Computations (1) and (2) are trickier. Specifically, (1) is impossible to perform exactly due to the continuous nature of the distribution represented using discrete values, and thus the correctness of the sampling needs to be ensured in some way.

Demonstration of wrong certificates

In the following discussion, we will be informal and will simplify examples to improve clarity. The key observation is that random variables $X = x \oplus \mathcal{N}(0, I_d)$ and $Y = y \oplus \mathcal{N}(0, I_d)$ with[4] $x, y \in \{0/255, \ldots, 255/255\}^d$ do not have completely overlapping supports and it might happen that when observing a sample from X, we can tell that it cannot be a sample from Y. We will demonstrate this phenomenon in one dimension, but the problem becomes even more pronounced as the dimension grows, which is the interesting regime as we are mainly interested in image classification. To see why, let the probability that $x \sim X$ falls outside of the support of Y be at least p when $x \neq y$ and $d = 1$. Now consider general d and x, y differing in every position, the probability that $x \sim X$ is in the support of Y is smaller than $(1-p)^d$. We can empirically check that $p = 0.01$ is a good bound, and for $d \sim 1000$ we already get $(1-p)^d \sim 10^{-5}$. Therefore, by observing samples $F(X)$, we likely get no information about $F(Y)$, contrasting the intuition for randomized smoothing.

```python
import numpy as np
from scipy.stats import norm

sigma = 0.5; num_samples = 100000; alpha = 0.001
F = lambda x: (x - 210/255) + 210/255 == x
noise = np.random.randn(num_samples)*sigma

h0 = F(0+noise).mean()                                  # 0.461
hx = F(210/255+noise).mean()                            # 1.000
hxlb = p2-(-np.log(alpha)/num_samples/2)**0.5           # 0.994
eps = sigma * norm.ppf(hxlb)                            # 1.260
```

Listing 4.1: Example of an incorrect randomized smoothing certificate. We compute $h(0) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,\sigma^2)} F(0 \oplus \varepsilon) < 0.5$, and so $H(0) = 0$. On the other hand, we have $h(x) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,\sigma^2)} F(210/255 \oplus \varepsilon) > h(x)_{\text{lower bound}} = 0.994$ w.h.p. due to Hoeffdings' inequality. Thus, H is ϵ-robust at $x = 210/255$ for class 1 with $\epsilon = \sigma \Phi^{-1}(0.994) = 1.26$, but 0 is an ϵ-adversarial example - a contradiction.

Now we describe the construction. In Reiser and Knuth (1975), it is shown that the identity $((x \oplus y) \ominus y) \oplus y = x \oplus y$ holds for virtually every pair of x, y. On the other hand, the equality $(x \oplus y) \ominus y = x$ holds only sometimes. Indeed, consider x to have a different exponent than y, then during the addition, $x \oplus y$, some low bits of the mantissa are lost. We note that in the special case when $x = a \ominus y$ for some a and y, the identity $(x \oplus y) \ominus y = x$ holds true. On the other hand, all the discussed identities clearly hold when we replace $\oplus, \ominus$ by $+, -$. This difference will be exploited using the following function:

[4] We are interested in image classification, so we consider the standard quantization

$$F_a(x) = [\![(x \oplus a) \ominus a = x]\!]. \tag{4.1}$$

From the discussion above it follows that $F_a(a \oplus \varepsilon) = 1$ for any ε, possibly yielding arbitrarily large certified radii around a. However, when $x \neq a$, then $F_a(x \oplus \varepsilon)$ might equal 0. Note the sharp contrast with the exact arithmetic case, where $F_a(\cdot) = 1$. The implementation producing an incorrect certificate is in Listing 4.1 with $a = 210/255$.

Sound randomized smoothing for floating-point arithmetic

Let us now describe a sound randomized smoothing certification procedure for floating-point arithmetic. The idea is to introduce a mapping g and replace the base classifier $F(\cdot)$ by $F(g(\cdot))$ and instead of evaluating $F(g(x \oplus \varepsilon))$ where $\varepsilon \sim \mathcal{N}(0, \sigma^2 I_d)$, we evaluate $F(t)$ with $t \sim g(x + \varepsilon), \varepsilon \sim \mathcal{N}(0, \sigma^2 I_d)$. This (seemingly useless) modification has two advantages: (1) we no longer need to perform floating point addition $x \oplus \varepsilon$ and (2) we do not need to sample ε anymore, we only need to sample t which might be easy when t follows a discrete distribution. Furthermore, if $g(x) \sim x$, then we might use an existing base classifier F trained for the standard smoothing certification. We choose

$$g(x) = \max\left\{-6\sigma, \min\left\{1 + 6\sigma, \frac{\lfloor 255x \rfloor}{255}\right\}\right\}. \tag{4.2}$$

Now t follows a multinomial distribution with corresponding probabilities $p_1, \dots p_n$ ($n \sim 10^3$) and we can compute all of them with arbitrary precision, but not exactly as they are some Gaussian integrals. This way, we can control the error of the sampler and account for it. We use k uniformly random bits for sampling t. We evaluate p_i precise enough that we get an integer a_i such that $a_i/2^k \leq p_i \leq (a_i + 1)/2^k$; we sample the respective outcomes with probabilities $a_i/2^k$. Alternatively, we declare a failure with probability $1 - \sum_{i=1}^{n} a_i/2^k$. If this failure occurs, we set $F(t) = 0$ (recalling we certify class 1) to be conservative. We choose $k = 64$ so this failure is extremely rare. All the values of a can be precomputed before certification which takes roughly 4 minutes on a single core of a laptop. We have conducted experiments and (perhaps unsurprisingly) found out that replacing the trained base classifier $F(\cdot)$ (ResNet) by $F(g(\cdot))$ with g from Equation (4.2) yields negligible differences in certificates on CIFAR10 and ImageNet.

4.4 Conclusion

We have provided an example where the randomized smoothing certificates are incorrect and explained how to fix it. The paper contains further arguments, insights and empirical validation. We focused for simplicity on Gaussian noise, but the problems and fix transfer to other smoothing distributions. We consider the results definitive and are not aware of any new work in this direction.

5 Improving ℓ_1 Certified Robustness via Randomized Smoothing By Leveraging Box Constraints

Václav Voráček and Matthias Hein. Improving ℓ_1-Certified Robustness via Randomized Smoothing by Leveraging Box Constraints. In *ICML*, 2023.

5.1 Backstory

In January 2023, I had some theoretical results on how to incorporate image constraints into randomized smoothing machinery and had some more ideas on how to extend them. After a discussion with Matthias, it turned out that the curse of dimensionality kills all the ideas and there will not be many more things that could possibly fit into the paper. In the 4 days before the submission, I wrote the paper and did the experiments. On the last day, Matthias focused on improving the writing (At that point, the paper was a collection of semi-related paragraphs in a semi-random order), while it was not yet completely clear what exactly would be in the paper. The resulting paper reads surprisingly well and resulted in the smoothest review experience so far. Eventually, the poster was presented by my friend Julian as I find hosting conferences in Hawaii absurd.

5.2 Introduction

The guarantees for randomized smoothing are often derived by constructing the worst-case classifier consistent with the observed samples and then considering the worst-case perturbation for it. However, it happens that sometimes the worst-case perturbation is outside of the image domain and this problem was largely overlooked.

In this chapter, we show that by taking into account the box constraints of the image domain $[0,1]^d$, we can certify significantly larger ℓ_1 robustness than before. Specially, we use the fact that the minimal volume of the overlap of two ℓ_∞-balls when the centers (their distance is fixed in ℓ_1 norm) of the balls are restricted to $[0,1]^d$ behaves quite differently from the unconstrained case.

5.3 Overview of results

In certifying ℓ_1 robustness, the common starting point is the following Proposition (We certify only class 1 to avoid notation clutter):

Proposition 5.1. *Let $F : \mathbb{R}^d \to \{0,1\}$, $\mathcal{U}^d(\lambda)$ be the uniform distribution in ℓ_∞ ball of radius λ and*

$$h(x) = \mathop{\mathbb{E}}_{\varepsilon \sim \mathcal{U}^d(\lambda)} F(x+\varepsilon).$$

Let B_1 and B_2 be the ℓ_∞-balls with radius λ centered at x, y respectively, then

$$h(y) \geq h(x) - 1 + \frac{Vol(B_1 \cap B_2)}{Vol(B_2)},$$

where $Vol(B)$ is volume of B.

Proof.

$$\begin{aligned} h(y) &= \frac{\int_{t \in B_2} F(t)dt}{Vol(B_2)} \geq \frac{\int_{t \in B_1 \cap B_2} F(t)dt}{Vol(B_2)} \\ &= \frac{\int_{t \in B_1} F(t)dt - \int_{t \in B_1 \setminus B_2} F(t)dt}{Vol(B_2)} \\ &\geq \frac{\int_{t \in B_1} F(t)dt - Vol(B_1 \setminus B_2)}{Vol(B_2)} \\ &= h(x) - 1 + \frac{Vol(B_1 \cap B_2)}{Vol(B_2)}, \end{aligned}$$

using $Vol(B_1 \setminus B_2) = Vol(B_2) - Vol(B_1 \cap B_2)$ and $Vol(B_1) = Vol(B_2)$. □

To get certificates, it remains to find a lower bound on the volume of intersection of two ℓ_∞-balls (B_1 and B_2) of radii λ centered at $x, y \in \mathbb{R}^d$ respectively. For a fixed x (the input point to be certified), this quantity can be expressed as

$$f_x(y) = \frac{Vol(B_1 \cap B_2)}{Vol(B_1)} = \prod_{i=1}^{d} \left(1 - \frac{|x_i - y_i|}{2\lambda}\right),$$

Thus, we need to find y closest to x for which we cannot certify that the class is 1; more formally, we are looking for $y \in \mathbb{R}^d$ minimizing $\|x-y\|_1$ such that $h(x) - 1 + f_x(y) = \frac{1}{2}$. It turns out that it is easier to solve the (somewhat equivalent) minimization of $f_x(y)$ given the value $\|x-y\|_1$.

We claim that $f_x(y)$ is minimized when x and y differ only at a single position. To see why, we give $f_x(y)$ a probabilistic interpretation. We have d independent events, where the i-th event occurs with probability $\frac{|x_i - y_i|}{2\lambda}$. Then $1 - f_x(y)$ is the probability that at least one of them occurred. By union bound, we have

$$1 - f_x(y) \leq \sum_{i=1}^{d} \frac{|x_i - y_i|}{2\lambda} = \frac{\|x-y\|_1}{2\lambda},$$

and thus we get the following bound

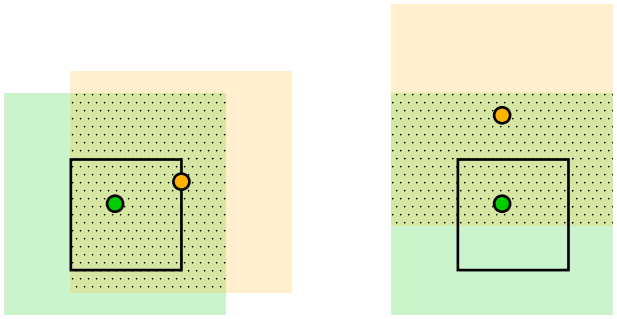


Figure 5.1: Effect of box constraints on the minimal overlap of two ℓ_∞ balls. The green point is at $(0.4, 0.6)$ while the orange one is at a distance 0.8 in ℓ_1-norm. On the left is depicted the minimal possible overlap considering box constraints (cf. Proposition 5.4), while on the right is the minimal possible overlap without considering box constraints (cf. Proposition 5.2)

.

Proposition 5.2. *Let B_1, B_2 be the ℓ_∞ balls with radii λ centered at $x, y \in \mathbb{R}^d$ Then we get the tight bound:*

$$\frac{Vol(B_1 \cap B_2)}{Vol(B_1)} \geq 1 - \frac{\|x-y\|_1}{2\lambda}.$$

This lower bound is attained when $x - y$ is a one-hot vector. Combination of Proposition 5.1 and Proposition 5.2 yields the following bound:

Corollary 5.3. *Classifier H is ϵ-robust at x with*

$$\epsilon = 2\lambda(h(x) - 1/2).$$

However, as discussed, the bound in Proposition 5.2 is only attained in the case when $x - y$ is a one-hot vector. When $\epsilon \geq 1$, then the minimizer y would be necessary outside of the image domain and we do not need to worry about it. Therefore, we would like to minimize $f_x(y)$ (with given $\|x-y\|_1$) and under the constraint that $y \in [0,1]^d$. We illustrate this in Figure 5.1.

The problem can be efficiently solved in a closed form for any x since f_x is Schur-concave and so we only need to find the element of the constraint set whose vector of differences $|x_i - y_i|$ is largest w.r.t. the majorization[1] order. For example, if we consider $\|x-y\| = 4.5$, then the majorizing vector would be $(1, 1, 1, 1, \frac{1}{2}, 0, \ldots, 0)$. In that case, we get the following bound:

[1]We say that x majorizes y when $\sum_{i=1}^k x_i^\downarrow \geq \sum_{i=1}^k y_i^\downarrow$ for all $1 \leq k \leq n$, where $x_i^\downarrow$ is the i–th largest element of vector x.

Proposition 5.4. *Let B_1, B_2 be the ℓ_∞ balls with radii λ centered at $x, y \in [0,1]^d$ and $\lfloor x \rfloor$ be the largest integer not larger than x. Then*

$$\frac{Vol(B_1 \cap B_2)}{Vol(B_1)} \geq \left(1 - \frac{1}{2\lambda}\right)^{\lfloor \|x-y\|_1 \rfloor} \left(1 - \frac{\|x-y\|_1 - \lfloor \|x-y\|_1 \rfloor}{2\lambda}\right) \geq \left(1 - \frac{1}{2\lambda}\right)^{\|x-y\|_1}.$$

The second inequality holds when $2\lambda \geq 1$. Both of the inequalities are attainable.

Consequently, we can combine Propositions 5.4 and 5.1 and streghten Corollary 5.3.

Corollary 5.5. *Classifier H is ϵ-robust at x with*

$$\epsilon = \frac{\ln(1.5 - h(x))}{\ln(1 - \frac{1}{2\lambda})}.$$

We can take one step further and provide a certificate specific for x. For example, if $x = (0.5, 0.5, \ldots, 0.5)$, then the largest element of the majorizing vector could only be 0.5. In this case, the final expression becomes cumbersome and thus we omit it.

5.4 Conclusion

We presented the central ideas of the corresponding paper. We omitted here the experimental evaluation and some further discussions mainly concerning statistical estimation. The resulting formulas yield state-of-the-art results and have not been improved so far.

6 Treatment of Statistical Estimation Problems in Randomized Smoothing

Václav Voráček. Treatment of Statistical Estimation Problems in Randomized Smoothing. *NeurIPS*, 2024

6.1 Backstory

The notorious problem of randomized smoothing is the speed. The custom is to sample 100 000 noise samples per image (taking seconds to minutes per image) and little work was done to speed it up. I started working on the problem in the fall of 2022, with the idea of designing a certain sequential estimation procedure. I had Walds' Sequential Analysis book but unfortunately, the book is not formal in many aspects and it was not clear to me if some of the results are approximate, or can be made formal (in hindsight... Yes, they can). In spring 2023, I presented some preliminary results in our group meeting. Preliminary in the sense that I could not prove the correctness of the algorithm, but it turned out to be very similar to the one I eventually used. In the fall of 2023, I found a relevant martingale inequality that filled the gap and I could finally complete the paper. Later I found an even simpler solution to the problem.

6.2 Introduction

The common criticism of randomized smoothing is the prohibitive time to certify the robustness. In the standard setting, this is roughly 8 seconds per CIFAR-10 image or 2 minutes per ImageNet image. There is an inherent trade-off between the allowed probability of incorrectly claiming robustness[1] (type-1 error, α), the probability of incorrectly claiming non-robustness (type-2 error, β), and the number of samples used n. The standard practice is to set $\alpha = 0.001$, $n = 100\ 000$, and the value of β is then implicit. it might not be the most practically relevant setting since the implicitly set value of β is usually exponentially small in n.

The arguably more natural setting is to set the values of α and β and leave n implicit. This is much more challenging since it is no longer possible to draw the predetermined

[1] of an input for a model at a certain radius

number of samples and invoke a concentration inequality. We propose a new certification procedure using confidence sequences to adaptively (and optimally) decide how many samples to draw to address the problem.

Additionally, we show that the standard Clopper-Pearson confidence intervals - used in virtually every randomized smoothing certification procedure - are conservative in general and present their randomized version that has the exact coverage.

6.3 Overview of results

We recall that in randomized smoothing, we have a classifier $F : \mathbb{R}^d \to \{0, 1\}$, its smoothed version is $h(x) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,\sigma^2)} F(x + \varepsilon)$, and the resulting classifier H is ϵ-robust with $\epsilon = \sigma \Phi^{-1}(h(x))$. Now consider the task where we want to decide if H is 1-robust. This is equivalent to deciding if $h(x) \geq \Phi(1/\sigma)$; in other words, if the mean of a Bernoulli random variable given by $F(x + \varepsilon)$ with $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ is larger than a certain threshold. Thus, we further consider only the statistical estimation problems regarding the mean of a Bernoulli (resp. binomial[2]) random variable.

Confidence intervals

First, we have a look at the confidence intervals for binomials appearing in almost every current randomized smoothing procedure. We introduce only the one-sided versions for simplicity as this is the one used in randomized smoothing.

Definition 6.1 (Confidence interval for binomials)**.** Let l be a possibly randomized function of a binomial sample. It forms a lower confidence bound on the mean with coverage $1 - \alpha$ if for any $p \in [0, 1]$ it holds that

$$\mathbb{P}_{X \sim \mathcal{B}(n,p), l} \left(p \geq l(X) \right) \geq 1 - \alpha.$$

- Clopper-Pearson lower confidence bound is given by

$$l(x) = \inf\{p \,|\, \mathbb{P}(\mathcal{B}(n, p) \geq x) > \alpha\}.$$

- Randomized Clopper-Pearson lower confidence bound is given by $l_r(x) = l'_r(x, W)$ where W is uniform on the interval $[0, 1]$ and

$$l'_r(x, w) = \inf\{p \,|\, \mathbb{P}(\mathcal{B}(n, p) > x) + w\mathbb{P}(\mathcal{B}(n, p) = x) > \alpha\}.$$

It can be checked by straightforward calculations that Randomized Clopper-Pearson intervals have coverage exactly $1 - \alpha$ for all values of $0 \leq p \leq 1$ and that almost surely (whenever $w \neq 0$) the randomized interval is shorter. We present a classical example implying that the standard Clopper-Pearson interval is conservative in general.

[2]We denote $\mathcal{B}(n, p)$ the binomial random variable with n trials and success probability p.

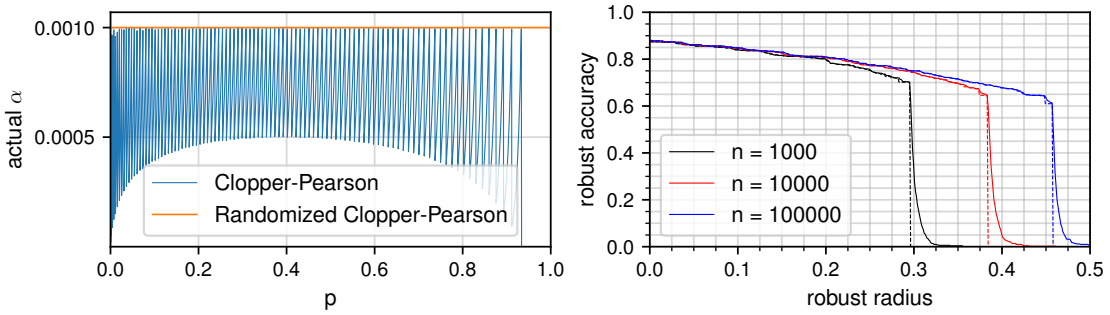


Figure 6.1: **left**: Comparison of coverages of confidence intervals for the mean estimation of $\mathcal{B}(100, p)$ when $\alpha := 0.001$. *Actual* α corresponds to how often is p outside of the confidence interval. **right**: Comparison of robustness curves with the standard (dashed) or the randomized (solid) Clopper-Pearson bounds on a CIFAR-10 dataset under the standard setting, where n is the number of samples. Point (x, y) on a curve captures the fact that the classifier is x-robust on a y-fraction of points from the dataset.

Example 6.2. Consider samples from $X \sim \mathcal{B}(2, p)$ and $\alpha = 0.05$. By definition, the Clopper-Pearson upper intervals are $[0, 1]$, $[0.025, 1]$, $[0.224, 1]$ for observations $x = 0, 1, 2$ respectively. When $p = 0.1$, it will be contained in the confidence interval with probability $\mathbb{P}(X \in \{0, 1\}) = 1 - p^2 = 0.99$, while we required it only to be 0.95 and so the confidence interval is unnecessarily large, yielding weaker certificates.

We compare the randomized and deterministic Clopper-Pearson intervals in Figure 6.1 and show the corresponding differences in certifying robustness.

Confidence sequences

To compute the confidence intervals presented in the previous subsection, we need to collect samples and then run an estimation procedure once which brings certain limitations. Consider the following two scenarios: (1) It might be the case that we do not need all 100 000 samples and after only 10 it would be enough for our purposes because we could already conclude that the point cannot be certified here; thus, we wasted 99 990 samples. (2) Alternatively, we could see that even 10^5 samples are not enough, and we need to draw more samples. However, we have already spent our failure budget α, so we cannot even carry another test at all.

This motivates the introduction of confidence sequences. They generalize confidence intervals in the way that they provide a confidence interval after every received sample such that we control the probability that the true parameter is contained in *all* the confidence intervals *simultaneously*.

Definition 6.3 (Confidence sequence)**.** Let $\{u_t, v_t\}_{t=1}^{\infty}$ be mappings from a sequence of observations to a real number. They form a confidence sequence $I_t(\mathbf{x}_{:t}) = [u_t(\mathbf{x}_{:t}), v_t(\mathbf{x}_{:t})]$

for all $t \geq 1$ with confidence level $1 - \alpha$ if

$$\mathbb{P}_{\mathbf{X}}\left(p \in I_t(\mathbf{X}_{:t}), \forall t > 1\right) \geq 1 - \alpha$$

for any $p \in [0, 1]$, where $\mathbf{X}$ is an infinite sequence of independent Bernoulli random variables with mean p, and $\mathbf{X}_{:t}$ denotes the first t elements of the sequence $\mathbf{X}$.

We present two ways how to obtain confidence sequences. The first is based on combining confidence intervals using union bound and is easier to analyze, while the second is based on martingale concentration, yields stronger performance, and does not require hyperparameters.

Confidence sequences based on union bound

For any random variable with positive finite variance, the optimal width of the confidence interval for the mean parameter scales as $\sqrt{\log(1/\alpha)/t}$ with the increasing number of samples t at confidence level $1-\alpha$. On the other hand, it is well known that the width of the optimal confidence sequence scales as $\sqrt{(\log(1/\alpha) + \log\log t)/t}$ as t increases due to the law of iterated logarithms. A naive use of union bound – computing a confidence interval using failure probability at time step t, $\alpha_t = \frac{\alpha c}{t^\gamma}$ for some c and $\gamma > 1$ such that $\sum_{i=1}^{\infty} \alpha_t = \alpha$ – yields a confidence sequence whose width scales as $\sqrt{\log(1/\alpha_t)/t} \approx \sqrt{(\log(t) + \log(1/\alpha))/t}$. We cannot choose any monotonous α_t schedule decaying slower because even for $\gamma = 1$ we still keep the log factor while the sum $\sum_{i=1}^{\infty} \delta_t$ diverges.

Now consider non-monotonous schedules of α_t, two key ideas follows. (1) In order to have the optimal rate $\log(1/\alpha_t) \approx \log(1/\alpha) + \log\log t$, we need $\log \alpha_t \asymp \log(\alpha/\log t)$. Clearly, if this holds for all t, then $\sum_{t=1}^{\infty} \alpha_t$ diverges. (2) A confidence interval at time t is also a valid confidence interval for all $t' > t$. Furthermore, if t' is not much larger than t, it may still asymptotically have the optimal width up to a multiplicative constant.

Combining these ideas, if we compute a confidence interval after k samples whenever $k = 2^i$ for some integer $i \geq 1$ at confidence level $1 - \alpha/(i(i+1))$, they will form a confidence sequence at confidence level α of asymptotically optimal width up to a constant factor.

Confidence sequences based on betting

We describe an alternative approach to confidence sequences based on a hypothetical betting game. For the illustration, consider a fair sequential game; e.g., sequentially betting on the outcomes of a coin. If we guess the outcome correctly, we win the staked amount, otherwise we lose it. If the coin is fair, in expectation, our wealth stays the same. On the other hand, if the game is not fair and the coin is biased, we can win money. Thus, if we win lots of money, we can conclude that the game is not fair. We instantiate a betting game for every possible mean $0 \leq p \leq 1$ that would be fair if the true mean is p. We bet on the samples of the random variable and as soon as we win enough money, we drop that particular p from the confidence sequence since it is not likely not the true mean. To make things formal, we introduce the necessary concepts from probability theory. The evolution of our wealth throughout a fair game is modeled

by martingales, sequences of random variables for which, independently of the past, the expected value stays the same.

Definition 6.4 (Martingale)**.** A sequence of random variables $W_1, W_2, \ldots$ is called a *martingale* if for any integer $n > 0$, we have $\mathbb{E}(|W_n|) < \infty$ and $\mathbb{E}(W_{n+1}|W_1, \ldots, W_n) = W_n$.

In the coin-betting example, $W_1, W_2, \ldots$ is a martingale where W_n represents our wealth after playing the game for n rounds. We stress that $W_t \geq 0$ for all $t > 0$. By convention, we will also have $W_1 = 1$. We further need a time-uniform generalization of Markov's inequality.

Proposition 6.5 (Ville's inequality)**.** *Let $W_1, W_2, \ldots$ be a non-negative martingale, then for any real $a > 0$*

$$\mathrm{P}\left[\sup_{n\geq 1} W_n \geq a\right] \leq \frac{\mathbb{E}[W_1]}{a}.$$

Thus, whenever we play a game and earn a lot, we can — with high probability[3] — rule out the possibility that the game is fair. So far, this is still an abstract framework. We still have to design the betting game, the betting strategy, and describe how to run the (uncountably) infinite number of games, one for every $p \in [0, 1]$.

Betting game Let $0 < p < 1$. Consider a coin-betting game where we win $1/p$ (resp. $1/(1-p)$) multiple of the staked amount if we correctly predicted heads (resp. tails). If the underlying heads probability is p, then regardless of our bet - in expectation - we still have the same amount of money; thus, this game is fair. We identify heads and tails with outcomes $1, 0$ respectively.

Betting strategy We deconstruct the betting strategy into the two sub-tasks: (1) If we know the underlying heads probability, we can design the optimal betting strategy for any criterion. (2) Estimate the heads probability. **First sub-task:** Let p define the betting game from the previous paragraph and q be the true heads probability; We bet q-fraction of wealth to heads and $1 - q$ fraction to tails which is known to be optimal due to Wald (1947). **Second sub-task:** we use a "regularized" sample mean and after observing H times heads in a sequence of length T, we estimate $\hat{q} = (H + 0.5)/(T + 1)$.

Parallel betting games We have described a betting game for a certain p and a betting strategy. Employing Ville's inequality we can reject the hypothesis that the true sampling distribution has mean p if the corresponding wealth is high. However, we need to run the game for all values of $p \in [0, 1]$. This is clearly impossible to do explicitly, but it turns out that there is an elegant solution to this.

First, we note that our betting strategy does not depend on p and is only based on the observed outcomes. Now, we express the log-wealth at time T as a function of p.

[3] in the frequentist sense

Let $\hat{q}_t$ (resp. x_t) be our estimate of q (resp. the coin-toss outcome) at time t and $H = \sum_{t=1}^{T} x_t$, then our log-wealth at time T can be written as a function of p:

$$\begin{aligned}\log W_T(p) &= \log \prod_{t=1}^{T} \left(\left(\frac{\hat{q}_t}{p} \right)^{x_t} \left(\frac{1-\hat{q}_t}{1-p} \right)^{1-x_t} \right) \\ &= \underbrace{\sum_{t=1}^{T} x_t \log(\hat{q}_t) + (1-x_t)\log(1-\hat{q}_t)}_{\text{LOGQ}} \underbrace{-H\log(p) - (T-H)\log(1-p)}_{\text{LOGP}(p)} .\end{aligned}$$

Specially, we note that $\log W_T(p)$ is convex. Thus, we can efficiently – using binary search – find the confidence interval in every iteration.

6.4 Conclusion

We have presented the randomized Clopper-Pearson bound and confidence sequences based on union bound and martingale concentration. The paper contains a more formal presentation of the topic and provides analyses of the algorithms and experiments. This work was just published; thus, there are no follow-ups in this direction as of now.

Bibliography

A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Mislav Balunovic and Martin Vechev. Adversarial training and provable defenses: Bridging the gap. In *ICLR*, 2020.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *ECML*, 2013.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. In *NeurIPS*, 2019.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *ICLR*, 2015.

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

M. Hein and M. Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NeurIPS*, 2017.

Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *CAV*, 2017.

A. Kurakin, I. J. Goodfellow, and S. Bengio. Adversarial examples in the physical world. In *ICLR Workshop*, 2017.

Cassidy Laidlaw, Sahil Singla, and Soheil Feizi. Perceptual adversarial robustness: Defense against unseen threat models. In *ICLR*, 2021.

Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy (SP)*, 2019.

Klas Leino, Zifan Wang, and Matt Fredrikson. Globally-robust neural networks. In *ICML*, 2021.

Alexander Levine and Soheil Feizi. Wasserstein smoothing: Certified robustness against wasserstein adversarial attacks. In *AISTATS*, 2020.

Qiyang Li, Saminul Haque, Cem Anil, James Lucas, Roger B Grosse, and Jörn-Henrik Jacobsen. Preventing gradient attenuation in lipschitz constrained convolutional networks. *NeurIPS*, 2019.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

John F. Reiser and Donald E. Knuth. Evading the drift in floating-point addition. *Information Processing Letters*, 3(3):84–87, 1975.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

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 *NeurIPS*, 2019.

Sascha Saralajew, Lars Holdijk, and Thomas Villmann. Fast adversarial robustness certification of nearest prototype classifiers for arbitrary seminorms. In *NeurIPS*, 2020.

Sahil Singla, Surbhi Singla, and Soheil Feizi. Improved deterministic l_2 robustness on CIFAR-10 and CIFAR-100. In *ICLR*, 2022.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020.

Asher Trockman and J Zico Kolter. Orthogonalizing convolutional layers with the cayley transform. In *ICLR*, 2021.

Václav Voráček and Matthias Hein. Provably adversarially robust nearest prototype classifiers. In *ICML*, 2022.

Václav Voráček and Matthias Hein. Improving l1-certified robustness via randomized smoothing by leveraging box constraints. In *ICML*, 2023a.

Václav Voráček and Matthias Hein. Sound randomized smoothing in floating-point arithmetics. In *ICLR*, 2023b.

Václav Voráček. Treatment of statistical estimation problems in randomized smoothing for adversarial robustness. In *NeurIPS*, 2024.

Abraham Wald. Sequential analysis. 1947.

Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. In *NeurIPS*, 2018.

Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *ICML*, 2019.

Greg Yang, Tony Duan, J Edward Hu, Hadi Salman, Ilya Razenshteyn, and Jerry Li. Randomized smoothing of all shapes and sizes. In *ICML*, 2020.

Bohang Zhang, Du Jiang, Di He, and Liwei Wang. Boosting the certified robustness of l-infinity distance nets. In *ICLR*, 2022.

7 Full papers

7.1 Provably Adversarially Robust Nearest Prototype Classifiers

Provably Adversarially Robust Nearest Prototype Classifiers

Václav Voráček [1] **Matthias Hein** [1]

Abstract

Nearest prototype classifiers (NPCs) assign to each input point the label of the nearest prototype with respect to a chosen distance metric. A direct advantage of NPCs is that the decisions are interpretable. Previous work could provide lower bounds on the minimal adversarial perturbation in the ℓ_p-threat model when using the same ℓ_p-distance for the NPCs. In this paper we provide a complete discussion on the complexity when using ℓ_p-distances for decision and ℓ_q-threat models for certification for $p, q \in \{1, 2, \infty\}$. In particular we provide scalable algorithms for the *exact* computation of the minimal adversarial perturbation when using ℓ_2-distance and improved lower bounds in other cases. Using efficient improved lower bounds we train our **P**rovably adversarially robust **NPC** (PNPC), for MNIST which have better ℓ_2-robustness guarantees than neural networks. Additionally, we show up to our knowledge the first certification results w.r.t. to the LPIPS perceptual metric which has been argued to be a more realistic threat model for image classification than ℓ_p-balls. Our PNPC has on CIFAR10 higher certified robust accuracy than the empirical robust accuracy reported in (Laidlaw et al., 2021). The code is available in our repository.

1. Introduction

The vulnerability of neural networks against adversarial manipulations (Szegedy et al., 2014; Goodfellow et al., 2015) is a major problem for their real world deployment in safety critical systems such as autonomous driving and medical applications. However, the problem is not restricted to neural networks as it has been shown that basically all machine learning algorithms are vulnerable to adversarial perturbations e.g. nearest neighbor methods (NN) (Wang et al., 2018), kernel SVMs (Xu et al., 2009; Biggio et al., 2013; Russu et al., 2016; Hein & Andriushchenko, 2017), decision trees (Papernot et al., 2016; Bertsimas et al., 2018; Chen et al., 2019; Andriushchenko & Hein, 2019). In the area of neural networks this lead to an arm's race between novel empirical defenses and attacks and even initially promising defenses were broken later on (Athalye et al., 2018). This still happens for papers published at top machine learning conferences (Tramer et al., 2020; Croce & Hein, 2020a) despite more reliable attacks for adversarial robustness evaluation (Croce & Hein, 2020b) and guidelines (Carlini et al., 2019) being available.

Thus classifiers with provable adversarial robustness guarantees are highly desirable. For neural networks computation of the exact minimal perturbation turns out to be restricted to very small networks (Tjeng & Tedrake, 2017). Instead one derives either deterministic (Hein & Andriushchenko, 2017; Wong & Kolter, 2018; Gowal et al., 2018; Mirman et al., 2018; Zhang et al., 2020; Lee et al., 2020; Huang et al., 2021; Leino et al., 2021) or probabilistic guarantees (Cohen et al., 2019; Jeong et al., 2021) on the robust accuracy. We refer to (Li et al., 2020) for a recent overview. While provable adversarial robustness has been studied extensively for neural networks, the literature for standard classifiers is scarce, e.g. decision trees (Bertsimas et al., 2018), boosted decision stumps and trees (Chen et al., 2019; Andriushchenko & Hein, 2019), and nearest neighbour (Wang et al., 2018; 2019) and nearest prototype classifiers (NPC) (Saralajew et al., 2020). NPC are also known as *Learning Vector Quantization (LVQ)*, see (Kohonen, 1995), and are directly interpretable, can be used for all data where a distance function is available and have the advantage compared to a nearest neighbour classifier that the prototypes can be learned and thus they are more efficient and achieve typically better generalization performance. Moreover, NPC have a maximum margin nature (Crammer et al., 2003) and (Saralajew et al., 2020) showed recently how to derive lower bounds on the minimal adversarial perturbation which in turn yield lower bounds on the robust accuracy. (Wang et al., 2019) have shown how to compute the minimal adversarial perturbation for nearest neighbor classifiers using the ℓ_2-distance which applies to NPC as well.

Contributions: we show that the results of (Saralajew et al., 2020) can be improved in various ways leading to our PNPC

[1]University of Tübingen, Germany. Correspondence to: Václav Voráček <vaclav.voracek@uni-tuebingen.de>, Matthias Hein <matthias.hein@uni-tuebingen.de>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

which perform better both in clean and robust accuracy.

A) We generalize the lower bounds on the minimal adversarial perturbation (Saralajew et al., 2020) provided for distances induced by semi-norms to general semi-metrics, thus improving significantly over standard ℓ_p-based certification. The original proof of (Saralajew et al., 2020) used the absolute homogenity of semi-norms; thus, it do not generalize to semi-metrics.

B) For NPC using the ℓ_2-distance we show that the lower bounds of (Wang et al., 2019) can be quickly evaluated so that training with them is feasible and show that these bounds improve the ones of (Saralajew et al., 2020). Moreover, we improve the certification of (Wang et al., 2019) by integrating that the domain in image classification is $[0,1]^d$. For MNIST our ℓ_2-PNPC has the best ℓ_2-robust accuracy even outperforming randomized smoothing for large radii. Moreover, we show how to certify exactly ℓ_1- and ℓ_∞-robustness for ℓ_2-NPC and in this way can certify multiple-norm robustness and show that our ℓ_2-PNPC outperforms the multiple-norm robustness guarantees of (Croce & Hein, 2020a).

C) For the ℓ_1-and ℓ_∞-NPC we provide novel lower bounds and analyze their complexity. For ℓ_∞-NPCs we thus improve over the bounds given in (Saralajew et al., 2020).

D) As the ℓ_p-distances are not suited for image classification tasks, we use a neural perceptual metric (LPIPS) (Zhang et al., 2018) as a semi-metric for the NPC and provide robustness guarantees in the perceptual metric. We improve both in terms of clean and certified robust accuracy over the clean and empirical robust accuracy of the adversarially trained ResNet 50 of (Laidlaw et al., 2021)

2. Provably Robust NPC Classifiers

Nearest prototype classifiers require for a given input space $\mathcal{X}$ only a (semi)-metric. To compare with previous work, we introduce also a (semi)-norm, which requires a vector-space structure; thus, assuming the existence of a norm is a stronger assumption than the assumption of the existence of a metric.

Definition 2.1. A mapping $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a semi-metric if the following properties holds for any $x, y, z \in \mathcal{X}$:

- $d(x, y) \geq 0$ (non-negativity)
- $d(x, y) = d(y, x)$ (symmetry)
- $d(x, y) \leq d(x, z) + d(z, y)$ (triangle inequality)

If we further require that $d(x, y) = 0 \implies x = y$, then the semi-metric becomes a metric.

Definition 2.2. A mapping $\|\cdot\| : \mathcal{X} \to \mathbb{R}$ is a semi-norm if the following properties holds for any $x, y \in \mathcal{X}$, $\alpha \in \mathbb{R}$:

- $\|x\| \geq 0$ (non-negativity)
- $\|\alpha x\| = |\alpha| \, \|x\|$ (absolute homogeneity)
- $\|x + y\| \leq \|x\| + \|y\|$ (triangle inequality)

If we further require $\|x\| = 0 \implies x = \mathbf{0}$, then the semi-norm becomes a norm.

Note that any (semi)-norm $\|x\|$ induces a (semi)-metric d with $d(x, y) = \|x - y\|$.

We denote by $(w_i)_I$ the set of prototypes. Each prototype is assigned to one class. Then $z \in \mathbb{R}^d$ is classified as

$$f(z) = \operatorname*{arg\,min}_{y=1,\dots,K} \min_{i \in I_y} d(z, w_i),$$

where I_y are the prototypes of class y. A nearest neighbor classifier (1NN) can also be understood as NPC where one uses the training set as prototypes and thus are not learned. However, by training prototypes one can achieve better classification performance, and also robustness, see Table 5, with less prototypes meaning that NPC are significantly more efficient than 1NN. We note that the classification for a point z with label y is correct if

$$\min_{i \in I_y} d(z, w_i) - \min_{j \in I_y^c} d(z, w_j) < 0,$$

where I_y^c is the set of all prototypes not belonging to class y (the complement of I_y in I).

2.1. Provable robustness guarantees for semi-metrics

Next we define the minimal adversarial perturbation of a point z for a semi-metric on $\mathcal{X}$, that is the radius r of the smallest ball $B_d(z, r) = \{x \in \mathcal{X} \mid d(x, z) \leq r\}$ around z such at least one point in $B_d(z, r)$ is classified differently than is z. If a point z is misclassified then we define the minimal adversarial perturbation to be zero. We assume that there is a non-empty set of prototypes for every class; thus, there always exists an adversarial example.

Definition 2.3. The **minimal adversarial perturbation** $\epsilon_d(z)$ of $z \in \mathcal{X}$ of a NPC using semi-metric d is defined as

$$\epsilon_d(z) = \min\{r \mid \max_{x \in B_d(z,r)} \big(\min_{i \in I_y} d(x, w_i) - \min_{j \in I_y^c} d(x, w_j) \big) \geq 0\}$$

If $\min_{i \in I_y} d(z, w_i) - \min_{j \in I_y^c} d(z, w_j) \geq 0$ then we set $\epsilon_d(z) = 0$.

In (Saralajew et al., 2020) they derive for semi-norms a lower bound on ϵ_d and in this way get robustness certificates. We generalize this lower bound to semi-metrics which is considerably more general as $\mathcal{X}$ need not be a vector space. It turns out that the only necessary technical requirement for the proof is the triangle inequality. This is unlike (Saralajew et al., 2020), where the proof also required the absolute homogeneity of semi-norms.

Theorem 2.4. *Let $(\mathcal{X}, d)$ be a semi-metric space, then it holds for the minimal adversarial perturbation $\epsilon_d(z)$ of $z \in \mathcal{X}$ with correct label y:*

$$\epsilon_d(z) \geq \max\left\{0, \frac{\min_{j \in I_y^c} d(z, w_j) - \min_{i \in I_y} d(z, w_i)}{2}\right\}.$$

We note that if the semi-metric d can be written as $d(x, y) = \|x - y\|$ for some semi-norm $\|\cdot\|$, then our bound is equal to the one given in (Saralajew et al., 2020)

2.2. The minimal adversarial ℓ_q-perturbation of the ℓ_p-NPC and lower bounds

In this section we derive the minimal adversarial ℓ_q-perturbation for the ℓ_p-PNPC in $\mathbb{R}^d$ where our main interest is $p, q \in \{1, 2, \infty\}$. In contrast to the semi-metric case, here we treat the case where the ℓ_q-metric measuring the size of the adversarial perturbation is different from the ℓ_p-metric used in the NPC. In this section we use the notation

$$B_q(x, r) = \{z \in \mathbb{R}^d \mid \|z - x\|_q \leq r\}.$$

Thus we first define

Definition 2.5. The **minimal adversarial perturbation** $\epsilon_p^q(z)$ of $x \in \mathcal{X} \subset \mathbb{R}^d$ with respect to the ℓ_q-metric for the ℓ_p-NPC is defined as:

$$\begin{aligned} \epsilon_p^q(z)_j = \min_{r \in \mathbb{R}, x \in \mathcal{X}} \quad & r \\ \text{sbj. to:} \quad & \|x - w_i\|_p - \|x - w_j\|_p \geq 0 \\ & x \in B_q(z, r) \end{aligned}$$

If $\min_{i \in I_y} \|x - w_i\|_p - \min_{j \in I_y^c} \|x - w_j\|_p > 0$ we set $\epsilon_p^q(z) = 0$.

The following reformulation of the optimization problem for the computation of the minimal adversarial perturbation $\epsilon_p^q(z)$ allows us to provide a generic and direct way to derive efficiently computable lower bounds on $\epsilon_p^q(z)$. Note that in the following we always integrate the constraint $x \in \mathcal{X}$ as we will see that this significantly improves the guarantees, e.g. when $\mathcal{X} = [0, 1]^d$ in image classification, compared to $\mathcal{X} = \mathbb{R}^d$ as done in (Saralajew et al., 2020; Wang et al., 2019).

Theorem 2.6 (Exact computation of $\epsilon_p^q(z)$)**.** *Let $z \in \mathcal{X} \subset \mathbb{R}^d$ and denote by I_y the index set of prototypes (w_j) of class y and by I_y^c its complement (the index set of prototypes not belonging to class y). Then define for every $j \in I_y^c$:*

$$\begin{aligned} r_p^q(z)_j = \min_{x \in \mathbb{R}^d} \quad & \|x - z\|_q \\ \textit{sbj. to:} \quad & \|x - w_i\|_p - \|x - w_j\|_p \geq 0 \quad \forall\, i \in I_y \\ & x \in \mathcal{X} \end{aligned} \tag{1}$$

Then $\epsilon_p^q(z) = \min_{j \in I_y^c} r_p^q(z)_j$.

ℓ_p-distance \ ℓ_q-threat model	ℓ_1	ℓ_2	ℓ_∞
ℓ_1	NP-hard	NP-hard	$O(d \log(d))$
ℓ_2	$\Theta(d)$	$\Theta(d)$	$\Theta(d)$
ℓ_∞	$\Theta(d)$	$O(d \log(d))$	$\Theta(d)$

Table 1: Computational complexity of $\rho_p^q(z)_{i,j}$.

ℓ_p-distance \ ℓ_q-threat model	ℓ_1	ℓ_2	ℓ_∞
ℓ_1	NP-hard	NP-hard	Poly
ℓ_2	Poly	Poly	Poly
ℓ_∞	NP-hard	NP-hard	NP-hard

Table 2: Computational Complexity of $r_p^q(z)$ and $\epsilon_p^q(z)$.

While the corresponding optimization problems are often non-convex, we will see in the following that they are equivalent to convex optimization problems in the case where the ℓ_2-distance is used in the NPC ($p = 2$). Using the formulation of the exact problem as an optimization problem we can now simply derive lower bounds on $\epsilon_p^q(z)$ by relaxing the optimization problem (1).

We consider for this reason the following optimization problems. For $i \in I_y$ and $j \in I_y^c$ we define:

$$\begin{aligned} \rho_p^q(z)_{i,j} = \min_{x \in \mathbb{R}^d} \quad & \|x - z\|_q \\ \text{sbj. to:} \quad & \|x - w_i\|_p - \|x - w_j\|_p \geq 0 \\ & x \in \mathcal{X} \end{aligned} \tag{2}$$

In Theorem 2.7 we show that these simpler problems can often be solved efficiently, although the computation of ϵ_p^q is often intractable, as we show in Theorem 2.8.

Theorem 2.7. *The computational complexities of optimization problems $\rho_p^q(z)_{i,j}$ for $p, q \in \{1, 2, \infty\}$ for $\mathcal{X} = \mathbb{R}^d$ are summarized in Table 1.*

Theorem 2.8. *The computational complexities of optimization problems $r_p^q(z)_j$ in (1) for $p, q \in \{1, 2, \infty\}$ and $\mathcal{X} = [0, 1]^d$ are summarized in Table 2.*

Apart from the known ℓ_2-case (see (Wang et al., 2019)) we show that also ℓ_1-NPC can be certified efficiently for the ℓ_∞-threat model. Because of this theorem it is even more important that at least for the ℓ_∞-NPCs efficient lower bounds are available for all threat models in $q = \{1, 2, \infty\}$. We note that the optimization problem for $r_2^q(z)_j$ in (1) is equivalent to a quadratic program for $q = 2$ and to a linear programs for $q \in \{1, \infty\}$ for both with and without box constraints.

The following lemma shows that (2) can be used to get a lower bound on the minimal adversarial perturbation, and

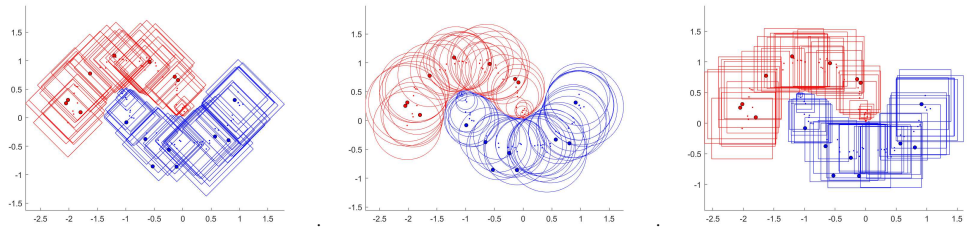


Figure 1: **Illustration of the ℓ_q-minimal adversarial perturbations of a ℓ_2-NPC for a binary classification problem.** The learned prototypes are shown as the larger red resp. blue dots. For each data point we draw the largest ℓ_1-(left), ℓ_2-(middle) and ℓ_∞-(right) ball which is fully classified as the same class. The radii are computed using Alg. 1. Though there is no specific optimization for multiple-norm robustness, ℓ_2-NPC possess non-trivial multiple-norm robustness.

subsequently we show that it improves on the previous bound given in Theorem 2.4 which has been derived by (Saralajew et al., 2020). In particular, this bound can be tight and we show in Table 4 in Section 5 that this happens frequently in practice and thus allows to avoid the significantly more complex problems in (1).

Lemma 2.9. *It holds*

$$\epsilon_p^q(z) \geq \min_{j \in I_c^y} \max_{i \in I_y} \rho_p^q(z)_{i,j}.$$

Moreover, let (j^*, i^*) be the prototype pair in $I_c^y \times I_y$ which realizes the lower bound and denote by x^* the minimizer of $\rho_p^q(z)_{i^*,j^*}$. Then if x^* fulfills

$$\|x^* - w_i\|_p - \|x^* - w_{j^*}\|_p \geq 0 \quad \forall i \in I_y,$$

then $\epsilon_q^p(z) = \min_{j \in I_c^y} \max_{i \in I_y} \rho_p^q(z)_{i,j}$.

Theorem 2.10. *The lower bound on $\epsilon_p^p(z)$ of Lemma 2.9 is at least as good as the one of Theorem 2.4. That is,*

$$\begin{aligned}\min_{j \in I_c^y} \max_{i \in I_y} \rho_p^p(z)_{i,j} &\geq \min_{j \in I_c^y} \rho_p^p(z)_{i^*,j} \\ &\geq \max\left\{0, \frac{\min_{j \in I_y^c} \|z - w_j\|_p - \min_{i \in I_y} \|z - w_i\|_p}{2}\right\},\end{aligned}$$

where $i^* \in \arg\min_{i \in I_y} \|z - w_i\|_p$.

In order to be able to use these lower bounds for certified training of our PNPC, their efficient computation is of high importance which we discuss next.

For better intuition we discuss some cases in more detail. The ℓ_2-NPC have a nice geometric descriptions as the set

$$\begin{aligned}&\{z \,|\, \|z - w_i\|_2 = \|z - w_j\|_2\} \\ =&\{z \,|\, \langle w_j - w_i, z\rangle + \frac{\|w_i\|_2^2 - \|w_j\|_2^2}{2} = 0\}\end{aligned}$$

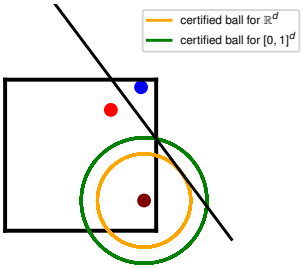


Figure 2: Illustration for ℓ_2-NPC for two prototypes (red and blue): when taking into account that the data lies in $[0,1]^d$ we can certify a larger ball than in $\mathbb{R}^d$.

is a hyperplane. Thus the computation of $\rho_2^q(z)_{i,j}$ for $\mathcal{X} = \mathbb{R}^d$ corresponds to the computation of the ℓ_q-distance of a point to a hyperplane:

$$\rho_2^q(z)_{i,j} = \frac{\|z - w_j\|_2^2 - \|z - w_i\|_2^2}{2\,\|w_i - w_j\|_{q^*}},$$

where q^* denotes the dual norm of q. This has also been derived in (Wang et al., 2019). As illustration how the constraints $\mathcal{X} = [0,1]^d$, e.g. in image classification, improve the certificates, we show in Figure 2 the ball which can be certified in $\mathbb{R}^d$ resp. $[0,1]^d$.

2.3. How to do the certification efficiently

Table 1 shows that $\rho_p^q(z)_{i,j}$ can be computed efficiently or even given in closed form except for the two cases when $(p,q) \in \{(1,1),(1,2)\}$. However, that would still mean

that the lower bound of Lemma 2.9

$$\epsilon_p^q(z) \geq \min_{i \in I_y} \max_{j \in I_y^c} \rho_p^q(z)_{i,j},$$

would require us to solve naively $|I_y||I_y^c|$ such problems. Seemingly, the bound in Theorem 2.4 is much cheaper as it requires only $(|I_y| + |I_y^c|)$ operations even though one has to note that the bound only exists for the case when $p = q$.

i) A lower bound: Theorem 2.10 shows that when fixing $i^* = \arg\min_{i \in I_y} \|z - w_i\|$ and then computing

$$\min_{j \in I_c^y} \rho_p^q(z)_{i^*,j},$$

yields by Lemma 2.9 a lower bound on $\epsilon_p^q(z)$. By Theorem 2.10 this lower bound is for the case $p = q$ still better than the one of Theorem 2.4 while having the same complexity of $|I_y| + |I_y^c|$ operations. Obviously, when integrating box constraints, that is $\mathcal{X} = [0,1]^d$, the gap can only become larger between the two bounds.

ii) Using simpler lower bounds: When certifying bounds for $\mathcal{X} = [0,1]^d$ we first compute the lower bounds for $\mathcal{X} = \mathbb{R}^d$ as they are often available in closed form and are definitely lower bounds for the more restricted case $\mathcal{X} = [0,1]^d$. By fixing again i^* we can then use $s_j := \rho_p^q(z)_{i^*,j}$ and define the minimum and minimizer as $(\lambda, j^*) = \min_{j \in I_y^c} \rho_p^q(z)_{i^*,j}$. Now, let us denote by $\kappa_p^q(z)_{i^*,j}$ the corresponding quantity when using $\mathcal{X} = [0,1]^d$ instead of $\mathcal{X} = \mathbb{R}^d$. Then we only need to compute $\kappa_p^q(z)_{i^*,j}$ if $s_j < \kappa_p^q(z)_{i^*,j^*}$, which is typically satisfied for very few instances, so most computations are pruned.

iii) Dual problems: as in (Wang et al., 2019) we use the dual problems when computing $r_2^q(z)_j$. This has three advantages. First, we always get a lower bound using weak duality, second, we stop solving $r_p^q(z)_j$ when the dual value is higher than our currently smallest upper bound and third; empirically only few constraints of the problems become active; thus, the solutions are dual-sparse.

Final Certification: in Algorithm 1 we sketch the certification process. It does not include all details (see above) which we use for speeding up the computation of lower bounds as well as the exact minimal adversarial perturbation.

Algorithm 1 Sketch of certification algorithm for correctly classified point z

// Computation of λ as lower bound on $\epsilon_p^q(z)$
$i^* = \arg\min_{i \in I_y} \|z - w_i\|_p$
$s_j = \rho_p^q(z)_{i^*,j}$, $j \in I_y^c$ (s_j lower bounds $r_p^q(z)_j$)
$(\lambda, j^*) = \min_{j \in I_y^c} \rho_p^q(z)_{i^*,j}$
if minimizer x^* of $\rho_p^q(z)_{i^*,j^*}$ is feasible for $r_p^q(z)_{j^*}$ **then**
  $\epsilon_p^q(z) = \lambda$ and return
else
  λ is lower bound on $\epsilon_q^p(z)$
end if
// Computation of $\epsilon_p^q(z)$ ($p = 2$ or $(p,q) = (1,\infty)$)
$\mu = r_p^q(z)_{j^*}$ // (it holds $\mu \geq \epsilon_p^q(z)$)
for $j = 1$ **to** $|I_y^c|$ **do**
  if $s_j < \mu$ **then**
    compute $r_p^q(z)_j$
    if $r_p^q(z)_j < \mu$ **then**
      $\mu = r_p^q(z)_j$
    end if
  end if
  $\epsilon_p^q(z) = \mu$
end for

3. Perceptual Metric

The hypothesis underlying the goal of adversarial robustness is that images which have the same semantic content, should be classified the same (with the exception at the true decision boundary). However, this would require a human oracle which judges if the semantic content is similar. A proxy is the typical ℓ_p-threat model, where for suitable chosen radius ϵ_p one expects that for a given image x also $B_p(x, \epsilon_p)$ should be classified the same as for humans the resulting images are (semantically) indistinguishable from the original image. However, it is well known that pixel-based ℓ_p-distances are not a good measure of image similarity. A huge literature in computer vision discusses the construction of metrics which better correspond to human perception of similarity of images e.g. the SSIM metric of (Wang et al., 2004). More recently, neural perceptual metrics, such as the LPIPS distance, have been proposed in (Zhang et al., 2018). The LPIPS distance is based on a feature mapping of a fixed neural network and has been shown to correlate better with human perception (Zhang et al., 2018; Laidlaw et al., 2021). In (Laidlaw et al., 2021) it has been used as threat model in adversarial training. Moreover, (Kireev et al., 2021) have shown that the LPIPS distance better correlates with the severity level of common corruptions than the ℓ_2-distance. Moreover, ℓ_p-distance based NPC are not competitive for CIFAR10. These two aspects motivate us to investigate the perceptual metric-based PNPC as well as novel techniques for the certification in the LPIPS-threat model.

The perceptual metric: Given the output $g^{(l)}(x) \in \mathbb{R}^{H_l \times W_l \times C_l}$ of the l-th layer of a fixed neural network (we use Alexnet as suggested by (Zhang et al., 2018)) of height H_l and width W_l and channels C_l, we define the normalized output of a layer as $\hat{g}_{h,w}^{(l)}(x) = \frac{g_{h,w}^{(l)}(x)}{\left\| g_{h,w}^{(l)}(x) \right\|_2}$. The LPIPS distance d is then defined in (Zhang et al., 2018) as

$$d^2(x,y) = \sum_{l \in I_L} \frac{1}{H_l W_l} \sum_{h,w} \left\| w_l \odot \left(\hat{g}_{h,w}^{(l)}(x) - \hat{g}_{h,w}^{(l)}(y) \right) \right\|_2^2,$$

where the weights w_l are learned using human perception data and I_L is the index set of layers used for the metric. We follow (Laidlaw et al., 2021) and use the unweighted (i.e., weights perform an identity mapping) version in order to be able to directly compare to them. However, it would be easy to adapt our approach for the weighted version. We define the embedding, $\phi : [0,1]^d \to \mathbb{R}^D$

$$x \mapsto \phi(x) = \left(\frac{\hat{g}^{(l)}}{\sqrt{H_l W_l}} \right)_{l \in I_L}, \tag{3}$$

so that the unweighted LPIPS distance can simply be written as a standard Euclidean distance $d(x,y) = \|\phi(x) - \phi(y)\|_2$ in the embedding space.

The mapped image space $\phi(I)$ of all natural images I is a subset of $\phi([0,1]^d)$, which can be seen as an at most d-dimensional continuous "submanifold" of the embedding space $\mathbb{R}^D$. Thus for all points $z \in \mathbb{R}^D \backslash \phi([0,1]^d)$ there exists no pre-image in $[0,1]^d$. However, the Euclidean distance between every mapped images $x, y \in I$ corresponds to the perceptual distance between them. Thus we train our PNPC in the embedding space $\mathbb{R}^D$ and certify it with respect to the Euclidean distance which in turn yields guarantees with respect to the LPIPS distance.

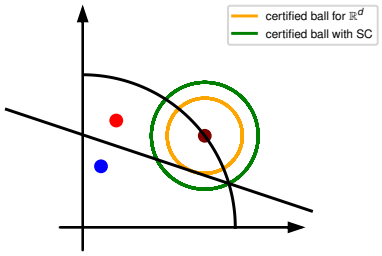


Figure 3: The embedded data $\phi(x)$ lies on the intersection of the positive orthant and the sphere (shown in black). In the embedding space the ℓ_2-metric corresponds to the perceptual metric. Taking these non-negative spherical constraints (SC) into account we can certify a much larger ball than using only the standard certification in $\mathbb{R}^d$.

3.1. Certification in the Perceptual threat model

Up to our knowledge this is the first paper showing results for certification with respect to this threat model aligned with human vision. We can use all techniques we have discussed in Section 2 as we are working with a Euclidean distance in $\mathbb{R}^D$. However, we have more knowledge about $\phi([0,1]^d)$ as the output of each layer is normalized so that $\phi(x)$ lies on a product of spheres with radius $r_l = \frac{1}{\sqrt{H_l W_l}}$ as

$$\left\| \phi^{(l)}_{h,w}(x) \right\|_2 = \left\| \frac{\hat{g}^{(l)}_{h,w}}{\sqrt{H_l W_l}} \right\|_2 = \frac{1}{\sqrt{H_l W_l}} := r_l, \tag{4}$$

for any $l \in I_L, h \in I_H, w \in I_W$. Additionally, we know due to the structure of Alexnet that $\phi_l(x)$ is non-negative for all layers, see Figure 3 for an illustration. While we can integrate some of the properties of the mapping ϕ into the certification, it is computationally intractable to use as constraint $x \in \phi([0,1]^d)$. Thus our certification works on an overapproximation of $\phi([0,1]^d)$ and thus yields lower bounds on the minimal adversarial perceptual distance.

Basically, we can write our constraints in $\mathbb{R}^D$ as

$$\begin{aligned} \mathcal{X} =& \mathcal{X}_1 \times \cdots \times \mathcal{X}_L \\ \mathcal{X}_l =& \left(\frac{1}{\sqrt{H_l W_l}} S^{c_l} \cap [0,\infty)^{c_l} \right)^{H_l W_l}, \ l = 1, \dots, L, \end{aligned} \tag{5}$$

where c_l is the number of channels in layer l of the output of the layer l and $D = \sum_{l=1}^{L} H_l W_l c_l$. We use upper indices (e.g., $x^{(h,w,l)}$) to denote slice of vector x which corresponds to vector of channels at position h, w in layer l. The constants r_l for $1 \le l \le L$ were defined in (4).

As we use ℓ_2-NPC we have to compute:

$$\begin{aligned} \rho(z)_{i,j} = \min_{x \in \mathbb{R}^D} \quad & \|x - z\|_2 \\ \text{sbj. to:} \quad & \langle x, w_j - w_i \rangle + \frac{\|w_i\|_2^2 - \|w_j\|_2^2}{2} \ge 0 \\ & \left\| x^{(h,w,l)} \right\|_2^2 = r_l^2, \ l = 1, \dots, L \\ & \qquad\qquad\qquad\quad h = 1, \dots, H_l \\ & \qquad\qquad\qquad\quad w = 1, \dots, W_l \\ & x_d \ge 0, \qquad\qquad d = 1, \dots, D. \end{aligned} \tag{6}$$

Despite this problem is non-convex due to the quadratic equality constraints we can derive a convex dual problem (we derive it for an equivalent problem) which is sufficient to provide us with lower bounds using weak duality.

Proposition 3.1. *Define* $v = w_j - w_i$ *and* $b = \frac{\|w_i\|_2^2 - \|w_j\|_2^2}{2}$. *A lower bound on the optimal value of the optimization problem* (6) *is given by*

$$\sqrt{2L + 2 \left(\max_{\lambda \ge 0} - \sum_{h,w,l} \left\| \left(z^{(h,w,l)} - \lambda v^{(h,w,l)} \right)^+ \right\|_2 r_l + \lambda b \right)}$$

which can be efficiently computed using bisection. The summation $\sum_{h,w,l}$ *is a shortcut for* $\sum_{1 \le l \le L} \sum_{1 \le h \le H_l} \sum_{1 \le w \le W_l}$.

In the experimental results in Figure 4 one can clearly see that using this lower bound improves significantly over the standard lower bound of Lemma 2.9.

4. Efficient Training of PNPC

In this section we describe the training procedure for our PNPC. The key advantage compared to the work of (Saralajew et al., 2020) is that despite our lower bounds, see Theorem 2.10, are better and often tight, they can be computed with the same time complexity as theirs if $p \in \{2, \infty\}$. Thus we can do efficient certified training. As objective we use the capped sum of the lower bounds:

$$\max_{(w_i)_{i\in I}} \frac{1}{n} \sum_{r=1}^{n} \min \left\{ \min_{j \in I_y^c} \max_{i \in I_y} \rho_p^q(z_r)_{i,j}, R \right\},$$

where we recall the definition of ρ_p^q from 2:

$$\begin{aligned} \rho_p^q(z)_{i,j} = \min_{x \in \mathbb{R}^d} \quad & \|x - z\|_q \\ \text{sbj. to:} \quad & \|x - w_i\|_p - \|x - w_j\|_p \geq 0 \\ & x \in \mathcal{X} \end{aligned} \tag{7}$$

and R is an upper bound on the margin we want to enforce. The cap is introduced in order to avoid that single training points have excessive margin at the price of many others having small margin; in turn, it is equivalent to minimizing hinge-loss. The loss is minimized via stochastic gradient descent resp. ADAM with large batch sizes. Note further that, for misclassified points we use a signed version of $\rho_p^q(z_r)_{i,j}$ by flipping the constraint in (2) and using $-\rho_p^q(z_r)_{j,i}$ instead, which can be interpreted as signed distance to the decision boundary. Doing this has the advantage that we get gradient information from all points. Maximizing our objective has a direct interpretation in terms of maximizing robust accuracy or more precisely the area under the robustness curve capped at radius R. This is in contrast to (Saralajew et al., 2020) who use as loss their lower bound divided by the sum of the distances where this interpretation is due to the rescaling not applicable.

5. Experiments

The code for experiments is available in our repository[1] where we also provide the training details. We first evaluate the improvements in the certification of better lower bounds resp. exact computation compared to the ones of (Saralajew et al., 2020) as well as (Wang et al., 2019). In a second set of experiments we compare our ℓ_p-PNPC to the ℓ_p-NPC of (Saralajew et al., 2020) resp. to nearest neighbor classification as well as deterministic and probabilistic certification techniques for neural networks on MNIST and CIFAR10 (see App. H). Finally, we discuss our NPC using the perceptual metric and its certification where there is no competitor as up to our knowledge this is the first paper providing robustness certificates. The training time is about a few hours on a laptop.

[1] https://github.com/vvoracek/Provably-Adversarially-Robust-Nearest-Prototype-Classifiers.

Table 3: **Lower bounds on $\epsilon_p^q(z)$.** Mean of the lower bounds of (Saralajew et al., 2020) (Theorem 2.4), the lower bounds of (Wang et al., 2019)) in (13) ($\mathcal{X} = \mathbb{R}^d$), our lower bounds integrating $\mathcal{X} = [0,1]^d$ and the exact radius on the test set for ℓ_2-NPC for ℓ_1-,ℓ_2- and ℓ_∞-threat model.

Model	Num. Proto.	Threat model	Lower bounds Th. 2.2 $\mathbb{R}^d$	Th. 2.6 $\mathbb{R}^d$	Th. 2.6 $[0,1]^d$	Exact radius $[0,1]^d$
ℓ_2-PNPC	4000	ℓ_1	-	9.71	11.77	12.11
MNIST		ℓ_2	0.39	1.86	1.96	1.99
		ℓ_∞	-	0.14	0.16	0.17

Comparison of our lower bounds: One of the major contributions of this paper are our efficient lower bounds on the minimal adversarial perturbation $\epsilon_p^q(z)$. They can be computed so fast that it is feasible to use them during training. We show in Table 3 that our ℓ_q-bounds improve significantly over the ones of (Saralajew et al., 2020) (Th. 2.4, $\mathcal{X} = \mathbb{R}^d$), which only work if $p = q$ and (Wang et al., 2019) (Lemma 2.9, $\mathcal{X} = \mathbb{R}^d$, see (13) for $p = 2$) as we are the only ones who integrate box constraints (Lemma 2.9, $\mathcal{X} = [0,1]^d$). In Table 3, we show that for the ℓ_1-, ℓ_2- and ℓ_∞ threat models our lower bounds are very close to the exact values. The computation of these lower bounds takes for the **full** test set of MNIST: ℓ_1: 188s, ℓ_2: 33s, ℓ_∞: 131s. This is two orders of magnitude faster than the computation of the exact bounds in Table 4. For our ℓ_∞-NPC and ℓ_∞-threat model we get mean lower bounds of 0.3545 for (Saralajew et al., 2020), 0.3560 for the ones from (15) with $\mathcal{X} = \mathbb{R}^d$, and 0.3616 for ours from Lemma 2.9 with $\mathcal{X} = [0,1]^d$ in (20). Here the differences are smaller than for the ℓ_2-NPC.

Time for certification: The computation of the exact minimal adversarial perturbation is only feasible for relatively small neural networks (Tjeng & Tedrake, 2017) and for ensemble of decision trees (Kantchelian et al., 2016). Both use mixed-integer formulations which do not scale well. For boosted decision stumps one can compute the exact robust accuracy (Andriushchenko & Hein, 2019). However, the computation of the exact robust accuracy is already considerably easier than the minimal adversarial perturbation. For ℓ_2-NPC we can compute the exact adversarial perturbation for the ℓ_1-, ℓ_2-, and ℓ_∞-threat model. In Table 4 we report the certification time per point and other statistics for our ℓ_2-PNPC prototypes on MNIST with 400 prototypes per class (ppc) and the ℓ_2-GLVQ -model of (Saralajew et al., 2020) on CIFAR10 with 128 ppc. We can also produce weaker

Table 4: **Time/Statistics for exact minimal adversarial perturbation** for ℓ_2-NPC

Model	Num. Proto.	Threat model	Direct solved	Total QP/LP	QP/LP per pt	Cert. Time per pt
ℓ_2-PNPC MNIST	4000	ℓ_1	4261 (43.8%)	10195	1.86	0.54s
		ℓ_2	3170 (32.6%)	11630	1.77	0.49s
		ℓ_∞	2073 (21.3%)	21081	2.75	1.3s
ℓ_2-GLVQ CIFAR10	1280	ℓ_1	3683 (75.8%)	1817	1.54	0.76s
		ℓ_2	3546 (73.0%)	1777	1.35	0.25s
		ℓ_∞	3511 (72.2%)	1933	1.43	0.9s

certificates faster. For instance, using Lemma 2.9, we can certify MNIST robust accuracy 67% in under $2s$ instead of the exact 73% reported in Table 5.

Regarding the model of ℓ_2-GLVQ on CIFAR10, we have an accuracy of 48.6% (which corresponds to 4859 correctly classified test points). Of these ones we can solve between 72.2% for ℓ_∞ and 75.8% for ℓ_1 directly using Lemma 2.9 by checking the condition after the computation of the lower bounds. This shows the usefulness of Lemma 2.9 as it avoids a lot of QPs (ℓ_2) rsp. LPs (ℓ_1, ℓ_∞) to be solved. Next we see that the number of LPs/QPs needed to be solved per point is less than 1.43 which has to be compared to the worst case of $|I_y^c| = 1152$. This shows that our prior reduction using our tight lower bounds integrating box constraints helps to significantly reduce the number of problems $r_p^q(z)_j$ which need to be solved. In total we get certification times between $0.25s$ (ℓ_2) and $0.9s$ (ℓ_∞) per point which allows us to do the exact certification for all three threat models.

Evaluation of our NPC: We report certified robust accuracy (CRA) and upper bounds on robust accuracy (URA), e.g. computed via an adversarial attack, on MNIST and CIFAR10 (in App. H) for PNPC and the GLVQ of (Saralajew et al., 2020). For ℓ_2-NPC CRA and URA are equal as we compute exact adversarial perturbations. As an interesting baseline, we report results for the one nearest neighbor classifier (1NN). Additionally, we compare to deterministic and probabilistic certification techniques of neural networks.

MNIST - ℓ_2-NPC: In Table 5 we show the results for the **ℓ_2-threat model** on MNIST. Our ℓ_2-PNPC outperforms the ℓ_2-GLVQ for all ϵ_2. The values for ϵ_2 were chosen according to the neural network literature. Note that our ℓ_2-PNPC outperforms all deterministic methods: GlobRob (Leino et al., 2021), OrthConv (Singla et al., 2022), LocLip (Huang et al., 2021), BCP (Lee et al., 2020) and CAP (Wong et al., 2018) in terms of certified robust accuracy and often in the terms of clean accuracy. For the details on comparison with orthogonal convolutions, see Appendix I. The randomized smoothing approach SmoothLip of (Jeong et al., 2021) outperforms us for $\sigma = 0.5$ in terms of clean accuracy and robust accuracy at $\epsilon_2 = 1.5$ but their robust accuracy at $\epsilon_2 = 2$ is zero, whereas we have 56.1% exact robust accuracy. Their second model with $\sigma = 1$ which is able to certify also larger radii is in all aspects worse than our ℓ_2-PNPC. This shows that our certified prototype classifiers can challenge neural networks in terms of certified robust accuracy. Moreover, (Saralajew et al., 2020) report for their ℓ_2-GLVQ a certified robust accuracy of 34.4% at $\epsilon = 1.58$ whereas with our exact computation we get that their exact robust accuracy is 67.1%. This shows the quality of our exact certification techniques. With our certified training PNPC has 6% better robust accuracy and 1.5% better standard accuracy (97.3% vs. 95.8%) than ℓ_2-GLVQ.

Table 5: **MNIST:** lower (CRA) and upper bounds (URA) on ℓ_2-robust accuracy for ℓ_2-NPC

MNIST	std. acc.	$\epsilon_2 = 1.5$ CRA	$\epsilon_2 = 1.5$ URA	$\epsilon_2 = 1.58$ CRA	$\epsilon_2 = 1.58$ URA	$\epsilon_2 = 2$ CRA	$\epsilon_2 = 2$ URA
ℓ_2-PNPC	97.3	**75.5**	75.5	**73.0**	73.0	**56.1**	56.1
ℓ_2-GLVQ	95.8	69.7	69.7	67.1	67.1	53.5	53.5
1-NN	96.9	52.1	52.1	47.3	47.3	23.7	23.7
GloRob	97.0	-	-	62.8	81.9	-	-
OrthConv	**98.1**	-	-	61.0	75.5	-	-
LocLip	96.3	-	-	55.8	78.2	-	-
BCP	92.4	-	-	47.9	64.7	-	-
CAP	88.1	-	-	44.5	67.9	-	-
SmoothLip$_{\sigma=0.5}$	**98.7**	**81.8***	-	-	-	0*	-
SmoothLip$_{\sigma=1}$	93.7	62.7*	-	-	-	44.9*	-

Table 6: **MNIST:** lower (CRA) and upper bounds (URA) on robust accuracy for multiple threat models for our ℓ_2-PNPC, the ℓ_2-NPC of (Saralajew et al., 2020), a 1-NN classifier. As comparison we show MMR-Univ of (Croce & Hein, 2020a) which is a neural network specifically trained for certifiable multiple-norm robustness.

MNIST	std. acc.	$\epsilon_1 = 1$ CRA	$\epsilon_1 = 1$ URA	$\epsilon_2 = 0.3$ CRA	$\epsilon_2 = 0.3$ URA	$\epsilon_\infty = 0.1$ CRA	$\epsilon_\infty = 0.1$ URA	union CRA	union URA
ℓ_2-PNPC	**97.3**	**96.2**	96.2	**95.6**	95.6	85.8	85.8	**85.8**	85.8
ℓ_2-GLVQ	95.8	94.2	94.2	93.2	93.2	80.9	80.9	80.9	80.9
1-NN	96.9	95.0	-	93.6	93.6	78.3	-	78.3	-
MMR-U	97.0	79.2	93.6	89.6	93.8	**87.6**	87.6	79.2	87.6

The advantage of our ℓ_2-NPC is that we can certify any ℓ_q-threat model, especially ℓ_1 and ℓ_∞. This allows us to compute the **exact robust accuracy in the union of the ℓ_1-, ℓ_2- and ℓ_∞-balls.** The only other approach which has provided certified lower bounds (CRA) on multiple-norm robustness is MMR-U from (Croce & Hein, 2020a) who certify a neural network. In Table 6 we compare our multiple-norm robust accuracy for the ϵ_q which were chosen in (Croce & Hein, 2020a). Our ℓ_2-PNPC outperforms MMR-U significantly in terms of certified ℓ_1-and ℓ_2-robustness as well as in the union.

Table 7: **MNIST:** lower (CRA) and upper bounds (URA) on ℓ_∞-robust accuracy for ℓ_∞-NPC obtained using Lemma 2.9.

MNIST	std. acc.	$\epsilon_\infty = 0.1$ CRA	$\epsilon_\infty = 0.1$ URA	$\epsilon_\infty = 0.3$ CRA	$\epsilon_\infty = 0.3$ URA	$\epsilon_\infty = 0.4$ CRA	$\epsilon_\infty = 0.4$ URA
ℓ_∞-PNPC	94.69	91.19	91.19	78.68	78.86	65.58	65.96
ℓ_∞-GLVQ	96.34	93.52	93.52	80.76	81.04	61.29	62.94
ℓ_∞-neuron	**98.6**	-	-	**93.1**	95.3	-	-
CROWN-IBP	98.2	-	-	93.0	94.0	**87.4**	90.4
ReLU-S	97.3	-	-	80.7	92.1	-	-
CAP	87.4	-	-	56.9	-	-	-

MNIST - ℓ_∞-NPC We compare our ℓ_∞-PNPC to the ℓ_∞-GLVQ of (Saralajew et al., 2020). For reference we provide the best results for the ℓ_∞-certfied neural networks: ℓ_∞-neurons (Zhang et al., 2021), CROWN-IBP (Zhang et al., 2020), as well as slightly older results; ReLU-stability (Xiao et al., 2019) and CAP (Wong et al., 2018) to put our results into context. We perform slightly worse than (Saralajew et al., 2020) for small radii, but significantly better for the bigger one. Due to our better lower bounds but also by using AutoAttack (Croce & Hein, 2020b) for computing the upper bounds we close the gap between upper and lower bounds from 4.2% in (Saralajew et al., 2020) to 0.3%. To attack the classifier with AutoAttack, we interpret the negative distance to the closest prototype from a particular class as the logit value.

Perceptual metric NPC As discussed in Section 3 it is unlikely that ℓ_p-NPC will work for image classifcation tasks like CIFAR10. However, with the perceptual metric LPIPS (based on Alexnet) which corresponds to an ℓ_2-metric in the embedding space, we get much better results with our Perceptual-PNPC (P-PNPC). In Figure 4 we show the certified robust accuracy (lower bound of Lemma 2.9) as a function of the LPIPS-radius for the standard ℓ_2-lower bounds and for the improved lower bounds taking into account the constraints of the embedding. We have three important observations. We achieve a clean accuracy of 80.3% which is quite remarkable for a classifier with certified robust accuracy. Second, this is up to our knowledge the first result on certified robustness with respect to the LPIPS-threat model. Third, (Laidlaw et al., 2021) who do empirical perceptual adversarial training with a a ResNet 50 get only 71.6% clean accuracy and only a URA of 9.8% which is more than 30% **worse** than our CRA of 40.5%. Moreover, our URA computed using the LPA-attack of (Laidlaw et al., 2021) is with 70.3% remarkably high. These are very promising results justifying more research in PNPC for perceptual metrics.

On the other hand, in (Laidlaw et al., 2021) it is noted that models trained to be robust w.r.t. LPIPS-threat model are empirically robust also to other threat models such as ℓ_2 or ℓ_∞ - even though one has to state that their model has only a robust accuracy of 9.8%. This generalization does not hold for P-PNPC. For ℓ_∞ threat model, we observed (empirical) robust accuracies $49\%, 23\%, 2\%, 0\%$ for radii $1/255, 2/255, 4/255, 8/255$. For ℓ_2 we have robust accuracy $51\%, 29\%, 5\%, 0\%$ for radii $0.14, 0.25, 0.5, 1$. While the robust accuracies are non-trivial, they are not comparable to the ones achieved in (Laidlaw et al., 2021). As our P-PNPC is much more robust with respect to the LPIPS-threat model than the neural network of (Laidlaw et al., 2021), it is thus an open question if this threat model leads indeed to a generalization to other threat models.

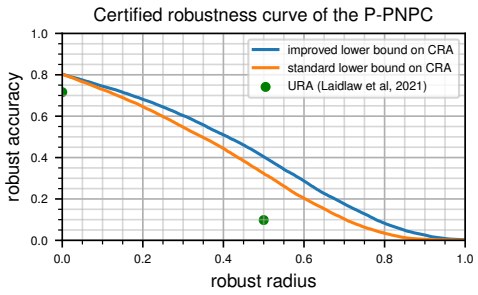


Figure 4: The certified robust accuracy as a function of the radius of the LPIPS-threat model. Integrating the spherical plus non-negativity constraints leads to huge improvements. The standard accuracy as well as the **empirical** robust accuracy of (Laidlaw et al., 2021) are *worse* than **certified** robust accuracy of P-PNPC by a large margin.

6. Conclusion

We have provided theoretical foundations as well as efficient algorithmic tools for the computation of the exact minimal adversarial perturbation, as well as lower bounds, for nearest prototype classifiers for several threat models, including the perceptual metric LPIPS. We have shown SOTA performance for deterministic ℓ_2-certification on MNIST and remarkably strong certified robustness results with respect to the LPIPS metric. Thus we think that NPC deserve more attention in our research community.

Acknowledgements

The authors acknowledge support from the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039A) and the DFG Cluster of Excellence “Machine Learning – New Perspectives for Science”, EXC 2064/1, project number 390727645.

References

Andriushchenko, M. and Hein, M. Provably robust boosted decision stumps and trees against adversarial attacks. In *NeurIPS*, 2019.

Athalye, A., Carlini, N., and Wagner, D. A. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Bertsimas, D., Dunn, J., Pawlowski, C., and Zhuo, Y. D. Robust classification. *INFORMS Journal on Optimization*, 1:2–34, 2018.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., Madry, A., and Kurakin, A. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Chen, H., Zhang, H., Boning, D., and Hsieh, C.-J. Robust decision trees against adversarial examples. In *ICML*, 2019.

Cohen, J. M., Rosenfeld, E., and Kolter, J. Z. Certified adversarial robustness via randomized smoothing. In *NeurIPS*, 2019.

Crammer, K., Gilad-bachrach, R., Navot, A., and Tishby, N. Margin analysis of the lvq algorithm. In *NeurIPS*, 2003.

Croce, F. and Hein, M. Provable robustness against all adversarial l_p-perturbations for $p \geq 1$. In *ICLR*, 2020a.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks, 2020b.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T. A., and Kohli, P. On the effectiveness of interval bound propagation for training verifiably robust models. preprint, arXiv:1810.12715v3, 2018.

Hein, M. and Andriushchenko, M. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NeurIPS*, 2017.

Huang, Y., Zhang, H., Shi, Y., Kolter, J. Z., and Anandkumar, A. Training certifiably robust neural networks with efficient local lipschitz bounds. In *NeurIPS*, 2021.

Jeong, J., Park, S., Kim, M., Lee, H.-C., Kim, D., and Shin, J. Smoothmix: Training confidence-calibrated smoothed classifiers for certified robustness. In *NeurIPS*, 2021.

Kantchelian, A., Tygar, J., and Joseph, A. Evasion and hardening of tree ensemble classifiers. In *ICML*, 2016.

Kireev, K., Andriushchenko, M., and Flammarion, N. On the effectiveness of adversarial training against common corruptions. *arXiv preprint, arXiv:2103.02325*, 2021.

Kohonen, T. *Learning Vector Quantization*, pp. 175–189. Springer Berlin Heidelberg, 1995.

Laidlaw, C., Singla, S., and Feizi, S. Perceptual adversarial robustness: Defense against unseen threat models. In *ICLR*, 2021.

Lee, S., Lee, J., and Park, S. Lipschitz-certifiable training with a tight outer bound. In *NeurIPS*, 2020.

Leino, K., Wang, Z., and Fredrikson, M. Globally-robust neural networks. In *ICML*, 2021.

Li, L., Qi, X., Xie, T., and Li, B. Sok: Certified robustness for deep neural networks. *arXiv preprint arXiv:2009.04131*, 2020.

Li, Q., Haque, S., Anil, C., Lucas, J., Grosse, R. B., and Jacobsen, J.-H. Preventing gradient attenuation in lipschitz constrained convolutional networks. In *NeurIPS*, 2019.

Mirman, M., Gehr, T., and Vechev, M. Differentiable abstract interpretation for provably robust neural networks. In *ICML*, 2018.

Papernot, N., McDaniel, P., and Goodfellow, I. Transferability in machine learning: from phenomena to black-box attacks using adversarial samples, 2016.

Russu, P., Demontis, A., Biggio, B., Fumera, G., and Roli, F. Secure kernel machines against evasion attacks. In *ACM workshop on AI and security*. ACM, 2016.

Saralajew, S., Holdijk, L., and Villmann, T. Fast adversarial robustness certification of nearest prototype classifiers for arbitrary seminorms. In *NeurIPS*, 2020.

Singla, S., Singla, S., and Feizi, S. Improved deterministic l2 robustness on CIFAR-10 and CIFAR-100. In *ICLR*, 2022.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *ICLR*, pp. 2503–2511, 2014.

Tjeng, V. and Tedrake, R. Verifying neural networks with mixed integer programming. preprint, arXiv:1711.07356v1, 2017.

Tramer, F., Carlini, N., Brendel, W., and Madry, A. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020.

Trockman, A. and Kolter, J. Z. Orthogonalizing convolutional layers with the cayley transform. In *ICLR*, 2021.

Wang, L., Liu, X., Yi, J., Zhou, Z.-H., and Hsieh, C.-J. Evaluating the robustness of nearest neighbor classifiers: A primal-dual perspective. *arXiv preprint, arXiv:1906.03972*, 2019.

Wang, Y., Jha, S., and Chaudhuri, K. Analyzing the robustness of nearest neighbors to adversarial examples. In *ICML*, 2018.

Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Wong, E. and Kolter, J. Z. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *ICML*, 2018.

Wong, E., Schmidt, F., Metzen, J. H., and Kolter, J. Z. Scaling provable adversarial defenses. In *NeurIPS*, 2018.

Xiao, K. Y., Tjeng, V., Shafiullah, N. M., and Madry, A. Training for faster adversarial robustness verification via inducing relu stability. In *ICLR*, 2019.

Xu, H., Caramanis, C., and Mannor, S. Robustness and regularization of support vector machines. *Journal of Machine Learning Research*, 10:1485–1510, 2009.

Zhang, B., Cai, T., Lu, Z., He, D., and Wang, L. Towards certifying l-infinity robustness using neural networks with l-inf-dist neurons. In *ICML*, 2021.

Zhang, H., Chen, H., Xiao, C., Gowal, S., Stanforth, R., Li, B., Boning, D., and Hsieh, C.-J. Towards stable and efficient training of verifiably robust neural networks. In *ICLR*, 2020.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

The appendix includes the missing proofs from the paper (App. A to App. G), results for ℓ_2-NPC for CIFAR10 in App. H and comparison to orthogonal convolutions in I.

A. Proof of Theorem 2.4

Proof. We note that for any x it holds by the triangle inequality

$$d(x, w_i) \leq d(z, x) + d(w_i, z).$$

Thus it holds

$$d(x, w_i) - d(x, w_j) \leq d(z, w_i) + d(x, z) - d(z, w_j) + d(x, z),$$

and we get that all points in $B_d(z, r)$ are classified the same as z if

$$\max_{x \in B_d(z,r)} \left(\min_{i \in I_y} d(x, w_i) - \min_{j \in I_y^c} d(x, w_j) \right) \leq \min_{i \in I_y} d(z, w_i) - \min_{j \in I_y^c} d(z, w_j) + 2r \leq 0$$

This yields that

$$r \leq \frac{\min_{j \in I_y^c} d(z, w_j) - \min_{i \in I_c} d(z, w_i)}{2}.$$

□

B. Proof of Theorem 2.6

Proof. We define the set

$$U_j^{(p)} = \{x \in \mathbb{R}^n \mid \|x - w_i\|_p - \|x - w_j\|_p \geq 0 \quad \forall\, i \in I_y\}.$$

as the set of points which are not classified as y when only the single prototype with index $j \in I_y^c$ would be considered. We get the full set of points not classified as y as the union $\bigcup_{j \in I_y^c} U_j^{(p)}$. We define $r_p^q(z)_j = \min_{x \in U_j^{(p)}} \|z - x\|_q$ as the radius of the largest ℓ_q-ball which still fits into $\mathbb{R}^d \backslash U_j^{(p)}$ and thus is fully classified as class y when only considering $j \in I_y^c$. Thus the radius $\epsilon_p^q(z)$ of the largest ℓ_q-ball fitting into $\mathbb{R}^d \backslash \bigcup_{j \in I_y^c} U_j^{(p)} = \bigcap_{j \in I_y^c} \left(\mathbb{R}^d \backslash U_j^{(p)} \right)$ is given by

$$\epsilon_p^q(z) = \min_{j \in I_y^c} r_p^q(z)_j,$$

which can be seen using the fact that $r_p^q(z)_j$ is the minimal ℓ_q-distance to $U_j^{(p)}$. □

C. Proof of Lemma 2.9

Proof. As for each $i \in I_y$ the problem for $\rho_p^q(z)_{i,j}$ is a relaxation of the problem for $r_p^q(z)_j$ (as we are omitting constraints), it holds for each $i \in I_y$:

$$r_p^q(z)_j \geq \rho_p^q(z)_{i,j} \implies r_p^q(z)_j \geq \max_{i \in I_y} \rho_p^q(z)_{i,j}.$$

Thus

$$\epsilon_q^p(z) = \min_{j \in I_y^c} r_p^q(z)_j \geq \min_{j \in I_c^y} \max_{i \in I_y} \rho_p^q(z)_{i,j}.$$

For the second part if x^* satisfies

$$\|x^* - w_i\|_p - \|x^* - w_{j^*}\|_p \geq 0 \quad \forall i \in I_y,$$

then it is a feasible point for the optimization problem of $r_p^q(z)_{j^*}$ in (1) and thus $\rho_p^q(z)_{i^*,j^*} = r_p^q(z)_{j^*}$. By definition and by the just derived result it holds

$$\rho_p^q(z)_{i^*,j^*} = r_p^q(z)_{j^*} \geq \epsilon_p^q(z) \geq \rho_p^q(z)_{i^*,j^*},$$

and thus equality has to hold. □

D. Proof of Theorem 2.10

Lemma D.1. $\rho_p^p(z)_{i,j} \geq \frac{\|w_j - z\|_p - \|w_i - z\|_p}{2}$

Proof. First we restate the definition of ρ:

$$\begin{aligned} \rho_p^q(z)_{i,j} = \min_{x \in \mathbb{R}^d} \quad & \|x - z\|_q \\ \text{sbj. to:} \quad & \|x - w_i\|_p - \|x - w_j\|_p \geq 0 \\ & x \in \mathcal{X} \end{aligned} \tag{8}$$

We consider $p = q$. By the triangle inequality the following holds for any $x \in \mathcal{X}$, thus also for any adversarial perturbation x for which $\|x - w_i\|_p - \|x - w_j\|_p \geq 0$:

$$\begin{aligned} &\|x - w_i\|_p \leq \|x - z\|_p + \|z - w_i\|_p \\ &\|z - w_j\|_p \leq \|x - z\|_p + \|x - w_j\|_p \quad \implies \quad \|x - w_j\|_p \geq \|z - w_j\|_p - \|x - z\|_p \end{aligned} \tag{9}$$

Summing the inequalities up we get for any feasible $x \in \mathcal{X}$ satisfying the inequality constraint,

$$\|z - w_i\|_p - \|z - w_j\| + 2\|x - z\|_p \geq \|x - w_i\|_p - \|x - w_j\|_p \geq 0. \tag{10}$$

which yields finally

$$\|x - z\|_p \geq \frac{\|w_j - z\|_p - \|w_i - z\|_p}{2}. \tag{11}$$

Therefore, $\rho_p^p(z)_{i,j} \geq \frac{\|w_j - z\|_p - \|w_i - z\|_p}{2}$. □

Proof of Theorem 2.10. If z is misclassified, then it reduces to $0 \geq 0$ which holds. Otherwise, by Lemma D.1, it holds $\rho_p^p(z)_{i,j} \geq \frac{\|z - w_j\|_p - \|z - w_i\|_p}{2}$. Then

$$\begin{aligned} \min_{j \in I_c^y} \max_{i \in I_y} \rho_p^p(z)_{i,j} &\geq \min_{j \in I_c^y} \rho_p^p(z)_{i^*,j} \\ &\geq \min_{j \in I_c^y} \frac{\|z - w_j\|_p - \|z - w_{i^*}\|_p}{2} \\ &= \frac{\min_{j \in I_y^c} \|z - w_j\|_p - \min_{i \in I_y} \|z - w_i\|_p}{2} \end{aligned}$$

We further show that there are cases where the inequality is strict. Consider a d-dimensional example where $z = (0, \ldots, 0)$, $\{w_j \mid j \in I_c^y\} = \{(2, 0, \ldots, 0)\}$, $\{w_i \mid i \in I^y\} = \{(1, 0, \ldots, 0)\}$. It clearly holds that $\min_{j \in I_c^y} \max_{i \in I_y} \rho_p^p(z)_{i,j} = 1.5$, while $\max\left\{0, \frac{\min_{j \in I_y^c} \|z - w_j\|_p - \min_{i \in I_y} \|z - w_i\|_p}{2}\right\} = 1$ for any p.

□

E. Proof of Theorem 2.7

Theorem 2.7 *The computational complexities of optimization problems $\rho_p^q(z)_{i,j}$ for $p, q \in \{1, 2, \infty\}$ for $\mathcal{X} = \mathbb{R}^d$ are summarised in Table 8.*

Throughout the proof, we assume z is correctly classified, otherwise the solution is 0. We prove the theorem gradually for cases $p = 2$ and any q, then $q = \infty$ and any p, then $p = 1$ and any $q \neq \infty$ and finally $p = \infty$, $q = 1, 2$. For most of the cases, we discuss the possibility of incorporating box constraints, which usually increases complexity from $O(d)$ to $O(d \log(d))$. We also remark that using the median of medians algorithm, one could avoid sorting coordinates, and could achieve $\Theta(d)$ complexities. We, for the sake of simplicity, will be sorting point for the price of $log(d)$ factor in complexity.

ℓ_p-distance \ ℓ_q-threat model	ℓ_1	ℓ_2	ℓ_∞
ℓ_1	NP-hard	NP-hard	$O(d\log(d))$
ℓ_2	$\Theta(d)$	$\Theta(d)$	$\Theta(d)$
ℓ_∞	$\Theta(d)$	$O(d\log(d))$	$\Theta(d)$

Table 8: Computational complexity of $\rho_p^q(z)_{i,j}$.

Proof for case $p = 2$ ***and any*** q.

$$\begin{aligned} \rho_2^q(z)_{i,j} = \min_{x\in\mathbb{R}^d} \quad & \|x-z\|_q \\ \text{sbj. to:} \quad & \|x-w_i\|_2 - \|x-w_j\|_2 \geq 0 \\ & x \in \mathcal{X} \end{aligned} \tag{12}$$

We equivalently rewrite the constraint in the following way:

$$\begin{aligned} \|x-w_i\|_2 - \|x-w_j\|_2 &\geq 0, \\ \|x-z+z-w_i\|_2^2 - \|x-z+z-w_j\|_2^2 &\geq 0, \\ \|x-z\|_2^2 + 2\langle x-z, z-w_i\rangle + \|z-w_i\|_2^2 - \left(\|x-z\|_2^2 + 2\langle x-z, z-w_j\rangle + \|z-w_j\|_2^2\right) &\geq 0, \\ 2\langle x-z, w_j-w_i\rangle &\geq \|z-w_j\|_2^2 - \|z-w_i\|_2^2, \\ 2\|x-z\|_q \|w_j-w_i\|_{\frac{q}{q-1}} \geq 2\langle x-z, w_j-w_i\rangle &\geq \|z-w_j\|_2^2 - \|z-w_i\|_2^2, \\ \|x-z\|_q &\geq \frac{\|z-w_j\|_2^2 - \|z-w_i\|_2^2}{2\|w_j-w_i\|_{\frac{q}{q-1}}}. \end{aligned}$$

Since Hölder's inequality is tight, we conclude

$$\rho_2^q(z)_{i,j} = \frac{\|z-w_j\|_2^2 - \|z-w_i\|_2^2}{2\|w_j-w_i\|_{\frac{q}{q-1}}}. \tag{13}$$

We note that analogical derivation holds for minimising $\|x-z\|$ in any norm, not just for the q-norm. In that case, $\|\cdot\|_{\frac{q}{q-1}}$ is replaced with the dual norm of the considered norm. The box-constrained version of this problem can be solved in $O(d\log d)$, see e.g., Section 4 of (Hein & Andriushchenko, 2017). □

Proof for case $p = q = \infty$.

$$\begin{aligned} \rho_p^\infty(z)_{i,j} = \min_{x\in\mathbb{R}^d} \quad & \|x-z\|_\infty \\ \text{sbj. to:} \quad & \|x-w_i\|_\infty - \|x-w_j\|_\infty \geq 0 \\ & x \in \mathcal{X} \end{aligned} \tag{14}$$

We note that whenever $\|x-w_i\|_\infty - \|x-w_j\|_\infty \geq 0$, then also $\|x'-w_i\|_\infty - \|x'-w_j\|_\infty \geq 0$, where $x'^{(l)} = x^{(l)} + \alpha\,\mathrm{sign}(w_j^{(l)} - w_i^{(l)})$ for any positive α and some $l = 1\ldots d$, and $x'^{(l)} = x^{(l)}$ for the coordinates. That is, we can move $x^{(l)}$ in the direction from $w_i^{(l)}$ to $w_j^{(l)}$, since if $|x'^{(l)} - w_j^{(l)}| > |x^{(l)} - w_j(l)|$, then also $|x'^{(l)} - w_i^l| > |x'^{(l)} - w_j(l)|$. On the other hand, if $|x^{(l)} - w_i^{(l)}| > |x'^{(l)} - w_i^{(l)}|$, then also $|x^{(l)} - w_i^{(l)}| < |x^{(l)} - w_j^{(l)}|$, thus l was not the maximising index of $\|x-w_i\|_\infty$, and consequently $\|x-w_i\|_\infty = \|x'-w_i\|_\infty$. The remaining case is trivial; thus, $\|x'-w_i\|_p - \|x'-w_j\|_p \geq 0$. This argument may be repeated d times to conclude that when $\rho_\infty^\infty(z)_{i,j} = \epsilon$, then a minimizer of Problem 14 is $x^* = z + \epsilon\,\mathrm{sign}\,(w_j - w_i)$.

Therefore, the problem is to find the smallest ϵ for which $\|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_i\|_\infty \geq \|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_j\|_\infty$. Note that

$$\|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_j\|_\infty = \max_{l=1,\dots,d} \max\left\{z^{(l)}+\epsilon\,\mathrm{sign}\left(w_j^{(l)}-w_i^{(l)}\right)-w_j^{(l)}, -\left(z^{(l)}+\epsilon\,\mathrm{sign}\left(w_j^{(l)}-w_i^{(l)}\right)-w_j^{(l)}\right)\right\};$$

thus, it is a maximum of $2d$ linear functions, each of which has slope either 1, or -1. Let $\alpha_i = \arg\min \mathrm{sign}(w_j-w_i)(z-w_i)$ and $\beta_i = \arg\max \mathrm{sign}(w_j-w_i)(z-w_i)$, analogously for α_j, β_j. Then

$$\begin{aligned}
&\|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_i\|_\infty - \|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_j\|_\infty = \\
&\max\left\{-\epsilon-\mathrm{sign}\left(w_j^{(\alpha_i)}-w_i^{(\alpha_i)}\right)\left(z^{(\alpha_i)}-w_i^{(\alpha_i)}\right), \epsilon+\mathrm{sign}\left(w_j^{(\beta_i)}-w_i^{(\beta_i)}\right)\left(z^{(\beta_i)}-w_i^{(\beta_i)}\right)\right\} - \\
&\max\left\{-\epsilon-\mathrm{sign}\left(w_j^{(\alpha_j)}-w_i^{(\alpha_j)}\right)\left(z^{(\alpha_j)}-w_j^{(\alpha_j)}\right), \epsilon+\mathrm{sign}\left(w_j^{(\beta_j)}-w_i^{(\beta_j)}\right)\left(z^{(\beta_j)}-w_j^{(\beta_j)}\right)\right\}.
\end{aligned}$$

Moreover, we can analyse to slope of $\|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_i\|_\infty - \|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_j\|_\infty$ and see that it is non-zero only in the interval between points

$$\epsilon_i = \frac{-\mathrm{sign}\left(w_j^{(\alpha_i)}-w_i^{(\alpha_i)}\right)\left(z^{(\alpha_i)}-w_i^{(\alpha_i)}\right) - \mathrm{sign}\left(w_j^{(\beta_i)}-w_i^{(\beta_i)}\right)\left(z^{(\beta_i)}-w_i^{(\beta_i)}\right))}{2},$$

and

$$\epsilon_j = \frac{-\mathrm{sign}\left(w_j^{(\alpha_j)}-w_i^{(\alpha_j)}\right)\left(z^{(\alpha_j)}-w_j^{(\alpha_j)}\right) - \mathrm{sign}\left(w_j^{(\beta_j)}-w_i^{(\beta_j)}\right)\left(z^{(\beta_j)}-w_j^{(\beta_j)}\right))}{2},$$

where the slope is 2. Now it is easy to compute the value of $\|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_i\|_\infty - \|z+\epsilon\,\mathrm{sign}(w_j-w_i)-w_j\|_\infty$ for very big (V_+) and very small (V_-) values of ϵ, where the active linear function is the one with negative slope. Concretely

$$\begin{aligned}
V_- &= \mathrm{sign}\left(w_j^{(\alpha_j)}-w_i^{(\alpha_j)}\right)\left(z^{(\alpha_j)}-w_j^{(\alpha_j)}\right) - \mathrm{sign}\left(w_j^{(\alpha_i)}-w_i^{(\alpha_i)}\right)\left(z^{(\alpha_i)}-w_i^{(\alpha_i)}\right), \\
V_+ &= \mathrm{sign}\left(w_j^{(\beta_i)}-w_i^{(\beta_i)}\right)\left(z^{(\beta_i)}-w_i^{(\beta_i)}\right) - \mathrm{sign}\left(w_j^{(\beta_j)}-w_i^{(\beta_j)}\right)\left(z^{(\beta_j)}-w_j^{(\beta_j)}\right).
\end{aligned}$$

Now we use the fact that the slope is 2 between ϵ_i and ϵ_j to find the point where the norms are equal; Thus, we can express $\rho_\infty^\infty(z)_{i,j}$ as

$$\rho_\infty^\infty(z)_{i,j} = \max\{\epsilon_i,\epsilon_j\} - \frac{V_+}{2},$$

or as

$$\rho_\infty^\infty(z)_{i,j} = \min\{\epsilon_i,\epsilon_j\} - \frac{V_-}{2}.$$

We can take the mean of both expression, then we arrive at

$$\rho_\infty^\infty(z)_{i,j} = \epsilon_i + \epsilon_j - \frac{V_- + V_+}{2},$$

which simplifies to

$$\rho_\infty^\infty(z)_{i,j} = -\frac{\mathrm{sign}\left(w_j^{(\alpha_j)}-w_i^{(\alpha_j)}\right)\left(z^{(\alpha_j)}-w_j^{(\alpha_j)}\right) + \mathrm{sign}\left(w_j^{(\beta_i)}-w_i^{(\beta_i)}\right)\left(z^{(\beta_i)}-w_i^{(\beta_i)}\right)}{2},$$

and by substituting back the definitions of α_j, β_i:

$$\rho_\infty^\infty(z)_{i,j} = \frac{\max\limits_{l=1,\dots,d} -\operatorname{sign}\left(w_j^{(l)} - w_i^{(l)}\right)\left(z^{(l)} - w_j^{(l)}\right) - \max\limits_{l=1,\dots,d} \operatorname{sign}\left(w_j^{(l)} - w_i^{(l)}\right)\left(z^{(l)} - w_i^{(l)}\right)}{2}. \tag{15}$$

□

Proof for case $q = \infty$, $p \neq \infty$. The value of $\rho_p^\infty(z)_{i,j}$ is the minimal non-negative ϵ for which the following maximization problem has non-negative value.

$$\begin{aligned} \max_{x \in \mathbb{R}^d} \quad & \|x - w_i\|_p^p - \|x - w_j\|_p^p \\ \text{sbj. to:} \quad & \|x - z\|_\infty \leq \epsilon \\ & x \in \mathcal{X} \end{aligned} \tag{16}$$

It can be decomposed into d independent problems indexed by l.

$$\begin{aligned} \max_{x^{(l)} \in \mathbb{R}} \quad & |x^{(l)} - w_i^{(l)}|^p - |x^{(l)} - w_j^{(l)}|^p \\ \text{sbj. to:} \quad & |x^{(l)} - z^{(l)}| \leq \epsilon \\ & x^{(l)} \in \mathcal{X}^{(l)} \end{aligned} \tag{17}$$

Derivative of the objective function w.r.t. $x^{(l)}$ is $p|x^{(l)} - w_i^{(l)}|^{p-1} \operatorname{sign}(x^{(l)} - w_i^{(l)}) - p|x^{(l)} - w_j^{(l)}|^{p-1} \operatorname{sign}(x^{(l)} - w_j^{(l)})$, which is non-zero whenever $w_i^{(l)} \neq w_j^{(l)}$. Thus, the maximum is attained at a point where a constraint is active, and the value of the problem is $|z^{(l)} + \epsilon \operatorname{sign}(w_j^{(l)} - w_i^{(l)}) - w_i^{(l)}|^p - |z^{(l)} + \epsilon \operatorname{sign}(w_j^{(l)} - w_i^{(l)}) - w_j^{(l)}|^p$. When $p = 2$, the value of the objective is a quadratic function in ϵ; thus, the value of the original objective is also a quadratic function in ϵ and we can easily obtain a solution to the original problem. For the sake of completeness, we show that this approach results in the same $\rho_2^\infty(z)_{i,j}$ as we derived before:

$$\begin{aligned} \sum_{l=1}^d \left(\left(z^{(l)} + \epsilon \operatorname{sign}(w_j^{(l)} - w_i^{(l)}) - w_i^{(l)}\right)^2 - \left(z^{(l)} + \epsilon \operatorname{sign}(w_j^{(l)} - w_i^{(l)}) - w_j^{(l)}\right)^2 \right) &\geq 0, \\ \sum_{l=1}^d \left((z^{(l)} - w_i^{(l)})^2 - (z^{(l)} - w_j^{(l)})^2 + 2\epsilon \operatorname{sign}(w_j^{(l)} - w_i^{(l)})(w_j^{(l)} - w_i^{(l)}) \right) &\geq 0, \\ \|z - w_i\|_2^2 - \|z - w_j\|_2^2 + 2\epsilon \|w_j - w_i\|_1 &\geq 0, \\ \epsilon &\geq \frac{\|z - w_j\|_2^2 - \|z - w_i\|_2^2}{2\|w_j - w_i\|_1}. \end{aligned} \tag{18}$$

If $p = 1$, the value of the objective is piecewise linear and non-decreasing; thus, the original objective is again, piecewise linear and non-decreasing. Then we can order the breaking points and find the smallest admissible ϵ for the original problem using binary search. Note that the objective is maximised not just in the aforementioned case, but also when

$$x^{(l)} = \begin{cases} w_j^{(l)}, & \text{if } |z^{(l)} - w_j^{(l)}| \leq \epsilon. \\ z_j^{(l)} + \epsilon \operatorname{sign}\left(w_j^l - z^l\right), & \text{otherwise.} \end{cases} \tag{19}$$

For other values of p, it may be difficult to solve the problem exactly. However, as we have already shown, it is easy ($\Theta(d)$) to determine if $\rho_p^\infty(z)_{i,j} > \epsilon$ given an ϵ, thus the problem can be solved approximately using binary search for any p with logarithmic complexity in accuracy.

To conclude the cases $\rho_\infty^p(z)_{i,j}$, we discuss the addition of box constraints. As we have shown, a minimizer of the problems is always $x^* = z + \epsilon \operatorname{sign}(w_j - w_i)$, and identical arguments would suggest that with box constraints, it would

hold that $x^* = \max(0, \min(1, z + \epsilon \operatorname{sign}(w_j - w_i)))$. Therefore, given a radius, certification is done in $O(d)$ even with box constraints. Otherwise, we would need to either order coordinates according to value of ϵ when a box constraint for $x^* = \max(0, \min(1, z + \epsilon \operatorname{sign}(w_j - w_i)))$ becomes active, and then perform a binary search over the constrained problems. This adds a $log(d)$ factor to the complexity. Note that for the case $p = 1$ we are already performing a binary search, so we do them at once. Or we can do a binary search over ϵ to find a minimal one which causes

$$x = \max(0, \min(1, z + \epsilon \operatorname{sign}(w_j - w_i))) \tag{20}$$

to be misclassified.

□

Proof for case $p = 1$, $q \neq \infty$. We ave already discussed the case of $\rho_1^\infty(z)_{i,j}$, so it is omitted here. For all other values of q, we show its NP-hardness by reducing the knapsack problem to the decision version of problem if given $\epsilon > 0$, $\rho_1^\infty(z)_{i,j} \leq \epsilon$.

Theorem E.1 (Knapsack)**.** *The following problem is NP-complete.*
Given vectors $w, p \in \mathbb{N}^n$ and constants W, P. Decide if there is a vector $x \in \{0,1\}^n$ such that $\langle p, x\rangle \geq P$ and $\langle w, x\rangle \leq W$.

For the sake of clarity, we use u, v instead of w_i, w_j to get rid of unnecessary subscript. Let w, p, W, P describe an instance of the knapsack problem. Let a pair of prototypes $u_t, v_t \in \mathbb{R}^{n+2}$ be defined in the following way for some real t and $l = 1, \ldots, n$

$$\begin{aligned} u_t^{(l)} &= \sqrt[q]{w^{(l)}}, \\ v_t^{(l)} &= \sqrt[q]{w^{(l)}} - \frac{p^{(l)}}{t}, \end{aligned} \tag{21}$$

let also

$$\begin{aligned} u_t^{(n+1)} &= \sqrt[q]{W} + \frac{\max\left(0, \left(2P - \sum_{i=1}^n p^{(i)}\right)\right)}{t}, \\ v_t^{(n+1)} &= \sqrt[q]{W}, \\ u_t^{(n+2)} &= \sqrt[q]{W}, \\ v_t^{(n+2)} &= \sqrt[q]{W} + \frac{\max\left(0, \left(\sum_{i=1}^n p^{(i)} - 2P\right)\right)}{t}, \end{aligned} \tag{22}$$

and $\epsilon = \sqrt[q]{W}$. Now we show that whenever there is an allocation $x \in \{0,1\}^n$ such that $\langle p, x\rangle \geq P$ and $\langle w, x\rangle \leq W$, then $\rho_1^q(0) \leq \epsilon$ for any sufficiently large t such that the first n components of $v^{(t)}$ are positive. It holds that:

$$\|v_t\|_1 \leq \sum_{i=1}^n \sqrt[q]{w^{(i)}} - \sum_{i=1}^n \frac{p^{(i)}}{t} + 2\sqrt[q]{W} + \frac{\max\left(0, \left(\sum_{i=1}^n p^{(i)} - 2P\right)\right)}{t} \leq \sum_{i=1}^n \sqrt[q]{w^{(i)}} + 2\sqrt[q]{W} \leq \|u_t\|_1 . \tag{23}$$

Consider the following point

$$\delta^{(k)} = \begin{cases} \sqrt[q]{w^{(k)}}, & \text{if } x^{(k)} = 1. \\ 0, & \text{otherwise.} \end{cases} \tag{24}$$

It has q-norm of at most ϵ:

$$\|\delta\|_q = \left(\sum_{i=1}^{n+2} \delta^{(i)q}\right)^{\frac{1}{q}} = \left(\sum_{i=1}^{n} x^{(i)} \cdot w^{(i)}\right)^{\frac{1}{q}} \leq \sqrt[q]{W} = \epsilon. \tag{25}$$

Also it holds that

$$\begin{aligned}\|v_t-\delta\|_1 &= \sum_{i=1}^{n}\left(x^{(i)}\cdot\frac{p^{(i)}}{t}+(1-x^{(i)})\sqrt[q]{w^{(i)}}\right)+2\sqrt[q]{W}+\frac{\max\left(0,\left(\sum_{i=1}^{n}p^{(i)}-2P\right)\right)}{t},\\ &\geq \sum_{i=1}^{n}(1-x^{(i)})\sqrt[q]{w^{(i)}}+2\sqrt[q]{W}+\frac{P+\max\left(0,\left(\sum_{i=1}^{n}p^{(i)}-2P\right)\right)}{t},\\ &\geq \sum_{i=1}^{n}(1-x^{(i)})\sqrt[q]{w^{(i)}}+2\sqrt[q]{W}+\frac{\sum_{i=1}^{n}p^{(i)}-P+\max\left(0,\left(2P-\sum_{i=1}^{n}p^{(i)}\right)\right)}{t}\geq\|u_t-\delta\|_1.\end{aligned}\tag{26}$$

Therefore, $\rho_1^q(0)\leq\epsilon$.

Now we move on to the second direction; we show that whenever the constructed problem is feasible, then also the knapsack problem is feasible.

Let there be a δ such that $\|\delta\|_q\leq\epsilon$ and $\|v_t-\delta\|\geq\|u_t-\delta\|$. Then we can WLoG assume $\delta^{(n+1)}=0$, and $\sqrt[q]{w^{(l)}}-p^{(l)}/t\leq\delta^{(l)}\leq\sqrt[q]{w^{(l)}}$ for $l=1,\ldots,n$. Now consider the following allocation for $k=1,\ldots,n$.

$$x^{(k)}=\begin{cases}0, & \text{if } \delta^{(k)}=0.\\ 1, & \text{otherwise.}\end{cases}\tag{27}$$

We show that if t is sufficiently large, then x is a valid allocation. First, let us look at the $\langle w,x\rangle\leq W$ constraint;

$$\sum_{i=1}^{n}\delta^{(i)q}=\sum_{i=1}^{n}w^i\cdot x^i-o(1)=\langle w,x\rangle-o(1)\leq W;$$

thus, $\langle w,x\rangle\leq W$ for sufficiently large t.

For the other constraint, first note for $l=1,\ldots,n$:

$$\left(|v_t-\delta|^{(l)}-|u_t-\delta|^{(i)}\right)=\begin{cases}p^{(i)}/t, & \text{if } x^{(i)}=0.\\ \geq -p^{(i)}/t, & \text{otherwise.}\end{cases}\tag{28}$$

Then

$$\sum_{i=1}^{n}\left(|v_t-\delta|^{(i)}-|u_t-\delta|^{(i)}\right)\geq\frac{\sum_{i=1}^{n}p^{(i)}-2\langle x,p\rangle}{t},\tag{29}$$

and finally

$$\begin{aligned}\frac{\sum_{i=1}^{n}p^{(i)}-2P}{t}&\geq\frac{\sum_{i=1}^{n}p^{(i)}-2\langle x,p\rangle}{t},\\ \langle x,p\rangle&\geq P.\end{aligned}\tag{30}$$

□

Proof for case $p=\infty$, $q=1$.

$$\begin{aligned}\rho_\infty^1(z)_{i,j}=\min_{x\in\mathbb{R}^d}\quad&\|x-z\|_1\\ \text{sbj. to:}\quad&\|x-w_i\|_\infty-\|x-w_j\|_\infty\geq 0\\ &x\in\mathcal{X}\end{aligned}\tag{31}$$

Let $\delta_x = x - z$ where $x \in \arg\min \rho^1_\infty(z)_{i,j}$, We note that there exists x such that δ_x contains only a single non-zero element. To see why, let there be some δ_x with multiple non-zero elements from which we construct $\delta_{x'}$ with more zeros such that $x' \in \arg\min \rho^1_\infty(z)_{i,j}$. Let $l^* = \arg\max_l |x^{(l)} - w_i^{(l)}|$. Take any index $k \neq l^*$ such that $\delta_x^{(k)} \neq 0$. Then consider a perturbation $\delta_x{}'$

$$\delta_{x'}^{(l)} = \begin{cases} \delta_x^{(l)} + |\delta_x^{(k)}| \operatorname{sign}(x^{(l)} - w_i^{(l)}), & \text{if } l = l^*. \\ 0, & \text{if } l = k. \\ \delta_x^{(l)}, & \text{otherwise.} \end{cases} \tag{32}$$

Now, $\|x' - w_i\|_\infty = \|x - w_i\|_\infty + |\delta_x^k| \geq \|x - w_j\|_\infty + |\delta^{(l)}| \geq \|\delta' - w_j\|_\infty$ which concludes the argument. Now it is sufficient to solve the problem for every coordinate separately and take the maximal value; thus, the original problem is solved in linear time.

□

Proof for case $p = \infty$, $q = 2$.

$$\begin{aligned} \rho^2_\infty(z)_{i,j} = \min_{x \in \mathbb{R}^d} \quad & \|x - z\|_2 \\ \text{sbj. to:} \quad & \|x - w_i\|_\infty - \|x - w_j\|_\infty \geq 0 \\ & x \in \mathcal{X} \end{aligned} \tag{33}$$

Let x be the minimizer. Then we split the proof into two cases. Either there is an index l such that $\|x - w_i\|_\infty = |x^{(l)} - w_i^{(l)}| = |x^{(l)} - w_j^{(l)} = \|x - w_j\|_\infty$. In that case, $|z^{(l)} - w_j^{(l)}| > |z^{(l)} - w_i^{(l)}|$ and $|w_i^{(l)} - w_j^{(l)}|$ is maximal. Then we can compute x in one pass and verify that indeed $\|x - w_i\|_\infty = \|x - w_j\|_\infty$.

Otherwise, let us Assume that we know $\|x - w_i\|_\infty = \|x - w_j\|_\infty = d$ for the optimal x. That is, for every coordinate l we have to ensure that $|x^{(l)} - w_j^{(l)}| \leq d$, and also that there is a coordinate k where $|x^{(l)} - w_i^{(l)}| = d$; thus, we can construct x minimizing $\|x - z\|_2$ as

$$x^{(l)} = \begin{cases} w_j^{(l)} + d \operatorname{sign}(z^{(l)} - w_i^{(l)}), & \text{if } |w_j^{(l)} - x^{(l)}| > d. \\ w_i^{(l)} + d \operatorname{sign}(w_j^{(l)} - w_i^{(l)}), & \text{if } l = k. \\ z^{(l)}, & \text{otherwise,} \end{cases} \tag{34}$$

where $k = \min \arg\max_l \quad \operatorname{sign}\left(w_j^{(l)} - w_i^{(l)}\right)\left(z^{(l)} - w_i^{(l)}\right)$.

Now we sort (so further we assume the array is sorted) the coordinates according to values of $|w_j^{(l)} - x^{(l)}|$.

Then minimum of $\|x - z\|_2^2$ is attained for some d which lies in some interval $[|w_j^{(m)} - x^{(m)}|, |w_j^{(m+1)} - x^{(m+1)}|]$. Inside every such interval, $\|x - z\|_2^2$ is a quadratic expression in d. For the m-th interval, the equation is

$$\|x - z\|_2^2 = \sum_{l=1}^{m} \left(z^{(l)} - w_j^{(l)} + d \operatorname{sign}(z^{(l)} - w_i^{(l)})\right)^2 + \left(z^{(l)} - \operatorname{sign}\left(w_j^{(k)} - w_i^{(k)}\right)\left(z^{(k)} - w_i^{(k)}\right)\right)^2.$$

So we can minimize a quadratic function $\|x - z\|_2^2$ over an interval $[|w_j^{(m)} - x^{(m)}|, |w_j^{(m+1)} - x^{(m+1)}|]$. We can also see that for the $m+1$-th equation, we only add one term to the m-th equation; thus, we can solve every interval in $O(1)$ and take the minimal ϵ. Consequently, the time complexity is dominated by $O(d \log(d))$ needed for sorting which concludes the proof.

□

ℓ_p-distance	ℓ_q-threat model		
	ℓ_1	ℓ_2	ℓ_∞
ℓ_1	NP-hard	NP-hard	Poly
ℓ_2	Poly	Poly	Poly
ℓ_∞	NP-hard	NP-hard	NP-hard

Table 9: Computational Complexity of $r^q(z)$.

F. Proof of Theorem 2.8

Theorem 2.8 *The computational complexities of optimization problems $r_p^q(z)_{i,j}$ in (1) for $p, q \in \{1, 2, \infty\}$ and $\mathcal{X} = [0,1]^d$ are summarized in Table 9.*

Proof. The Problem $r_1^q(z)_j$ for $q \neq \infty$ cannot be easier than the problem $\rho_1^q(z)_{i,j}$, thus since the latter is NP-hard, the first also has to be NP-hard. For the case $r_1^\infty(z)_j$, we recall that the optimal argument of $\rho_1^\infty(z)_{i,j}$ was in the form

$$x^{(l)} = \begin{cases} w_j^{(l)}, & \text{if } |z^{(l)} - w_j^{(l)}| \leq \epsilon, \\ z_j^{(l)} + \epsilon \operatorname{sign}\left(w_j^l - z^l\right), & \text{otherwise,} \end{cases} \tag{35}$$

where ϵ is the value of $\rho_1^\infty(z)_{i,j}$. Therefore, $r_1^\infty(z)_j = \max_i \rho_1^\infty(z)_{i,j}$ and the overall complexity is $O(d \log(d) |I_c^y|)$. When $p = 2$, then the problem reads as

$$\begin{aligned} r_2^q(z)_j = \min_{x \in \mathbb{R}^d} \quad & \|x - z\|_q \\ \text{sbj. to:} \quad & \|x - w_i\|_2 - \|x - w_j\|_2 \geq 0 \quad \forall\, i \in I_y \\ & x \in [0,1]^d \end{aligned}$$

which is a convex optimization problem for any q and can be solved in polynomial time.

Finally, for the case $p = \infty$ we show that it is $NP - complete$ to solve the feasibility problem of $r_p^q(z)_j$, thus the problem is NP-hard for any q. To shorten the notation, we consider $I_y = 1, \ldots, n$ and whenever we say that some proposition holds for w_i, then we mean it holds for any w_i, $i \in 1, \ldots, n$.

We show this by reducing 3-SAT to it. Let there be a formula in CNF $\bigwedge_{i=1}^n \left(\alpha^{(i)} \vee \beta^{(i)} \vee \gamma^{(i)}\right)$, where all all the literals are from a set of v variables. For the sake of brevity, we make a correspondence between the literals and indices $1, \ldots, v$. Also when literal corresponding to i is negative, we will write it as $-i$. We will consider the following set of prototypes from $\mathbb{R}^{(v+1)}$.

$$w_j = (0, \ldots, 0, 3)$$

$$w_i^{(l)} = \begin{cases} -1, & \text{if } l \in \{\alpha^{(i)}, \beta^{(i)}, \gamma^{(i)}\}, \\ 2, & \text{if } -l \in \{\alpha^{(i)}, \beta^{(i)}, \gamma^{(i)}\}, \\ 0, & \text{otherwise.} \end{cases}$$

Clearly, for any $x \in [0,1]^d$ it holds that $\|w_j - x\|_\infty \geq 2$, and also $\|x - w_i\|_\infty \leq 2$. Therefore, if $r_\infty^p(z)$ is feasible, then $\|x - w_i\|_\infty = 2$, which is equivalent to proposition $\left(x^{(|\alpha_i|)} = \frac{1+\operatorname{sign}\alpha_i}{2}\right) \vee \left(x^{(|\beta_i|)} = \frac{1+\operatorname{sign}\beta_i}{2}\right) \vee \left(x^{(|\gamma_i|)} = \frac{1+\operatorname{sign}\gamma_i}{2}\right)$. Such proposition have to be satisfied for every i, therefore it is equivalent to a formula in CNF

$$\bigwedge_{i=1}^n \left(\left(x^{(|\alpha_i|)} = \frac{1 + \operatorname{sign}\alpha_i}{2}\right) \vee \left(x^{(|\beta_i|)} = \frac{1 + \operatorname{sign}\beta_i}{2}\right) \vee \left(x^{(|\gamma_i|)} = \frac{1 + \operatorname{sign}\gamma_i}{2}\right)\right),$$

which is clearly equisatisfiable with the original CNF formula; thus, the feasibility problem is NP-complete. □

G. Proofs from the Perceptual Metric NPC

Here we slightly deviate from the main text, that we consider the squared objective which clearly is an equivalent problem

$$\begin{aligned}\rho^2(z)_{i,j} = \min_{x\in\mathbb{R}^d} \quad & \|x-z\|_2^2 \\ \text{sbj. to:} \quad & \langle x, w_j - w_i\rangle + \frac{\|w_i\|_2^2 - \|w_j\|_2^2}{2} \geq 0 \\ & \left\|x^{(l)}\right\|_2^2 = r_l^2 \\ & x \geq 0,\end{aligned}$$

where we use a shortcut $x^{(l)}$, instead of $x^{(h,w,l)}$, to simplify notation.

Proof of Proposition 3.1. Note that

$$\|x-z\|_2^2 = \sum_{l\in I_l} \left\|x^{(l)} - z^{(l)}\right\|_2^2,$$

and as $\left\|z^{(l)}\right\|_2 = r_l$ and we have $\left\|x^{(l)}\right\|_2 = r_l$ as constraint, we can equivalently minimize $-\sum_{l\in I_L}\left\langle x^{(l)}, z^{(l)}\right\rangle$ as objective.

Let $v = w_j - w_i$ and $b = \frac{\|w_i\|_2^2 - \|w_j\|_2^2}{2}$. The Lagrangian of the non-convex problem (due to the quadratic *equality* constraints) is

$$\underset{\mu\geq 0}{L(x,\lambda,\alpha,\mu)} = -\sum_{l\in I_L}\left\langle x^{(l)}, z^{(l)}\right\rangle + \lambda\Big(\sum_{l\in I_l}\left\langle v^{(l)}, x^{(l)}\right\rangle + b\Big) + \sum_{l\in I_L}\frac{\alpha_l}{2}\Big(\left\|x^{(l)}\right\|_2^2 - r_l^2\Big) - \sum_{l\in I_L}\left\langle \mu^{(l)}, x^{(l)}\right\rangle$$

We get as critical point condition:

$$\nabla_{x^{(l)}} L = -z^{(l)} + \lambda v^{(l)} + \alpha_l x^{(l)} - \mu^{(l)} = 0,$$

which yields

$$x^{(l)} = \frac{1}{\alpha_l}\left(z^{(l)} - \lambda v^{(l)} + \mu^{(l)}\right).$$

The dual function $q(\lambda,\alpha,\mu)$ becomes

$$\begin{aligned} q(\lambda,\alpha,\mu) = & -\sum_{l\in I_L}\frac{1}{\alpha_l}\left(\left\|z^{(l)}\right\|_2^2 - \lambda\left\langle v^{(l)}, z^{(l)}\right\rangle + \left\langle \mu^{(l)}, z^{(l)}\right\rangle\right) \\ & + \lambda\left(\sum_{l\in I_L}\frac{1}{\alpha_l}\left(\left\langle v^{(l)}, z^{(l)}\right\rangle - \lambda\left\|v^{(l)}\right\|_2^2 + \left\langle v^{(l)}, \mu^{(l)}\right\rangle\right) + b\right) \\ & + \sum_{l\in I_L}\frac{1}{2\alpha_l}\left(\left\|z^{(l)}\right\|_2^2 + \lambda^2\left\|v^{(l)}\right\|_2^2 + \left\|\mu^{(l)}\right\|_2^2 - 2\lambda\left\langle z^{(l)}, v^{(l)}\right\rangle + 2\left\langle z^{(l)}, \mu^{(l)}\right\rangle - 2\lambda\left\langle v^{(l)}, \mu^{(l)}\right\rangle\right) \\ & - \sum_{l\in I_L}\frac{\alpha_l r_l^2}{2} - \sum_{l\in I_L}\frac{1}{\alpha}\left(\left\langle \mu^{(l)}, z^{(l)}\right\rangle - \lambda\left\langle \mu^{(l)}, v^{(l)}\right\rangle + \left\|\mu^{(l)}\right\|_2^2\right), \end{aligned}$$

which simplifies to

$$q(\lambda,\alpha,\mu) = -\sum_{l\in I_L}\frac{1}{2\alpha_l}\left\|z^{(l)} - \lambda v^{(l)} + \mu^{(l)}\right\|_2^2 + \lambda b - \sum_{l\in I_L}\frac{\alpha_l r_l^2}{2}.$$

We solve explicitly for α and get

$$\alpha_l = \frac{1}{r_l}\left\|z^{(l)} - \lambda v^{(l)} + \mu^{(l)}\right\|_2.$$

Then we get

$$q(\lambda,\mu) = -\sum_{l\in I_L}\left\|z^{(l)} - \lambda v^{(l)} + \mu^{(l)}\right\|_2 r_l + \lambda b.$$

Table 10: **CIFAR10:** lower (CRA) and upper bounds (URA) on ℓ_2-robust accuracy

CIFAR10	std. acc.	$\epsilon_2 = 0.1$ CRA	$\epsilon_2 = 0.1$ URA	$\epsilon_2 = 36/255$ CRA	$\epsilon_2 = 36/255$ URA	$\epsilon_2 = 0.25$ CRA	$\epsilon_2 = 0.25$ URA
PNPC	49.2	43.9	43.9	41.9	41.9	36.4	36.4
GLVQ	48.6	43.3	43.3	41.5	41.5	37.9	37.9
1-NN	35.7	31.2	-	29.7	29.7	25.7	-
GloRob	77.0	-	-	58.4	69.2	-	-
LocLip	**77.4**	-	-	**60.7**	70.4	-	-
BCP	65.7	-	-	51.3	60.8	-	-
SmoothLip$_{\sigma=0.25}$	77.1	-	-	-	-	**67.9***	67.9*

Solving explicitly for μ, we get

$$q(\lambda) = -\sum_{l \in I_L} \left\| \left(z^{(l)} - \lambda v^{(l)} \right)^+ \right\|_2 r_l + \lambda b.$$

So this is a lower bound on $-\sum_{l \in I_L} \langle x^{*(l)}, z^{(l)} \rangle$, where x^* is the optimal primal variable by weak duality and thus going back to our actual objective we get using $\left\| x^{*(l)} \right\|_2 = \left\| z^{(l)} \right\| = r_l$ that

$$\|x^* - z\| = \sqrt{\|x^* - z\|_2^2} = \sqrt{2 \sum_{l \in I_L} r_l^2 - 2 \sum_{l \in I_L} \langle x^{*(l)}, z^{(l)} \rangle} \geq \sqrt{2 \sum_{l \in I_L} r_l^2 + 2 \left(\max_{\lambda \geq 0} - \sum_{l \in I_L} \left\| \left(z^{(l)} - \lambda v^{(l)} \right)^+ \right\|_2 r_l + \lambda b \right)},$$

where we have used weak duality. Now we go back to indexing using h, w, l instead of just l. Since $r_l = \frac{1}{\sqrt{H_l W_l}}$, it holds that

$$\sum_{h=1,\dots,H_l} \sum_{w=1,\dots,W_l} r_l^2 = 1;$$

thus, we can simplify the final expression as

$$\sqrt{2L + 2 \left(\max_{\lambda \geq 0} - \sum_{h,w,l} \left\| \left(z^{(h,w,l)} - \lambda v^{(h,w,l)} \right)^+ \right\|_2 r_l + \lambda b \right)}.$$

Thus we have a one-dimensional convex optimization problem to solve in order to get a lower bound on the original objective, which is all we need for the certification. □

H. Results for CIFAR10 with ℓ_2-NPC

CIFAR10 - ℓ_2-NPC In Table 10 we compare certified robust accuray (CRA) and an upper bound on the robust accuray (URA) of several models on CIFAR10 for ℓ_2-threat model. Our ℓ_2-PNPC (800ppc) is slightly better than ℓ_2-GLVQ (128ppc) in terms of clean accuracy, and robust accuracy for $\epsilon_2 \in \{0.1, 36/255\}$, but ℓ_2-GLVQ is better for $\epsilon_2 = 0.25$. Note that the 1-NN is significantly worse showing that learning the prototypes helps improving the performance. Nevertheless, all NPC models are not competitive with neural networks which is to be expected as the ℓ_2-distance is not a good measure for image similarity. This is why we study PNPC with the perceptual metric which achieves to clean accuracies which are higher than the one of neural networks with provable robustness guarantees.

Table 11 shows the performance of ℓ_2-NPC for multiple threat models. ℓ_2-PNPC outperforms ℓ_2-GLVQ in terms of clean accuracy, ℓ_1- and ℓ_2-robust accuracy but is worse for ℓ_∞-robust accuracy and as this is the most difficult threat model it is also worse in the union. MMR-U outperforms the ℓ_2-NPC but the margin is relatively small.

Table 11: **CIFAR10:** lower (CRA) and upper bounds (URA) on robust accuracy for multiple threat models for our ℓ_2-PNPC, the ℓ_2-NPC of (Saralajew et al., 2020), a 1-NN classifier. As comparison we show MMR-Univ of (Croce & Hein, 2020a) which is a neural network specifically trained for certifiable multiple-norm robustness.

CIFAR10	std. acc.	$\epsilon_1 = 2$		$\epsilon_2 = 0.1$		$\epsilon_\infty = 2/255$		union	
		CRA	URA	CRA	URA	CRA	URA	CRA	URA
ℓ_2-PNPC	49.2	**42.5**	42.5	41.9	41.9	32.7	32.7	32.7	32.7
ℓ_2-GLVQ	48.6	42.3	42.3	41.5	41.5	35.2	35.2	35.2	35.2
1-NN	35.7	30.0	-	29.7	29.7	22.5	-	22.5	-
MMR-U	**53.0**	36.6	43.6	**46.4**	48.1	**36.2**	36.2	**35.4**	36.2

Table 12: **MNIST:** Certified robust accuracy of networks with orthogonal convolutions. We computed robust accuracy after every epoch and the reported numbers are the maximal ones. The radius is 1.58

blocks \ γ	0	0.1	0.2	0.5	1
1	57.17	58.23	58.75	58.82	58.57
2	58.31	58.85	59.63	59.21	58.99
4	59.50	60.75	**61.02**	60.33	58.82
6	59.78	60.47	59.05	59.99	57.53

I. Comparison with orthogonal convolution networks

We evaluated the robustness orthogonal convolution networks on MNIST at radius 1.58. According to the evaluation in (Singla et al., 2022), the currently best method for orthogonal convolution networks is to combine skew orthogonal convolutions with Householder activations. According to the official repository, they suggest to choose to set the following parameters

- `--conv-layer` - We chose `soc` because it consistently outperformed baselines in the paper.
- `--activation` - We chose `hh1` activation, which is used in the experiments in the original paper.
- `--num-blocks` - We tried $1, 2, 4, 6$ blocks, possible values are $1 \dots 8$. In the original paper, it did not seem that more blocks boost performance.
- `--gamma` - We tried $0, 0.1, 0.2, 0.5, 1$. The original experiments used 0.1.
- `--lln` - The authors suggest to use last layer normalization when the number of classes is large, e.g., for CIFAR100, and do not use it for CIFAR10. We also did not use it.

We padded the MNIST images by 2 black pixels, so that we can directly use the original architecture which relied on the fact that the input images are 32×32. We also turned off the normalization by mean and variance as it is not commonly use for MNIST. We removed random horizontal flip from the set of possible augmentations, otherwise the setup is exactly as recommended. We note that the padding of MNIST image by 2 pixels is likely not the optimal way how to adapt the network to work with MNIST dataset.

The orthogonal convolutions from (Li et al., 2019) reports 56.4% certified robust accuracy. The method of (Trockman & Kolter, 2021) yielded 54% robust accuracy with the suggested setup.

I.1. Empirical robustness

We evaluated the empirical robustness of (Singla et al., 2022) using AutoAttack which is a stronger attack than what the competing methods used in Table 5. Thus, we don't conclude that orthogonal convolutions are (significantly) less empirically robust than the other evaluated methods.

7.2 Sound Randomized Smoothing in Floating-Point Accuracy

SOUND RANDOMIZED SMOOTHING IN FLOATING-POINT ARITHMETIC

Václav Voráček, Matthias Hein
Tübingen AI Center, University of Tübingen

ABSTRACT

Randomized smoothing is sound when using infinite precision. However, we show that randomized smoothing is no longer sound for limited floating-point precision. We present a simple example where randomized smoothing certifies a radius of 1.26 around a point, even though there is an adversarial example in the distance 0.8 and show how this can be abused to give false certificates for CIFAR10. We discuss the implicit assumptions of randomized smoothing and show that they do not apply to generic image classification models whose smoothed versions are commonly certified. In order to overcome this problem, we propose a sound approach to randomized smoothing when using floating-point precision with essentially equal speed for quantized input. It yields sound certificates for image classifiers which for the ones tested so far are very similar to the unsound practice of randomized smoothing. Our only assumption is that we have access to a fair coin.

1 INTRODUCTION

Shortly after the advent of deep learning, it was observed in Szegedy et al. (2014) that there exist adversarial examples, i.e., small imperceptible modifications of the input which change the decision of the classifier. This property is of major concern in application areas where safety and security are critical such as medical diagnosis or in autonomous driving. To overcome this issue, a lot of different defenses have appeared over the years, but new attacks were proposed and could break these defenses, see, e.g., (Athalye et al., 2018; Croce and Hein, 2020; Tramer et al., 2020; Carlini et al., 2019). The only empirical (i.e., without guarantees) method which seems to work is adversarial training (Goodfellow et al., 2015; Madry et al., 2018) but also there, a lot of defenses turned out to be substantially weaker than originally thought (Croce and Hein, 2020).

Hence, there has been a focus on certified robustness. Here, the aim is to produce certificates assuring no adversarial example exists in a small neighborhood of the original image. For the neighborhood, typically called threat model, one often uses ℓ_p- balls centered at the original image. However, there also exist other choices, such as Wasserstein balls (Wong et al., 2019; Levine and Feizi, 2020) or balls induced by perceptual metrics (Laidlaw et al., 2021; Voráček and Hein, 2022). The common certification techniques include (1) Bounding the Lipschitz constant of the network, see Hein and Andriushchenko (2017); Li et al. (2019); Trockman and Kolter (2021); Leino et al. (2021); Singla et al. (2022) for the ℓ_2 threat model and Zhang et al. (2022) for ℓ_∞. (2) Overapproximating the threat model by its convex relaxation (admittedly, bounding Lipschitz constant can also be interpreted this way), possibly combined with mixed-integer linear programs or SMT; see, e.g., Katz et al. (2017); Gowal et al. (2018); Wong et al. (2018); Balunovic and Vechev (2020). (3) Randomized smoothing (Lecuyer et al., 2019; Cohen et al., 2019; Salman et al., 2019), which is hitherto the only method scaling to ImageNet. Note that the concept of randomized smoothing may also be interpreted as a special case of (1), see Salman et al. (2019).

All of these certificates expect that calculations can be done with unlimited precision and do not take into account how finite precision arithmetic affects the certificates. For Lipschitz networks (1), the round-off error is of the order of the lowest significant bits of mantissa, which we can estimate to be in the orders of $\sim 10^{-8}$ for single-precision floating-point numbers. Thus, we should assume that the adversary can also inject ℓ_∞-perturbation bounded by $\sim 10^{-8}$ in every layer. However, since the networks have small Lipschitz constants by construction, those errors will not be significantly mag-

nified. Although we cannot universally quantify the numerical errors of Lipschitz networks, they will likely be very small and in particular, can be efficiently traced during the forward pass so that the certificates can be made sound. For the verification methods from category (2), previous works have shown that numerical errors may lead to false certificates for methods based on SMT or mixed-integer linear programming (Jia and Rinard, 2021; Zombori et al., 2021). However, it is possible (and often done in practice) to adapt the verification procedure to be sound w.r.t. floating-point inaccuracies (Singh et al., 2019); thus, the problem is not fundamental, and these verification techniques can be made sound. For randomized smoothing certificates (3), Jin et al. (2022) perform floating-point attacks on certifiably robust networks and indicate the existence of false certificates; see Appendix F for a discussion. The recent work of Lin et al. (2021) focuses on randomized smoothing when using only integer arithmetic in neural networks for embedded devices, so they will, by definition, not have problems with floating-point errors. On the other hand, it does not cover some modern architectures, such as transformers. Furthermore, the way the certificates are computed is derived from the continuous normal distribution; thus, the certificates are approximate, see Appendix G. Another direction is so-called derandomized smoothing - methods that remind randomized smoothing but are deterministic. See, e.g., Levine and Feizi (2021).

In this paper, we make the following contributions[1]:

1. We perform a novel analysis of numerical errors in randomized smoothing approaches when using floating-point arithmetic and identify qualitatively new problems.
2. Building on the observations, we present a simple approach for developing classifiers whose smoothed version will provide fundamentally wrong certificates for chosen points and discuss how this could be exploited in practice.
3. We propose a sound randomized smoothing procedure for floating-point arithmetic with negligible computational overhead for image classification compared to the unsound practice.

While we could not find substantial differences of our sound certificates compared to the unsound practice for our tested classifiers, a lack of a counterexample is not a proof of the correctness. The past has shown that such gaps will be exploited by malicious actors in the future. It is to be expected that certificates of adversarial robustness are required for classifiers used in safety-critical systems (see European AI act European Commission (2021)) and thus will be controlled by regulatory bodies. A malicious company could use then the problems of randomized smoothing in floating-point arithmetic to provide fake certificates on a known/leaked test set. Since our sound randomized smoothing procedure for floating-point arithmetic comes at essentially no additional cost for quantized input e.g., images, we believe that using our sound procedure should always be used for such domains.

Manuscript organization: We start with the definition of randomized smoothing in Section 2, then we continue with the introduction of floating-point arithmetic following the IEEE standard 754 (iee, 2008) in Section 3. In Section 4, we exploit the properties of floating-point arithmetic and present a simple classifier producing wrong certificates, and we follow with the identification of the implicit assumptions of randomized smoothing. In Section 5 we conclude the main result by proposing a method of sound randomized smoothing in floating-point arithmetic and provide an experimental comparison of the old unsound and the new sound certificates.

2 RANDOMIZED SMOOTHING

Throughout the paper, we consider for clarity the problem of binary classification, but every phenomenon we discuss can be easily transferred to the multiclass setting. We note that the proposed algorithmic fix, see Appendix K, as well as the experiments in A, are done for the multiclass setting.

We first introduce randomized smoothing and define certificates with respect to a norm ball.

Definition 2.1. A classifier $F : \mathbb{R}^d \to \{0, 1\}$ is said to be certifiably robust at point $x \in \mathbb{R}^d$ with radius r, w.r.t. norm $\|\cdot\|$ if the correct label at x is $y \in \{0, 1\}$, and $\|x - x'\| \leq r \implies F(x') = y$.

[1]Code is available at `https://github.com/vvoracek/Sound-Randomized-Smoothing`

One way to get such a certificate is randomized smoothing (Lecuyer et al., 2019; Cohen et al., 2019; Salman et al., 2019) which we introduce following. We are given a *base classifier* $F : \mathbb{R}^d \to \{0, 1\}$. Its smoothed version is $\hat{f}(x) = \mathbb{E}_{\varepsilon\sim\mathcal{N}(0,\sigma^2 I_d)} F(x + \varepsilon)$, and the resulting hard classifier is $\hat{F}(x) = [\![\hat{f}(x) > 0.5]\!]$, where the Iverson bracket $[\![\text{statement}]\!]$ evaluates to 1 if and only if the statement inside holds true. Using the Neyman-Pearson lemma the following result has been shown:

Theorem 2.2 ((Cohen et al., 2019)). *Let F be a deterministic or random classifier and let Φ^{-1} be the inverse Gaussian CDF. If*

$$\mathbb{P}_{\varepsilon\sim\mathcal{N}(0,\sigma^2 I)}(F(x + \varepsilon) = c_A) \geq p_A.$$

for some $p_A \in (\frac{1}{2}, 1]$, then $\hat{F}(x + \delta) = c_A$ for all $\delta \in \mathbb{R}^d$ with $\|\delta\|_2 < \sigma\, \Phi^{-1}(p_A)$.

We call in the following $r(x) = \sigma\, \Phi^{-1}(\hat{f}(x))$ the certified ℓ_2-radius of $\hat{F}$ at x.

We note we require that the output of the base classifier F to be independent of previous inputs and outputs. It is easy to construct an F violating this assumption and producing false certificates, e.g., take F that returns 0 in the first 10^6 calls and 1 afterwards. For the majority of classifiers, it is intractable to evaluate $\hat{f}(x)$ exactly; therefore, random sampling is used to estimate it and thus only a probabilistic certificate is possible where the probability that the certificate holds can be made arbitrarily close to one if one uses more samples or weakens the certificate. Following the literature, we use $100\,000$ samples to estimate $\hat{f}(x)$ and then lower bound this by p for certifying class 1 (resp. upper bound it for class 0) so that the failure probability, that is when $p > \hat{f}(x)$ (resp. $p < \hat{f}(x)$), is at most 0.001. The value of p can be computed using tail bounds or classical Clopper-Pearson confidence intervals for the binomial distribution. The actual certification procedure is described in Algorithm 1. However, to keep the example below in Listing 1 as simple as possible, we computed p using a simple Hoeffding bound which we derive in Appendix I. Although it produces a weaker certificate, it is still sufficient for the demonstration.

3 Computer representation of floating-point numbers

In this section we briefly introduce the floating-point representation and arithmetic according to standard IEEE-754 (iee, 2008). A detailed version with examples and treatment of other precisions can be found in Appendix C where we present examples in a toy, 8−bit, arithmetic. Here, we introduce only single-precision floating-point numbers.

Single-precision floating-point numbers are represented in memory as sequences of bits $x_1 x_2 \dots x_{32}$. The first bit is a sign bit, the next 8 bits determine the exponent, and the last 23 numbers determine the mansissa. The conversion in normalized form is as follows:

$$(-1)^{x_1} \cdot 2^{\left(\sum_{i=2}^{9} x_i \cdot 2^{9-i}\right)-127} \cdot \left(1 + \sum_{i=10}^{32} x_i \cdot 2^{9-i}\right).$$

We will write the floating-point operations in circles; e.g., $\oplus, \ominus$ instead of $+, -$ to distinguish them from the mathematical ones which do not suffer from rounding errors.

The addition (or analogically subtraction) of two floating-point numbers is performed in three steps. First, the number with the lower exponent is transformed to the higher exponent; then the addition is performed (we assume with infinite precision), and then the result is rounded to fit into the floating-point representation. An example is provided in C.2 in Appendix C.

Thus, it happens that $x \oplus y = x \oplus z$ for any x and some $y \neq z$. Consequently, there will exist some w such that there is no v for which $x \oplus v = w$. This is the main observation that we will built on and is treated in detail in Appendix C in Example C.3.

3.1 Connection to randomized smoothing

We have identified some unpleasant properties of floating-point arithmetic that we will exploit in sequel to provide false certificates. In particular, We will try to determine if a given number could be a smoothed version of a specific number or not. The following observations will help us.

In Reiser and Knuth (1975), it is shown that the identity $((x \oplus y) \ominus y) \oplus y = x \oplus y$ holds apart from a single y for any x. It is further shown that the identity $(((x \oplus y) \ominus y) \oplus y) \ominus y = (x \oplus y) \ominus y$ holds always true. On the other hand, there is no evidence that the equality $(x \oplus y) \ominus y = x$ should hold. Indeed, consider x to have a lower exponent than y. Then during the addition, $x \oplus y$, the low bits of the mantissa of x are lost. Similarly, if $x \oplus y$ has a different exponent than y, then a loss of significance may occur during the second rounding. Finally, consider the case where $x = a \ominus y$, then the identity $(x \oplus y) \ominus y = x$ holds.

Our idea is to make the classifier determine if the observed value x could be a smoothed version of a. This can be done precisely, but we only approximate this using the previous observation. The reason is that it is sufficient for the demonstration, and the resulting function (introduced in Equation (1) in the next section) will be simple, suggesting that the phenomenon may occur in standard networks.

3.2 Floating-Point Issues in the Context of Differential Privacy

Randomized smoothing has been motivated by differential privacy (Lecuyer et al., 2019). In differential privacy it has been shown in the seminal work of Mironov (2012) that the lowest bits of mantissa can serve as a side channel which yields a substantial discrepancy between the theoretical properties of algorithms of differential privacy, and the properties of their naive implementations, see Mironov (2012); Jin et al. (2021); Bichsel et al. (2021). Consequently, revisions of the standard differential privacy mechanisms accounting for the floating-point errors have appeared, see, e.g., Casacuberta et al. (2022); Canonne et al. (2020), and included in the framework OpenDP (Gaboardi et al., 2020).

In our construction, we utilize of the rounding errors of floating-point addition. On a high level, this is similar to what Mironov (2012) exploit. However, their procedure considers Laplacian noise and the example also exploits the Laplace distribution samplers' properties.

4 Construction of Classifiers with False Certificates

We present an example of a function $F : \mathbb{R} \to \{0, 1\}$ which is prone to giving incorrect certificates via randomized smoothing; the whole "experimental setup" is captured in Listing 1. The example is based on the observation that we are able to determine if a floating-point number x could be a result of floating-point addition $a \oplus n$ where a is known and n is arbitrary. We construct a function F_a whose behavior we analyzed in Subsection 3.1.

$$F_a(x) = [\![(x \ominus a) \oplus a = x]\!]. \tag{1}$$

We take F_a as the base classifier and consider the smoothed classifier $\hat{f}_a$ it induces with $\sigma = 0.5$. It holds that $\hat{f}_a(a) \approx 1$, therefore if we have enough samples, we may obtain a very large certified radius. Specially, in the example considered in Listing 1 with $100\,000$ samples, we can certify a ℓ_2-radius of 1.26 around point $a = 210/255$, however $0 = \hat{F}_a(0) \neq \hat{F}_a(a) = 1$, and the point 0 is nowhere near the boundary of the certified ball. In the example in 1, we use a simple Hoeffding bound I instead of the standard bounds of Clopper-Pearson. The Clopper-Pearson bounds certify robust radius 1.9.

```
import numpy as np
from scipy.stats import norm

sigma = 0.5; num_samples = 100000; alpha = 0.001
f = lambda x: (x - 210/255) + 210/255 == x
noise = np.random.randn(num_samples)*sigma

p1 = f(0+noise).sum()/num_samples                    # 0.46
p2 = f(210/255+noise).sum()/num_samples              # 1.0
p = p2-(-np.log(alpha)/num_samples/2)**0.5
r = sigma * norm.ppf(p)                              # 1.26
```

Listing 1: example of an incorrect randomized smoothing certificate

This construction does not rely on the fact that $F_a(a+\varepsilon) = 1$ for $\varepsilon \sim \mathcal{N}(0,\sigma^2)$ with very high probability, it only serves as a striking example. Similarly, we get $0 = \hat{F}_a(0) \neq \hat{F}_a(200/255) = 1$, despite every point $0, 1/255, \ldots, 255/255$ would be class 1 according to the certificate.

4.1 Consequences for image classifiers

We stress that the simple construction generalizes to images. For the remainder of the section, we consider $x \in \{0, 1/255, \ldots 255/255\}^d$ to be a vectorized image with e.g., $d = 3 \cdot 32^2 = 3072$ for CIFAR dataset. Indeed, we could employ a function

$$F_{a,i}(x) = [\![(x_i \ominus a) \oplus a = x_i]\!], \tag{2}$$

which takes a vectorized version of an image as an input. Using such function in Listing 1 would certify that any image with intensity $210/255$ at position i is class 1 with robust radius 1.26, while any image with intensity 0 at position i would be classified as 0; a clear contradiction. We take one step further. Consider a function with a parameter $a \in \mathbb{R}^d$:

$$G_a(x) = \min_{i=1}^{d} [\![(x_i \ominus a_i) \oplus a_i = x_i]\!]. \tag{3}$$

It holds that $\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,1)} G_a(a+\varepsilon) \approx 1$; thus, certifying "arbitrarily" high radius (to be specific, with $100\,000$ samples it is 3.8115 in ℓ_2 norm), and $\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,1)} G_a(a'+\varepsilon) < 0.5$ for the vast majority of inputs $a \neq a'$. We tried the following experiment; For every image a in the CIFAR10 test set, we created an image a' by increasing the image intensity of a by $1/255$ at 512 random positions. Then it holds that $\mathbb{E}_{\varepsilon\sim\mathcal{N}(0,1)} G_a(a'+\varepsilon) \leq 0.2$ for every CIFAR image a with high probability, even though $\|a - a'\|_2 < 0.09$.

Following this line of examples, let us introduce the base classifier:

$$H_A(x) = \max_{a\in A} G_a(x) = \max_{a\in A} \min_{i=1}^{d} [\![(x_i \ominus a_i) \oplus a_i = x_i]\!], \tag{4}$$

where A is a set of images. Therefore, when A is the set of CIFAR10 test set images, then we can certify the robustness of the smoothed version of H_A at every point of the CIFAR10 test set for large radii, even though it is vulnerable even to small random perturbations. We remark that H_A can be implemented with a standard network architecture using only linear layers and ReLU non-linearities. To conclude the examples, we state the findings in the upcoming proposition. Since we introduced the machinery only for binary classification, we treat CIFAR10 as a binary classification dataset. For time reasons, we (as it is common in the context of randomized smoothing) only consider 1000 test images for the upcoming proposition; the first 500 images from the test set of both classes.

Proposition 4.1. *There is a classifier with certified robust accuracy* 100% *on the first* 1000 *CIFAR10 test set images* $X \subset [0, \frac{1}{255}, \ldots, 1]^{3072}$ *(where we define class* 0 *to include classes* $0, 1, 2, 3, 4$ *of CIFAR10 and class* 1 *contains the other classes) with* ℓ_2*-robust radius of* 3 *and failure probability* 0.001 *using randomized smoothing certificates, while for every point* $x \in X$ *there is an adversarial example* x' *with* $\|x - x'\|_2 \leq 1$.

The proof can be found in Appendix D The past has shown that loopholes can and will be exploited in the future by malicious actors trying to trick certification agencies e.g. see the diesel scandal where car manufacturers detected the test in a lab to fake significantly better pollution values. As the European AI act requires a certain level of adversarial robustness in safety-critical applications, certification agency are likely to evaluate certified robustness in the future. In order to illustrate the problem, we just sketch how the fake certificates of Proposition 4.1 could be exploited. In fact let M be the classifier described in the proof of Proposition 4.1 and let $m(x) = \mathbb{E}_{\epsilon\sim\mathcal{N}(0,\sigma^2 I_d)} M(x+\epsilon)$ be the smoothed version of M. One can see that roughly $m(x) \approx \frac{1}{2}$ if $x \notin N(X)$, where $N(X)$ denotes a small neighborhood of the test set X. Given a neural network for image classification a simple way to trick the certification agency, would be a new classifier where one uses the neural network whenever $\delta \leq m(x) \leq 1-\delta$, e.g. $\delta = 0.1$, and otherwise the classifier M of Proposition 4.1. This classifier would inherit the strong *fake* robustness guarantees on the test set from M but behave like a normal classifier on any other input. We emphasize that this problem is resolved by our fix to randomized smoothing in floating point representation of Section 5 which has negligible computational overhead for image classification.

4.2 IMPLICIT ASSUMPTIONS OF RANDOMIZED SMOOTHING

The obvious questions after this negative result are: i) what is the key underlying problem in floating-point arithmetic? ii) what are the implicit assumptions in randomized smoothing?, and iii) how can we fix the problem?

The first assumption of randomized smoothing is that samples from a normal distribution are indeed i.i.d. samples. This is not true for floating-point precision due to the rounding; Thus, the resulting distribution from which we observe samples is uncontrolled, and for certification, we should not rely on it. However, violation of this assumption is not the cause of the wrong certificate in Listing 1.

The intuition behind randomized smoothing is that the distributions $D_1 = \mathcal{N}(x, \sigma^2 I)$ and $D_2 = \mathcal{N}(x + \varepsilon, \sigma^2 I)$ have significant overlap for small values of ε. As a consequence, the smoothed classifier $\hat{f}(x) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I_d)} F(x + \varepsilon)$ evaluated at x also carries information about its value at points near x. However, the following observation will prove this wrong in floating-point arithmetic.

Roughly speaking, the supports of two high dimensional normal distributions appear to be almost disjoint, although in one dimension the overlap may be substantial. To support this claim, We performed the following experiment; given point $a \in \{0, 1/255, \dots 255/255\}$ and $\sigma > 0$, find a point $b \in \{0, 1/255, \dots 255/255\}$ such that $|a - b| \leq 2/255$ which minimizes the probability that for an $\varepsilon_1 \sim \mathcal{N}(0, \sigma^2)$ there exists a number ε_2 such that $a \oplus \varepsilon_1 = b \oplus \varepsilon_2$. For example, if $a \geq 5/255$ and $\sigma = 1$, then the minimized probability is less than 0.99, and for the majority of $a \geq 5/255$ it is even smaller. In order to see that the distributions are almost disjoint, consider an image, say from a CIFAR dataset, $a \in \mathbb{R}^{3072}$ which has at least half of its channels with intensities greater than $4/255$. According to the previous observation, we can find an image a' such that $\|a - a'\|_\infty = 2/255$ and that the probability that smoothed a at any (non black) position could be a smoothed version of a' is at most 0.99 (this can be exploited by function $F_{a,i}$ from Equation (2)). Therefore, the probability that a smoothed version of the first image could also be a smoothed version of the second image is at most $0.99^{3072/2} \approx 2 \times 10^{-7}$ (this can be exploited by function G_a from Equation (3)). Thus, when we follow the standard practise and use 10^5 samples to estimate $\hat{f}(a)$ from base classifier F, the chances that at least one of the samples belongs to the distribution from which we sample to estimate $\hat{f}(a')$ is at most in the orders 10^{-2}. Consequently, without any assumptions on the base classifier F, $\hat{f}(a)$ carries almost no information about $\hat{f}(a')$.

4.3 POTENTIAL REVISIONS OF RANDOMIZED SMOOTHING

The described experiment exploits the floating-point rounding. The errors are in the order of the least significant bits, which are in the order of 10^{-8} for single-precision and 10^{-4} for half-precision. Since these numerical errors are not controlled, we should assume that the model is adversarially attacked during smoothing, where the attacker's budget is the possible rounding error, denoted as $\mathcal{B}$; therefore, the smoothing (for certifying class 1) should be performed as:

$$\hat{f}(x) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I_d)} \min_{\varepsilon_2 \in \mathcal{B}} F(x + \varepsilon + \varepsilon_2).$$

To mitigate this problem, during estimating $\hat{f}(x)$, we should certify $F(x + \varepsilon)$. Although the attacker's budget $\mathcal{B}$ is very small for single accuracy and possibly noticeable for the half accuracy, it is not clear how it should be certified, since in randomized smoothing, there are no assumptions on F.

Consider F to be a thresholded classifier $F(x) = [\![f(x) > 0.5]\!]$, where f is a neural network, then we could certify that f is constant in $\mathcal{B}$-neighbourhood of the smoothed image. For generic models, this can be done by either bounding the Lipschitz constant of f (w.r.t. an ℓ_∞-like norm), or by propagating a convex relaxation (e.g., IBP) through the network. For smoothing, there are usually used deep models. E.g., Salman et al. (2019) used ResNet110 and ResNet50 for certifying CIFAR10 and ImageNet respectively. The bound on the global Lipschitz constant of a deep network by bounding the operator norms of each layer is thus very weak ($\approx 10^{30} - 10^{130}$, depending on the model) and cannot certify $F(x + \varepsilon)$ even under such a weak threat model as the rounding errors in $\mathcal{B}$.

A possible defense against this problem would be to round the input on a significantly larger scale than $\mathcal{B}$ before evaluating F. Let the rounding be performed by a mapping g, then we would in

fact smooth a classifier $F \circ g$. If we consider $\mathcal{B}$ to be in the orders of 10^{-8} and we would round it to orders 10^{-2}, then the probability that $x + \varepsilon$ will be close to the boundary of rounding, i.e., $\exists \varepsilon_2 \in \mathcal{B} : g(x + \varepsilon + \varepsilon_2) \neq g(x + \varepsilon)$ would be on the order of 10^{-6}, which is then the probability that the attack within the threat model $\mathcal{B}$ could indeed change the input of F at a single position. Consequently, the probability that there is no $\varepsilon_2 \in \mathcal{B}$ which would change the result of rounding is very roughly $\approx (1 - 10^6)^{3072} \approx 0.997$ for CIFAR and $\approx (1 - 10^6)^{150528} \approx 0.86$ for ImageNet. This means that for approximatelly 86% of the smoothed ImageNet images we can guarantee that $F(x + \varepsilon + \varepsilon_2) = F(x + \varepsilon)$ and for the others, we could e.g., set $\min_{\varepsilon_2 \in \mathcal{B}} F(x + \varepsilon + \varepsilon_2) = 0$. This replacement of F by $F \circ g$ during smoothing seem to solve the problem for CIFAR and partially also for ImageNet for single precision. For half precision, the problem will persist.

However, even if this adjustment solved the problem with numerical errors satisfactorily during the addition of noise to images, the certificate will still not be sound because we are unlikely to control the normal distribution sampler's performance.The normal distribution samplers implementations used in standard software libraries (e.g., Ziggurat algorithm; Box-Muller transform) transform i.i.d. uniform distribution samples to i.i.d. normal distribution samples. While this is true in theory for unlimited precision; here, we perform floating-point operations. Therefore, the sampling is subject to floating-point errors and we don't observe the actual rounded samples from normal distribution.

While we do not present any example exploiting the subtle errors of the normal distribution samplers, relying on them only keeps a possible loophole in the procedure. As our goal is to propose a sound randomized smoothing procedure in floating-point arithmetic, our work would be incomplete if we addressed only some potential causes of floating-point errors and not the others, even though we think they are harder to exploit. In particular, we note that Mironov (2012) exploited the (standard) floating-point implementation of the Laplace distribution sampler in order to attack the guarantees of differential privacy.

5 Sound randomized smoothing for floating-point arithmetic

In this section, we will derive a sound randomized smoothing certification procedure for floating-point arithmetic. Our only assumption is the access to i.i.d. samples of a fair coin toss, which is equivalent to having access to samples from the uniform distribution on integers $0, \ldots, 2^n - 1$ for some n. Thus, we assume to have access to uniform samples from numbers representable by `Long` datatype, that is when $n = 64$. We further consider classification tasks where the input is quantized as it is true for images. Throughout the section, we consider the input space to be $\{0, 1, \ldots, 255\}^d$ in order to have clear notation. The generalization to other forms of quantized inputs is generic, but the generalization to real-valued inputs is a bit more involved; we move the discussion to Appendix E. The resulting algorithm is captured in Appendix K in Algorithm 2.

5.1 Certification of quantized input

As discussed in the previous section, it is appealing to quantize the smoothed images before feeding them into the network. Thus, we prepend a mapping $g_k : \mathbb{R}^d \to \{-k, -k + 1, \ldots, k + 255\}$, for some positive integer k which rounds the input to the nearest integer from its range before the function to be smoothed F; therefore, the smoothed classifier (with base classifier $F \circ g_k$) is defined as:

$$\hat{f}(x) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I_d)} F(g_k(x + \varepsilon)),$$

which we further equivalently rewrite as

$$\hat{f}(x) = \mathbb{E}_{t \sim g_k(x+\varepsilon),\, \varepsilon \sim \mathcal{N}(0, \sigma^2 I_d)} F(t).$$

This treatment is crucial for the method. Instead of adding noise to the input, which is subject to rounding errors, we sample the noised input directly.

5.2 Discretized normal distribution

It remains to show how to obtain i.i.d. samples from the discretized normal distribution

$$\mathcal{N}_D^k(x, \sigma^2) = g_k(x + \varepsilon), \quad \varepsilon \sim \mathcal{N}(0, \sigma^2).$$

We note that the discretized normal distribution is different from the discrete Gaussian distribution used in the context of differential privacy (Canonne et al., 2020).

As discussed in the two final paragraphs of Subsection 4.3, it is not enough to round samples from normal distribution since we cannot guarantee the correctness of the normal sampler. The key observation is that we do not even need to obtain samples from $\mathcal{N}(0, \sigma^2)$ anymore. The resulting distribution from which we want to sample now is discrete. Concretely, we have

$$\mathbb{P}_{t\sim\mathcal{N}_D^k(x,\sigma^2)}[\![t = a]\!] = \begin{cases} \int_{-\infty}^{-k+\frac{1}{2}} \frac{1}{\sqrt{2\pi\sigma^2}}\ e^{-\frac{(x-u)^2}{2\sigma^2}}\, du & \text{if } a = -k, \\ \int_{a-\frac{1}{2}}^{a+\frac{1}{2}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-u)^2}{2\sigma^2}}\, du & \text{if } -k < a < k + 255,\ a \in \mathbb{Z} \\ \int_{k+255-\frac{1}{2}}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-u)^2}{2\sigma^2}}\, du & \text{if } a = k + 255. \\ 0 & \text{otherwise} \end{cases}$$

Additionally, the following well-known property of normal distribution holds for the discretized normal distribution as well.

Proposition 5.1. *Let k be a positive integer and $x \in \{0, 1, \ldots, 255\}$, then it holds that $\mathcal{N}_D^k(x, \sigma^2) = \max\{-k, \min\{k + 255, t' + x\}\},\ t' \sim \mathcal{N}_D^{k+255}(0, \sigma^2)$.*

Thus, it is enough to have a sampler from $\mathcal{N}_D^{k+255}(0, \sigma^2)$. The value of k is chosen such that the vast majority of samples from $\mathcal{N}(x, \sigma^2)$ falls into the interval $[-k, k + 255]$. The choice of k does not affect the correctness of the certificates, but may affect the accuracy. In the experiments we chose $k = 6\sigma_{max} = 6$ for inputs from $[0, 1]^d$, so it corresponds to $k = 6 \cdot 255$ in the notation of this section.

5.2.1 Discretized normal distribution sampler

Let us denote the quantile function (inverse cdf) of $\mathcal{N}_D^k(0, \sigma^2)$ as $\Phi_{D,k}^{-1}$, then $\Phi_{D,k}^{-1}$ transforms i.i.d. samples from the uniform distribution on the interval $[0, 1) =: \mathcal{U}(0, 1)$ to i.i.d. samples from distribution $\mathcal{N}_D^k(0, \sigma^2)$. To sample from $\mathcal{N}_D^k(0, \sigma^2)$, we first approximate the samples from $\mathcal{U}(0, 1)$ by samples from the discrete uniform distribution on $\{0, \ldots, 2^n - 1\}$ which we will interpret as uniform distribution on $\{0, \frac{1}{2^n}, \ldots, \frac{2^n-1}{2^n}\} =: \mathbf{U}(0, 1)$. Then we only need to compute the $2k + 255$ probabilities with high enough accuracy that we can claim the correctness of $\Phi_{D,k}^{-1}(u)$ for $u \in \{0, \frac{1}{2^n}, \ldots, \frac{2^n-1}{2^n}\}$. This is ensured by using symbolic mathematical libraries allowing computations in arbitrary precision.

Since the distribution $\mathcal{N}_D^k(0, \sigma^2)$ is supported on $2k + 256$ events, there will be $2k + 255$ points $x \in \{0, \frac{1}{2^n}, \ldots, \frac{2^n-1}{2^n}\}$ such that $\Phi_{D,k}^{-1}(x) \neq \Phi_{D,k}^{-1}\left(x + \frac{1}{2^n}\right)$. Now, consider the mapping between samples from $\mathcal{U}(0, 1)$ and $\mathbf{U}(0, 1)$ which rounds down a sample u from the continuous real interval $[0, 1)$ to the closest point v from the set $\{0, \frac{1}{2^n}, \ldots, \frac{2^n-1}{2^n}\}$. Then it holds for the probability $\Phi_{D,k}^{-1}(u) \neq \Phi_{D,k}^{-1}(v) \leq \frac{255+2k}{2^n} \leq 2^{12-n}$ for a choice $k = 7.5 \times 255$. The probability that a produced sample is not the actual i.i.d. sample is thus at most 2^{12-n} at one position. Therefore, the probability that all the smoothed images, considering ImageNet sized images with shape $3 \times 224 \times 224$, out of 100 000 smoothed samples are indeed the correct i.i.d. samples from discrete normal distribution is at least $1 - 2^{46-n} > 0.999996$ for $n = 64$.

Therefore, the probability of receiving a sample that might not be the actual i.i.d. sample is negligible. Still, we can check if we receive such a potentially flawed sample x and in that case, we would set $F(x) = 0$ when certifying class 1 (resp. $F(x) = 1$ when certifying 0) for that particular sample.

5.2.2 Sampling speed of discretized normal distribution

The sampling is slightly more expensive since we need to threshold the observed uniform samples; however, this is only an implementation issue. On the other hand, it is sufficient to sample i.i.d. noise for just one image 100 000 times and reuse it for all the other images. The certificates will be valid, only the case of failure for different images will not be independent, but it is not required in the literature. As we sample just once for the whole data set, the time spent for sampling is negligible, see Table 1. We discuss the timing in detail in Appendix A.1.

Table 1: Time comparison of certification times per image of the standard randomized smoothing and the proposed sound procedure with and without reusing noise. Details in Appendix A.1. For CIFAR10 we used 100 000 random samples and for ImageNet 10 000.

dataset	standard	proposed w/ reusing	proposed w/o reusing
CIFAR10 (ResNet-110)	9.82 s	9.87s	31.70 s
ImageNet (ResNet-50)	8.65 s	8.67 s	129 s

Table 2: Certified radii for a model F smoothed with $\mathcal{N}(0, \sigma^2 I)$ on CIFAR10 test set. Evaluated on 500 images from the test set highlighting the differences. The model is ResNet-110 taken from (Salman et al., 2019), more details in Appendix A.

Sound smoothing of $F \circ g_k$ via Algorithm 2 for $k = 6$

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.12$	0.878	0.848	0.778	0.000	0.000	0.000	0.000	0.000	0.000
$\sigma = 0.25$	0.836	0.808	0.746	0.600	0.466	0.000	0.000	0.000	0.000
$\sigma = 0.5$	0.708	0.672	0.618	0.502	0.410	0.338	0.248	0.174	0.000
$\sigma = 1$	0.512	0.492	0.448	0.380	0.316	0.278	0.230	0.182	0.112

Standard smoothing of F via Algorithm 1

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.12$	0.880	0.848	0.778	0.000	0.000	0.000	0.000	0.000	0.000
$\sigma = 0.25$	0.836	0.808	0.746	0.602	0.468	0.000	0.000	0.000	0.000
$\sigma = 0.5$	0.706	0.672	0.618	0.502	0.408	0.338	0.248	0.174	0.000
$\sigma = 1$	0.516	0.492	0.448	0.378	0.316	0.278	0.230	0.182	0.110

Finally, we wrap up the observations in the following corollary:

Corollary 5.2. *Let $F : \mathbb{R}^d \to \{0, 1\}$ be a deterministic or a random function and $g_k : \mathbb{R}^d \to \{\frac{-k}{255}, \frac{-k+1}{255}, \dots \frac{k+255}{255}\}$ maps input to the closest point of its range, breaking ties arbitrarily. Then the following two functions are identical:*

$$\hat{f}_1(x) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0, \sigma^2 I_d)} F(g_k(x + \varepsilon)), \qquad \hat{f}(x) = \mathbb{E}_{t \sim \mathcal{N}_D^k(x, \sigma^2 I_d)} F(t).$$

Therefore, to certify $\hat{f}_1$ with base classifier $F \circ g_k$ using randomized smoothing, we can estimate the value of $\hat{f}$ and use it for the certification. Furthermore we can get i.i.d. samples from $t \sim \mathcal{N}_D^k(x, \sigma^2 I_d)$ with arbitrarily high precision using exact arithmetic; thus, the certificate is sound.

Remark 5.3. The empirical performance of the sound and unsound versions of randomized smoothing are essentially equivalent in practice; see Table 2 and also Appendix A for the evidence. However, it no longer incorrectly certifies the example from Listing 1, where it only certifies a radius 0.6, and the points are distant 0.82 from each other. Similarly, the smoothed classifier of M from Proposition 4.1 does not contain the universal adversarial perturbations in the certified balls around the points. See Appendix H for more details.

To summarize: we showed how to replace sampling from the normal distribution, where one cannot trace the numerical errors, by sampling from the uniform distribution on integers, where we can bound the failure probability in order to obtain high probability estimates of the output of a smoothed classifier with a prepended rounding mapping. See Algorithms 1, 2 for the comparison of the standard and the proposed certification procedure. We also provide an empirical comparison of the methods in Table 2 and in Appendix A.

6 CONCLUSION

In the paper, we described multiple simple ways how to construct models that will be certifiably robust for points of our choice using the standard randomized smoothing certification procedure, although there will be adversarial examples in their close neighborhood.

Most importantly, we provided a sound way to do randomized smoothing in floating point representation which comes at negligible cost in image classification.

ACKNOWLEDGEMENTS

The authors thank the anonymous reviewers for their comments, which helped improve the quality of the manuscript. The authors acknowledge support from the DFG Cluster of Excellence "Machine Learning – New Perspectives for Science", EXC 2064/1, project number 390727645 and the Carl Zeiss Foundation in the project "Certification and Foundations of Safe Machine Learning Systems in Healthcare". The authors are thankful for the support of Open Philanthropy.

REFERENCES

IEEE standard for floating-point arithmetic. *IEEE Std 754-2008*, Aug 2008. doi: 10.1109/IEEESTD.2008.4610935.

A. Athalye, N. Carlini, and D. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Mislav Balunovic and Martin Vechev. Adversarial training and provable defenses: Bridging the gap. In *ICLR*, 2020.

Benjamin Bichsel, Samuel Steffen, Ilija Bogunovic, and Martin Vechev. Dp-sniper: Black-box discovery of differential privacy violations using classifiers. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 391–409. IEEE, 2021.

Clément L. Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy, 2020. preprint, arXiv:2004.00010.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Sílvia Casacuberta, Michael Shoemate, Salil Vadhan, and Connor Wagaman. Widespread underestimation of sensitivity in differentially private libraries and how to fix it, 2022. preprint, arXiv:2207.10635.

Jeremy M Cohen, Elan Rosenfeld, and J Zico Kolter. Certified adversarial robustness via randomized smoothing. In *NeurIPS*, 2019.

Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

European Commission. Regulation for laying down harmonised rules on AI. *European Commission*, 2021. URL `https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:52021PC0206&from=EN`.

Marco Gaboardi, Michael Hay, and Salil Vadhan. A programming framework for opendp. *Manuscript, May*, 2020.

I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

Sven Gowal, Krishnamurthy Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv:1810.12715*, 2018.

M. Hein and M. Andriushchenko. Formal guarantees on the robustness of a classifier against adversarial manipulation. In *NeurIPS*, 2017.

Kai Jia and Martin Rinard. Exploiting verified neural networks via floating point numerical error. In *International Static Analysis Symposium*, 2021.

Jiankai Jin, Eleanor McMurtry, Benjamin IP Rubinstein, and Olga Ohrimenko. Are we there yet? timing and floating-point attacks on differential privacy systems. *arXiv preprint arXiv:2112.05307*, 2021.

Jiankai Jin, Olga Ohrimenko, and Benjamin IP Rubinstein. Getting a-round guarantees: Floating-point attacks on certified robustness. *arXiv preprint arXiv:2205.10159*, 2022.

Guy Katz, Clark Barrett, David L Dill, Kyle Julian, and Mykel J Kochenderfer. Reluplex: An efficient smt solver for verifying deep neural networks. In *CAV*, 2017.

Cassidy Laidlaw, Sahil Singla, and Soheil Feizi. Perceptual adversarial robustness: Defense against unseen threat models. In *ICLR*, 2021.

Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy (SP)*, 2019.

Klas Leino, Zifan Wang, and Matt Fredrikson. Globally-robust neural networks. In *ICML*, 2021.

Alexander Levine and Soheil Feizi. Wasserstein smoothing: Certified robustness against wasserstein adversarial attacks. In *AISTATS*, 2020.

Alexander J Levine and Soheil Feizi. Improved, deterministic smoothing for L_1 certified robustness. In *ICML*, 2021.

Qiyang Li, Saminul Haque, Cem Anil, James Lucas, Roger B Grosse, and Jörn-Henrik Jacobsen. Preventing gradient attenuation in lipschitz constrained convolutional networks. *NeurIPS*, 2019.

Haowen Lin, Jian Lou, Li Xiong, and Cyrus Shahabi. Integer-arithmetic-only certified robustness for quantized neural networks. In *ICCV*, 2021.

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Valdu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Ilya Mironov. On significance of the least significant bits for differential privacy. In *Proceedings of the 2012 ACM conference on Computer and communications security*, pages 650–661, 2012.

John F. Reiser and Donald E. Knuth. Evading the drift in floating-point addition. *Information Processing Letters*, 3(3):84–87, 1975.

Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. *NeurIPS*, 2019.

Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. In *POPL*, 2019.

Sahil Singla, Surbhi Singla, and Soheil Feizi. Improved deterministic l_2 robustness on CIFAR-10 and CIFAR-100. In *ICLR*, 2022.

C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020.

Asher Trockman and J Zico Kolter. Orthogonalizing convolutional layers with the cayley transform. In *ICLR*, 2021.

Václav Voráček and Matthias Hein. Provably adversarially robust nearest prototype classifiers. In *ICML*, 2022.

Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J. Zico Kolter. Scaling provable adversarial defenses. In *NeurIPS*, 2018.

Eric Wong, Frank Schmidt, and Zico Kolter. Wasserstein adversarial examples via projected sinkhorn iterations. In *ICML*, 2019.

Andrew C. Yao. Theory and application of trapdoor functions. In *Symposium on Foundations of Computer Science*, 1982.

Bohang Zhang, Du Jiang, Di He, and Liwei Wang. Boosting the certified robustness of l-infinity distance nets. In *ICLR*, 2022.

Dániel Zombori, Balázs Bánhelyi, Tibor Csendes, István Megyeri, and Márk Jelasity. Fooling a complete neural network verifier. In *ICLR*, 2021.

A EXPERIMENTS

To run the experiments, we used the publicly available codebase of Salman et al. (2019) which is distributed under MIT licence. Our modifications will be publicly available under MIT licence. The experiments were run on a single Tesla V100 GPU. The models we evaluated were chosen arbitrarily from the models Salman et al. (2019) provide in their repository. Their identifications are:

pretrained_models/cifar10/finetune_cifar_from_imagenetPGD2steps/PGD_10steps_30epochs_multinoise/2-multitrain/eps_64/cifar10/resnet110/noise_σ/checkpoint.pth.tar,

pretrained_models/cifar10/PGD_4steps/eps_255/cifar10/resnet110/noise_σ/checkpoint.pth.tar

pretrained_models/cifar10/PGD_4steps/eps_512/cifar10/resnet110/noise_σ/checkpoint.pth.tar

where $\sigma \in \{0.12, 0.25, 0.50, 1.00\}$ for tables 2, 3 and 4 respectively. In Table 2, $100\,000$ samples are used, whereas for Tables 3, 4 we used only $10\,000$ samples to evaluate the smoothed classifier.

For Imagenet experiments, we used models:

pretrained_models/imagenet/replication/resnet50/noise_σ/checkpoint.pth.tar,
pretrained_models/imagenet/DNN_2steps/imagenet/eps_512/resnet50/noise_σ/checkpoint.pth.tar

where $\sigma \in \{0.25, 0.50, 1.00\}$ for tables 5 and 6 respectively. Again, we used $10\,000$ samples to evaluate the smoothed classifier.

A.1 SPEED

The speed is essentially equal for both of the methods, described in Algorithm 1 and 2 respectively because we compute the noise beforehand and then we can use the same set of n noises for every image, where n is the number of samples used to evaluate a smoothed classifier. The time needed to generate the noise is in the order of minutes; thus, negligible compared to the time needed to run the experiments.

To be more precise; we run the experiment on a GPU Tesla V100. For CIFAR10, The (standard) time per image for $100\,000$ samples used to evaluate the classifier is 9.82 ± 0.05s. If we precompute the noise batches (batch size 1000, thus we have 100 batches) and save them to files. Then with every image and every batch we load the corresponding noise, the time is then 10.47 ± 0.03s per image. The advantage of this approach is that the change of the codebase is minimal. In our case, we changed two lines of code of Salman et al. (2019) and added one class. The disadvantage is that we do a lot of unnecessary work by loading the same batch of noises multiple times. Finally, if we compute a batch of noises and evaluate the classifier for every test-set point with this noise before sampling a new batch, we get to average time 9.87s If we compute noises for every image separately, the per image time is 31.70 ± 0.15. The $\pm$ denotes standard deviation of time per image. Thus, it is not applicable for the second to last last case. The reported times are from experimental setup of 2 with $\sigma = 0.12$, $k = 1$ (ResNet110). For these experiments, we used (vectorized) procedure analogical to the one in Algorithm 2. It takes 4 minutes to compute the breaking points with SymPy library using exact arithmetic evaluated with sufficient precision (on a single core). Here, we assumed that `torch.randint` produces i.i.d. samples.

For ImageNet, we used the model from Table 5 with $\sigma = 0.5$, $k = 3$ (ResNet50). We used batch size 100, $10\,000$ smoothed versions to evaluate the per-image times follow:. The time of the standard method 8.65 ± 0.07s, the time of the reloading reuse is 12.36 ± 0.07s. The time when we sample noise and evaluate every image on that noise is 8.67s. New noises for every image yields 129 ± 1.29s. The time to compute breaking points is about 6 minutes (again, single core).

Table 3: Certified radii for a model F smoothed with $\mathcal{N}(0, \sigma^2 I)$ on CIFAR10 test set. Evaluated on 500 images from the test set highlighting the differences. The model is ResNet-110 taken from (Salman et al., 2019). See Appendix A for the details.

Sound smoothing of $F \circ g_k$ via Algorithm 2 for $k = 6$

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.12$	0.714	0.678	0.630	0.000	0.000	0.000	0.000	0.000	0.000
$\sigma = 0.25$	0.660	0.636	0.590	0.526	0.444	0.000	0.000	0.000	0.000
$\sigma = 0.50$	0.554	0.538	0.500	0.442	0.386	0.340	0.284	0.204	0.000
$\sigma = 1$	0.440	0.428	0.402	0.368	0.332	0.292	0.248	0.202	0.160

Standard smoothing of F via Algorithm 1

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.12$	0.712	0.678	0.630	0.000	0.000	0.000	0.000	0.000	0.000
$\sigma = 0.25$	0.664	0.638	0.592	0.522	0.444	0.000	0.000	0.000	0.000
$\sigma = 0.50$	0.556	0.540	0.500	0.440	0.386	0.338	0.284	0.194	0.000
$\sigma = 1$	0.440	0.428	0.402	0.366	0.330	0.290	0.248	0.204	0.164

Table 4: Certified radii for a model F smoothed with $\mathcal{N}(0, \sigma^2 I)$ on CIFAR10 test set. Evaluated on 500 images from the test set highlighting the differences. The model is ResNet-110 taken from (Salman et al., 2019). See Appendix A for the details.

Sound smoothing of $F \circ g_k$ via Algorithm 2 for $k = 6$

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.12$	0.560	0.544	0.522	0.000	0.000	0.000	0.000	0.000	0.000
$\sigma = 0.25$	0.534	0.514	0.492	0.450	0.408	0.000	0.000	0.000	0.000
$\sigma = 0.50$	0.466	0.458	0.440	0.414	0.378	0.342	0.306	0.258	0.000
$\sigma = 1$	0.370	0.364	0.342	0.320	0.298	0.276	0.252	0.226	0.166

Standard smoothing of F via Algorithm 1

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.12$	0.558	0.544	0.522	0.000	0.000	0.000	0.000	0.000	0.000
$\sigma = 0.25$	0.534	0.514	0.492	0.450	0.410	0.000	0.000	0.000	0.000
$\sigma = 0.50$	0.464	0.458	0.440	0.414	0.380	0.340	0.308	0.264	0.000
$\sigma = 1$	0.372	0.364	0.344	0.318	0.298	0.274	0.250	0.222	0.168

Table 5: Certified radii for a model F smoothed with $\mathcal{N}(0, \sigma^2 I)$ on Imagenet test set. Evaluated on 1000 images from the test set highlighting the differences. The modelis ResNet-50 taken from (Salman et al., 2019). See Appendix A for the details.

Sound smoothing of $F \circ g_k$ via Algorithm 2 for $k = 12$

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.25$	0.661	0.636	0.614	0.559	0.498	0.000	0.000	0.000	0.000
$\sigma = 0.50$	0.597	0.586	0.549	0.509	0.460	0.428	0.383	0.330	0.000
$\sigma = 1$	0.447	0.438	0.424	0.390	0.365	0.344	0.319	0.299	0.238

Standard smoothing of F via Algorithm 1

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.25$	0.660	0.635	0.614	0.559	0.497	0.000	0.000	0.000	0.000
$\sigma = 0.50$	0.598	0.584	0.548	0.507	0.459	0.429	0.385	0.323	0.000
$\sigma = 1$	0.447	0.439	0.424	0.390	0.365	0.344	0.320	0.297	0.240

Table 6: Certified radii for a model F smoothed with $\mathcal{N}(0, \sigma^2 I)$ on Imagenet test set. Evaluated on 1000 images from the test set highlighting the differences. The model is ResNet-50 taken from (Salman et al., 2019). See Appendix A for the details.

Sound smoothing of $F \circ g_k$ via Algorithm 2 for $k = 12$

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.25$	0.672	0.642	0.592	0.505	0.393	0.000	0.000	0.000	0.000
$\sigma = 0.50$	0.580	0.566	0.534	0.484	0.425	0.378	0.331	0.268	0.000
$\sigma = 1$	0.448	0.439	0.416	0.379	0.348	0.327	0.299	0.266	0.210

Standard smoothing of F via Algorithm 1

certified radius	0	0.1	0.25	0.5	0.75	1	1.25	1.5	2.0
$\sigma = 0.25$	0.672	0.641	0.593	0.503	0.389	0.000	0.000	0.000	0.000
$\sigma = 0.50$	0.581	0.564	0.534	0.486	0.423	0.377	0.337	0.270	0.000
$\sigma = 1$	0.449	0.440	0.418	0.380	0.349	0.324	0.299	0.268	0.211

B PROOF OF PROPOSITION5.1

Let us inspect the probability of observing some $a \in \{-k+1, \dots, k+254\}$. In that case $\mathbb{P}_{t\sim\mathcal{N}_D^k(x,\sigma^2)}[\![t = a]\!] = \int_{a-0.5}^{a+0.5} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-u)^2}{2\sigma^2}} du$. For the other distribution it holds that $t' = a - x$ and $\mathbb{P}_{t\sim\mathcal{N}_D^k(0,\sigma^2)}[\![t = a - x]\!] = \int_{a-x-0.5}^{a-x+0.5} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{u^2}{2\sigma^2}} du$ and the change of the variable $u \to v - x$ concludes the proof of this case. The other two cases are analogical. □

C FLOATING-POINT NUMBERS

In this appendix, we introduce the floating-point representations and arithmetic according to standard IEEE-754 (iee, 2008). For the sake of clarity, in this section, we use 8-bit floating-point number representation instead of the usual $16, 32, 64$ bits, respectively for half, single, and double precision. This appendix is self-contained and repeats the contains also (in a more detailed way) the content presented in the main paper.

Floating-point numbers are represented in memory using three different sequences of bits. The split is $1/3/4$ for the 8-bit example, $1/5/10$ for half-precision, $1/8/23$ for standard single precision, and $1/11/52$ for double precision. The modern GPUs use most often single-precision floating point numbers. We will represent the binary numbers as binary strings of 8 numbers and start with an example translating binary floating-point representation to the standard decimal one.

Example C.1. Consider the binary number $1110\ 1010$. The *first bit is the sign bit*. The number is negative iff the bit is set to 1. In our example, the bit is 1; thus, the number is negative. The *next* 3 *bits* (110) *determine the exponent*. It is the integer value of this encoding minus 3, thus, in our example, the exponent is $6-3 = 3$. The subtraction of 3 enables that one can represent exponents $-3, -2, \dots, 4$. The *last sequence is called mantissa* and encodes the number after the decimal point. There is also an implicit (not written) 1 before it. This is a so-called normalized form. Thus, the encoded value of the mantissa is 1.1010 in binary representation, which is $1{+}1{\cdot}0.5{+}0{\cdot}0.25{+}1{\cdot}0.125{+}0{\cdot}0.0625 = 1.625$ in base 10. The represented number is thus $-1.625 \cdot 2^3 = -13$ in base 10.

C.1 SUBNORMAL NUMBERS, NANS AND INFS

We note that the introduced floating-point representation is not able to represent 0 and the smallest representable positive number is $0000\ 0000$ which is 0.125 in base 10. To represent even smaller numbers, there are so-called subnormal numbers. That is, whenever the exponent consists only of zeros, there is no implicit 1 in the mantissa, but the exponent is higher by one. That is, the exponents represented by bits 000 and 001 both correspond to -2. If our 8 bit toy arithmetic also used subnormal numbers, then $0000\ 0000$ would be 0 and $000\ 0001$ would be $(0 \cdot 1 + 0 \cdot 0.5 + 0 \cdot 0.25 + 0 \cdot 0.125 + 1 \cdot 0.0625) \cdot 2^{-2} = 0.015625$. We note that there is a positive and a negative zero (and also inf).

Similarly, floating-point numbers whose exponents consist only of ones are special. If additionally the mantissa is all zeros, then it represents inf and if the mantissa contains a non-zero bit, then it represents not-a-number (NaN) and the set bits correspond to error messages. ¨

C.2 OPERATIONS WITH FLOATING-POINT NUMBERS

To distinguish the mathematical operations (infinite precision) from the computer arithmetic ones, we will use $\oplus, \ominus$ instead of $+, -$ to represent floating-point operations. When writing, e.g., $5 \oplus 7 = 12$, we mean that the floating-point representation of 5 added to the floating-point representation of 7 results in a floating-point 12. We also note that $a \oplus -b = a \ominus b$.

The addition (or analogically subtraction) of two floating-point numbers is performed in three steps. First, the number with the lower exponent is transformed to the higher exponent; then the addition is performed (we assume with infinite precision), and then the result is rounded to fit into the floating-point representation. The standard allows for several rounding schemes, but the common one is to round to the closest number breaking the ties by rounding to the number with mantissa ending with 0.

For the sake of completeness, we also mention the multiplication of floating-point numbers. The multiplication is done in a way that exponents are added; the mantissas are multiplied and consequently normalized. We will not use (nontrivial) floating-point multiplication in our constructions.

Let us show an example of the floating-point addition.

Example C.2. Consider the addition of binary numbers, $1110\ 1010$, and $0101\ 0011$. The first one we already decoded as -1.1010×2^3 and the other one is 1.0011×2^2; both in base 2.

$$1110\ 1010 \oplus 0101\ 0011 = -1.1010 \times 2^3 + 1.0011 \times 2^2 = -1.1010 \times 2^3 + 0.10011 \times 2^3,$$
$$= -1.00001 \times 2^3 \approx -1.0000 \times 2^3 = 1110\ 0000.$$

In base 10, we would have $-13 \oplus 4.75 = -8$ due to the loss of the least significant bits. This happened even though the exponents were different by the smallest possible difference.Consider further $6.5 \oplus 4.75 = 11$; Here, the loss of precision appeared even with equal exponents.

Unsurprisingly, it also holds that $6.5 \oplus 4.5 = 11$. Therefore, the addition to 6.5 is not injective and, as a consequence, it is not surjective. Connecting this to randomized smoothing, we know that there are numbers which cannot be smoothed from 6.5 as the following example shows.

Example C.3. When observing 2.125, it could not arise as $6.5 \oplus a$ for any a. Indeed, $6.5 \oplus -4.5 = 2$, while $6.5 \oplus -4.25 = 2.25$. The representations are: $6.5 \sim 0101\ 1010$, $2.125 \sim 0100\ 0001$, $-4.25 \sim 1101\ 0001$ and $-4.5 \sim 1101\ 0010$. Here -4.25 is the smallest number bigger than -4.5. Note again that the exponents of 6.5 and 2.25 differ only by the smallest possible difference.

Another consequence is that floating-point addition is not associative. That is, the following identity does not always hold $(a \oplus b) \oplus c = a \oplus (b \oplus c)$.

Example C.4. Consider the numbers $a = 2.375 \sim 0100\ 0011$, $b = 3.75 \sim 0100\ 1110$, and $c = 3.25 \sim 0100\ 1010$. Then $a \oplus b = 6 \sim 0101\ 1000$ and $(a \oplus b) \oplus c = 9 \sim 0110\ 0010$. On the other hand, $b \oplus c = 7 \sim 0101\ 1100$ and $a \oplus (b \oplus c) = 9.5 \sim 0110\ 0011$; thus, the triple a, b, c serves as a counterexample for associativity of $\oplus$.

D Proof of Proposition 4.1

Proposition D.1. *There is a classifier with certified robust accuracy* 100% *on the first* 1000 *CIFAR10 test set images* $X \subset [0, \frac{1}{255}, \dots, 1]^{3072}$ *(where we define class* 0 *to include classes* $0, 1, 2, 3, 4$ *of CIFAR10 and class* 1 *contains the other classes) with* ℓ_2*-robust radius of* 3 *and failure probability* 0.001 *using randomized smoothing certificates, while for every point* $x \in X$ *there is an adversarial example* x' *with* $\|x - x'\|_2 \leq 1$.

Proof. We take $X_0 \subseteq X$ to be the set of all images from X with class 0 and $X_1 = X \setminus X_0$. Then we construct a hard classifier

$$M(x) = \begin{cases} 1 & \text{if } H_{X_1}(x) = 1 \text{ or } (H_{X_0}(x) = 0 \text{ and } x_1 > \frac{127}{255}), \\ 0 & \text{otherwise,} \end{cases}$$

where we use H_A from (4). Experimentally, we conclude that for the smoothed classifier of M with $\sigma = 1$, randomized smoothing certifies robust radius 3 in ℓ_2 norm for every point x of the test set. At the same time, the perturbation $p = (\alpha \cdot \frac{240}{255}, \frac{-1}{255}, \frac{-1}{255}, \dots) \in \mathbb{R}^{3072}$, where α is 1 when looking for adversarial perturbation of class 0 and -1 otherwise are universal adversarial perturbations. It holds for $x \in X$ that $\hat{M}(x)$ is correct; $\hat{M}(x) \neq \hat{M}(x + p)$, and also $\|p\|_2 \leq 1$. □

E Certification of real-valued inputs or different quantizations

Here we shall discuss the adaptation of the method to real-valued inputs, and also to other quantization levels.

E.1 GENERALIZATION TO OTHER QUANTIZATION LEVELS

We described the method assuming 256 quantization levels and later we rounded the input to exact the same levels. This choice was arbitrary. The motivation was that the more fine-grained the levels are, the more similar the less changes will the rounding introduce. However, when choosing the quantization level after rounding, one should have in mind that the efficiency of the method relies on Proposition 5.1. It is easy to see that in general, we need to compute approximately $\mathrm{lcm}(q_{in}, q_{out})$ integrals to be able to perform sampling, when q_{in}, q_{out} are the respective inverse distances between neighbouring quantization levels of input, and after rounding. E.g., If we decided to round the input on scale $1/256$ instead of $1/255$, we would need to compute 256 times more integrals than when the quantization with the current rounding.

E.2 GENERALIZATION TO REAL-VALUED INPUTS

Here we propose two potential generalizations of the method to real-valued inputs. The first produces maximal (in the sense of NP lemma) certificates at the cost of slow execution. The other brings no slow down, but the certificates will not be maximal.

As discussed in the previous subsection, Proposition 5.1 is crucial for the efficiency. However, with the real-valued inputs, it cannot be taken advantage of, because the inputs will (in general) be arbitrarily distant from 0. Thus, we would need to compute the samples from $\mathcal{N}_D^k(x, \sigma^2 I_d)$ independently for every input. While it is not a fundamental problem and we, in principle, can do so, it will become slow for high-dimensional images. To soften this problem, we can decide to have small number of quantization levels (e.g., 2) so we would need to compute just a single integral per input dimension. Using 2 quantization levels was already considered in Levine and Feizi (2021) and it yields the state-of-the-art ℓ_1 robustness.

Another possible solution is to round the input before the certification. Let $x \in \mathbb{R}^d$ be the input point, we chose a quantization grid and its closest element to x is x'. Then if we certify robust radius r around x', it implies that the robust radius centered at x is at least $r - \|x - x'\|$. For instance, considering CIFAR10 dimension $d = 3072$ and the distance between quantization points is $1/1000$, then $\|x - x'\|_2 \leq \left(3072 * 1/2000^2\right)^{1/2} \approx 0.055$. At the same time, we know that the certified radius at x' would be at-most $r + \|x - x'\|$. Thus, the error is controlled and not significant. We can use more fine-grained quantization to make this error even smaller.

Finally, we note that the main focus of this paper is on quantized input as used in image classification. We expect that there are more sophisticated solutions to this problem e.g., by combining both proposed variants with rejection sampling. We leave this to future work.

F GETTING A-ROUND GUARANTEES: FLOATING-POINT ATTACKS ON CERTIFIED ROBUSTNESS

In this appendix we discuss the floating-point attacks on randomized smoothing certificates of Jin et al. (2022). We could not reproduce the result which could be due to the following problems:

- The certificates of randomized smoothing are w.r.t. the smoothed classifier which is impossible to evaluate and we approximate it by random sampling. Thus, if we certify robust radius r at point x for classifier f, then if some x_1, $\|x - x_1\| <= r$ should be considered as an adversarial example (with high probability), we should also ensure that $f(x_1) = f(x)$ with high probability. That is, from the certification procedure we know $f(x)$ with high probability, but to know $f(x_1)$ with high probability, one has to determine confidence intervals e.g. using Clopper-Pearson or Hoeffding's inequality. However, in the paper, on bottom of page 13, there is written: `Given an instance x, the smoothed classifier g runs the base classifier f on M noise corrupted instances of x, and returns the top class kA that has been predicted by f.` This suggests that only a majority vote is performed; but to claim that one has found with high probability an adversarial samples not only the majority vote has to be wrong but there needs a significant gap between wrong and correct

class. Otherwise the result might just be bad luck due to random sampling and not indicate the existence of an adversarial sample.

- During the attack, they iteratively "refine" the adversarial perturbation. Thus, they evaluate the smoothed classifier multiple times. Since the outputs are probabilistic, when one tries enough candidates, one should in principle (incorrectly) find a "high probability" adversarial example just because one will be "lucky" with the randomness. This seems not to be taken into account in their method.

If their method is indeed a successful attack on randomized smoothing in floating point arithmetic, then this just emphasizes the need for a fix, which is exactly what we propose in this paper and overcomes the possibility of such an attack.

G INTEGER-ARITHMETIC-ONLY CERTIFIED ROBUSTNESS FOR QUANTIZED NEURAL NETWORKS

Here we describe why the technique of Lin et al. (2021) for sampling from the discrete normal distribution and the consequent certification is not sufficient for our purposes.

The definition of the discrete normal distribution from Lin et al. (2021) (coinciding with the one in Canonne et al. (2020)) is as follows:

$$\mathbb{P}_{x\in\mathcal{N}_H(\mu,\sigma^2)}[\![x=a]\!] = Ze^{-\frac{(a-\mu)^2}{2\sigma^2}},$$

where Z is an appropriate normalization constant and the distribution is supported on the set of integers. For the certification, similarly to the standard smoothing, first, the lower bound p on the probability of the correct class for the smoothed classifier is estimated. Then, the robust radius is computed as $\sigma\Phi^{-1}_{\mathcal{N}_H}(p)$, where $\Phi^{-1}_{\mathcal{N}_H}$ is the inverse CDF of discrete Gaussian. This can be seen at the very bottom of the second column on page 4 in Lin et al. (2021). Note, in Algorithm 1 of Lin et al. (2021) there is written only Φ^{-1}, which according to the neighbouring discussions (and according to Thm 3.2 there) corresponds to $\Phi^{-1}_{\mathcal{N}_H}$. This certified radius is clearly not exact, because the possible certified radii can only be σ multiples of the quantization levels because the smoothing distribution is discrete. For $\sigma = 1$, the possible robust radii are $0, 1, 2, \dots$, while the actual robust radius may clearly be non-integral which makes sense even when considering quantized inputs; e.g., consider the perturbation $(1, 1, 0, \dots)$ which has distance $\sqrt{2}$. Therefore, the smoothing as described in Lin et al. (2021) is restricted to certify only integer radii which is a significant restriction.

However, we were not able to verify the correctness of the method proposed in Lin et al. (2021). In the proof of Theorem 3.1, there is: "Notice that that $p^{lb}_{c_A} = \mathbb{P}[X \in \mathcal{S}_A]$, where $\mathcal{S}_A = \{z : \langle z - x, \delta\rangle \leq \sigma \|\delta\|_2 \Phi^{-1}_{\mathcal{N}_\mathbb{H}}(p^{lb}_{C_A})\}$". Where $p^{lb}_{c_A}$ is the lower bounded probability of the target class. However, since the smoothing distribution is discrete, the function $\Phi^{-1}_{\mathcal{N}_\mathbb{H}}$ is piecewise constant, therefore there are probabilities $p_1 \neq p_2$ with $\Phi^{-1}_{\mathcal{N}_\mathbb{H}}(p_1) = \Phi^{-1}_{\mathcal{N}_\mathbb{H}}(p_2)$, thus they will both generate the same set $\mathcal{S}_A$, but using the stated fact in the proof, it would yield $p_1 = \mathbb{P}[X \in \mathcal{S}_A] = p_2$, which is absurd. Since the (incorrect) fact ($p^{lb}_{c_A} = \mathbb{P}[X \in \mathcal{S}_A]$, where $\mathcal{S}_A = \{z : \langle z - x, \delta\rangle \leq \sigma \|\delta\|_2 \Phi^{-1}_{\mathcal{N}_\mathbb{H}}(p^{lb}_{C_A})\}$) is given without a proof, we cannot rely on the correctness of the method. Even assuming the correctness of the method, the produced certificates cannot be, as discussed above, exact and are only lower-bounds on the actual robust radius certifiable by randomized smoothing.

H DISCUSSION ON PRESENTED MALICIOUS EXAMPLES

To reproduce the malicious examples from Section 4, it is important to carry every computation in the same precision. We tested it for both, single and double-precision, it likely holds also for the half-precision. If some calculations are done in single, and some in double precision, then the claimed results will probably not hold. Specially, if some calculation is performed in the single precision (e.g., transforming images from $\{0, 1, \dots, 255\}$ to $\{0, 1/255, \dots, 1\}$), casting it to double precision afterwards is not sufficient because the low mantissa bits are already lost. Although the codes are very simple, we enclose some of the snippets in the supplementary materials. Proposition 5.1 is

verified by a C++ program. We believe that for the demonstration it is sufficient to run only 100 noises per sample. With $100\,000$ samples, it is sufficient to observe $99\,900$ successes to claim robust radius 3 with probability 99.9%, and $50\,500$ successes to claim with probability 99.9% that the result is class 1. However, the runtime is about one day on a 64 core machine.

I HOEFFDING'S BOUND

Proposition I.1 (Hoeffding's inequality)**.** *Let $X_1, \dots X_n$ be random variables with $\frac{1}{n}\mathbb{E}[\sum_{i=1}^{n} X_i] = \mu$ and $0 \le X_i \le 1$. Then it holds that*

$$\mathbb{P}\left(\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) - \mu \ge t\right) \le e^{-2t^2 n}.$$

for any $t \ge 0$.

We rewrite the inequality as

$$\mathbb{P}\left(\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) - t \ge \mu\right) \le e^{-2t^2 n}.$$

Now the lhs stands for the probability that when we subtract t from the average, it will still be bigger than the mean. This is the failure case of randomized smoothing that we want to allow only with probability $\alpha = 0.001$. Thus, we want to compute t - how much do we subtract from the average.

$$\begin{aligned} e^{-2t^2 n} &= \alpha \\ -2t^2 n &= \ln(\alpha) \\ t^2 &= \frac{\ln(\alpha)}{-2n} \\ t &= \sqrt{\frac{\ln(\alpha)}{-2n}} \end{aligned}$$

J PSEUDO RANDOM NUMBERS

Here we discuss the issues regarding the random number generators. In reality, we don't have true random number generators, we only have pseudo-random number generators. In order to estimate the quantity $\hat{f}(x) = \mathbb{E}_{\varepsilon \sim \mathcal{N}(0,\sigma^2 I_d)} F(x+\varepsilon)$ with probabilistic guarantees, we need the actual random numbers. Otherwise, the probabilistic statement does not make sense (apart from trivial cases). Thus, a reasonable alternative is to require that no statistical test would distinguish in polynomial time between the generated pseudo-random numbers and the actual random numbers with non-negligibly better probability than chance. This property is guaranteed by so-called cryptographically secure random number generators (Yao, 1982).

K ALGORITHMS

Here we compare the actual algorithms of the standard randomised smoothing in Algorithm 1, and of the proposed method in Algorithm 2. The differences in the methods of the same name are highlighted by colors. The algorithms assume input to be in $\{0, 1/255, \dots, 1\}$. We emphasize that our method is a simple extension (differences highlighted) of the standard randomized smoothing, where the two additional procedures in Algorithm 2 can be evaluated just once before the certification; thus, they do not slow down the method, neither it decreases the accuracy.

Algorithm 1 Randomized smoothing certification of Cohen et al. (2019)

procedure SAMPLEUNDERNOISE(f, x, n, σ)
    counts $\leftarrow [0, 0]$
    for $i \leftarrow 1, n$ **do**
        $\varepsilon \leftarrow \mathcal{N}(0, \sigma^2 I)$
        $x' \leftarrow x + \varepsilon$
        if $f(x') > 0.5$ **then**
            counts$[1] \leftarrow$ counts$[1] + 1$
        else
            counts$[0] \leftarrow$ counts$[0] + 1$
    return counts

procedure CERTIFY($f, \sigma, x, n_0, n, \alpha$)
    counts0 $\leftarrow$ SAMPLEUNDERNOISE(f, x, n_0, σ)
    $\hat{c}_A \leftarrow$ top index in counts0
    counts $\leftarrow$ SAMPLEUNDERNOISE(f, x, n, σ)
    $p_A \leftarrow$ LOWERCONFBOUND(counts$[\hat{c}_A], n, 1 - \alpha$)
    if $p > \frac{1}{2}$ **then**
        return prediction $\hat{c}_A$ and radius $\sigma \Phi^{-1}(p)$
    else
        return ABSTAIN

Algorithm 2 Sound randomized smoothing certification of $F \circ g_k$

procedure PRECOMPUTE ARRAY OF BREAKING POINTS(k, σ^2) ▷ This function is evaluated only once
  $\texttt{arr} \leftarrow [0, \ldots, 0]$ ▷ Array of $2 \cdot 255 \cdot (k+1) + 1$ zeros
  for $i = -255k - 255, 255k + 254$ **do**
    $\texttt{arr}[255k - 255 + i] \leftarrow \lceil 2^{64} \int_{-\infty}^{(i+0.5)/255} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \, dx \rceil$
  $\texttt{arr}[2 \cdot 255 \cdot (k+1)] \leftarrow 2^{64}$
  return $\texttt{arr}$

procedure $\mathcal{N}_D^{k+1}(0, \sigma^2 I)$ ▷ This function is evaluated only on the first call with given arguments and the result is memorized. The relevant arguments are k, σ.
  $\texttt{arr} \leftarrow$ Precomputed array of breaking points for $\mathcal{N}_D^k(0, \sigma^2)$
  $\varepsilon \leftarrow [0, \ldots, 0]$ ▷ Array of d zeros
  for $i \leftarrow 1, d$ **do**
    $t \leftarrow \mathbf{U}(0, 2^{64-1})$
    for $j \leftarrow -255(k+1), 255(k+1)$ **do**
      if $\texttt{arr}[j + 255k] = t$ **then**
        return $\texttt{Failure}$
      else if $\texttt{arr}[j + k] > t$ **then**
        $\varepsilon_i \leftarrow j/255$
        Break
  return ε

procedure SAMPLEUNDERNOISE(f, x, n, σ, k)
  $\texttt{counts} \leftarrow [0, 0]$
  for $i \leftarrow 1, n$ **do**
    $\varepsilon \leftarrow \mathcal{N}_D^{k+1}(0, \sigma^2)$
    if $\varepsilon \neq \texttt{Failure}$ **then**
      $x' \leftarrow \max\{-k, \min\{k+1, x + \varepsilon\}\}$
      if $f(x') > 0.5$ **then**
        $\texttt{counts}[1] \leftarrow \texttt{counts}[1] + 1$
      else
        $\texttt{counts}[0] \leftarrow \texttt{counts}[0] + 1$
  return $\texttt{counts}$

procedure CERTIFY($f, \sigma, x, n_0, n, \alpha, k$)
  $\texttt{counts0} \leftarrow$ SAMPLEUNDERNOISE(f, x, n_0, σ, k)
  $\hat{c}_A \leftarrow$ top index in $\texttt{counts0}$
  $\texttt{counts} \leftarrow$ SAMPLEUNDERNOISE(f, x, n, σ, k)
  $p_A \leftarrow$ LOWERCONFBOUND$(\texttt{counts}[\hat{c}_A], n, 1 - \alpha)$
  if $p > \frac{1}{2}$ **then**
    return prediction $\hat{c}_A$ and radius $\sigma \Phi^{-1}(p)$
  else
    return ABSTAIN

7.3 Improving ℓ_1 Certified Robustness via Randomized Smoothing By Leveraging Box Constraints

Improving ℓ_1-Certified Robustness via Randomized Smoothing by Leveraging Box Constraints

Václav Voráček [1] **Matthias Hein** [1]

Abstract

Randomized smoothing is a popular method to certify robustness of image classifiers to adversarial input perturbations. It is the only certification technique which scales directly to datasets of higher dimension such as ImageNet. However, current techniques are not able to utilize the fact that any adversarial example has to lie in the image space, that is $[0, 1]^d$; otherwise, one can trivially detect it. To address this suboptimality, we derive new certification formulae which lead to significant improvements in the certified ℓ_1-robustness without the need of adapting the classifiers or change of smoothing distributions. Code is released at https://github.com/vvoracek/L1-smoothing.

1. Introduction

While neural networks have demonstrated excellent performance in a variety of tasks, they are susceptible to small (adversarial) changes of the input (Szegedy et al., 2014; Biggio et al.). Such behaviour is undesired, especially in the safety-critical applications. To mitigate the issue, initially the focus was on constructing empirically robust classifiers, and then check how the resulting model performs against adversarial attacks. However, such an approach only gives an upper bound on the actual robustness of the classifier and many initially considered promising methods later turned out to be broken (Athalye et al., 2018; Carlini et al., 2019; Tramer et al., 2020) due to stronger attacks. The only seemingly working method that does not produce any guarantees is adversarial training (Madry et al., 2018); but more powerful attacks show that the empirical robustness of classifiers is lower than originally claimed (Croce & Hein, 2020; Lin et al., 2022).

[1]Tübingen AI Center, University of Tübingen. Correspondence to: Václav Voráček <vaclav.voracek@uni-tuebingen.de>, Matthias Hein <matthias.hein@uni-tuebingen.de>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

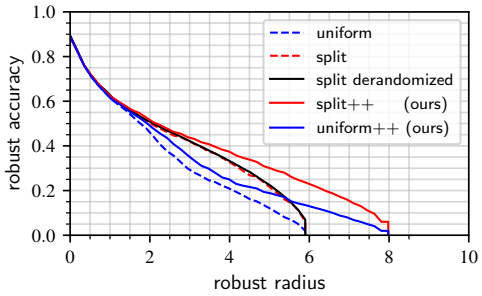


Figure 1: Certified ℓ_1 robust accuracies for CIFAR-10 dataset via randomized smoothing for three different types of noise. The curves uniform++ and split++ use the same networks and noise as uniform and split respectively, however, with the proposed improved certification we are able to significantly increase the certified robustness. The reported curves are pointwise maxima of robustness curves with different noise magnitudes.

Thus, an alternative approach is to certify robustness. Here, we are no longer interested in whether we can find (or fail to find) an adversarial example in the neighbourhood (called *threat model*), but rather focus on whether we can prove (or fail to prove) that there is no adversarial example. These methods roughly fall into three (arguably overlapping) categories:

- Propagate a "nice" set containing the threat model through the network; see, e.g., (Gowal et al., 2018; Wong et al., 2018).
- Force the Lipschitz constant of a model to be small; see, e.g., (Leino et al., 2021; Trockman & Kolter, 2021; Singla et al., 2022; Zhang et al., 2022a).
- Randomized smoothing; see, e.g., (Lecuyer et al., 2019; Cohen et al., 2019; Salman et al., 2019; Yang et al., 2020).

The two approaches discussed yielded either an upper bound (empirical robustness) or a lower bound (certified robustness) respectively on the actual adversarial robustness and in general, there are points for which the certification methods

cannot prove that they are robust, nor the attack is able to find an adversarial example. Although this property is undesirable, it is also inevitable in practice because the problem of finding adversarial examples even in ReLU networks is in NP-Complete (Katz et al., 2017). Therefore, determining the true robustness of a model is only possible for very small networks (Tjeng & Tedrake, 2017) and simple classifiers such as linear models, boosted decision stumps (Kantchelian et al., 2016) , or nearest neighbour (resp. prototype) classifiers (Wang et al., 2019; Saralajew et al., 2020; Voráček & Hein, 2022). Nevertheless, there is a line of work aiming at finding the actual robustness, or at least tightening the gap between the certifiable lower and upper bounds; see (Zhang et al., 2018; 2022b).

We discuss shortly the choice of the threat model. It is the perturbation set with respect to which we want to be robust. The common choices are the ℓ_p balls centered at the input points. While the choice of a threat model is always somewhat arbitrary, if it is not directly motivated by an application, the attacks (and defenses) to many interesting threat models strongly rely on techniques developed for the ℓ_p threat models for both empirical and certified robustness; see (Laidlaw et al., 2021; Voráček & Hein, 2022) for a perceptual metric threat model; (Wong et al., 2019; Levine & Feizi, 2020) for the Wasserstein distance threat model; (Brown et al., 2017; Metzen & Yatsura, 2021; Salman et al., 2022) for a patch threat model, in which the attacker is allowed to arbitrarily set the pixel values in a small patch. The choice of using different ℓ_p norms as threat models leads to qualitatively different adversarial perturbations. For example, when applying the ℓ_∞-threat model with a sufficiently small radius, the changes are typically imperceptible but affect every pixel. On the other hand, the ℓ_1-threat model allows for potentially significant changes in individual pixels, although limited to only a few of them. Instead of a given threat model, that is fixing the perturbation budget, one can also ask for the largest radius of a ball in a given norm in which the classifier does not change - the so called *robust radius*.

Definition 1.1. A classifier $f : \mathbb{R}^d \to \{0, 1\}$ is said to be *robust* at x with respect to a norm $\|\cdot\|$ with robust radius r if $\|x - y\| \leq r \implies f(x) = f(y)$ for every $y \in [0, 1]^d$.

The certified radii and perturbation magnitudes typically considered in empirical robustness for the ℓ_∞-norm are very similar. However, for the ℓ_1-norm, there is a significant difference in the radii considered between certified robustness and empirical defenses. For example, on ImageNet, radii around 4 are considered for certified robustness, while empirical defenses consider radii ranging from 60 to 255 (Croce & Hein, 2022; 2021). This suggests that there exists a gap between what can be certifiably attained and what appears to be empirically achievable for the ℓ_1-norm.

On the other hand it has been argued by Croce & Hein (2021) that ℓ_1-attacks are much more difficult and prone to fail compared to ℓ_∞-attacks and thus it could also be an overestimation problem. They conclude that the intersection of the image domain $[0, 1]^d$ and the ℓ_1-ball as the effective threat model has a quite different geometry than an ℓ_1-ball and construct their attack accordingly. Thus, our motivation is to consider the box-constraints even in the context of certification.

Contributions: In previous work of Levine & Feizi (2021) and Yang et al. (2020) on certified ℓ_1-robustness using randomized smoothing, it has been assumed that the input domain is $\mathbb{R}^d$, even though the techniques are mainly applied to image classifiers, where the domain is $[0, 1]^d$. We show in this paper that taking into account the box constraints of the image domain $[0, 1]^d$ can be used to certify significantly larger ℓ_1-balls than previous work. Our main result is based on the fact that the volume of the overlap of two ℓ_∞-balls when the centers of the balls are restricted to $[0, 1]^d$ behaves quite different from the unconstrained case which leads to an improved control of the smoothed classifier yielding the better guarantees. Our technique can be applied when the smoothing distribution is uniform Yang et al. (2020) as well as for the Splitting Noise of (Levine & Feizi, 2021). We also discuss an improved control of the failure probability as well the better scaling of the ℓ_1-certificates in the failure probability compared to ℓ_2 and ℓ_∞. Finally, we show in the experiments that our improved technique allows to certify much larger ℓ_1-radii than previous work.

1.1. Notation

Real interval between a, b is denoted $[a, b]$. We use Iverson brackets $[\![\text{statement}]\!]$ which is the indicator function of the set for which the statement is true. The floor function, $\lfloor x \rfloor$ stands for the maximal integer no larger than x:

$$\lfloor x \rfloor = \max\{m \in \mathbb{Z} \mid m \leq x\}.$$

The ℓ_p-ball with radius λ centered at $x \in \mathbb{R}^d$ is denoted as

$$\mathcal{B}_p(x, \lambda) = \{z \in \mathbb{R}^d \mid \|z - x\|_p \leq \lambda\}.$$

The uniform distribution on $\mathcal{B}_\infty(0, \lambda)$ in d dimensions is denoted $\mathcal{U}^d(\lambda)$. Volume of a set A is denoted as $Vol(A)$. A one hot vector with 1 at position i is denoted e_i and its dimension will be clear from the context. When the meaning is clear from context, we use f as a base classifier, q as a smoothing distribution, h as a smoothed classifier of f and H as the thresholded version of h as in Equation (1) and (2). Number of samples is denoted by n and the dimension is d.

2. Randomized smoothing

For the simplicity of exposure, we introduce randomized smoothing for the case of binary classification and discuss

the multiclass setting in Section 2.4. We start by treating the mathematical foundations of randomized smoothing and postpone the discussion on the algorithmic implementation to Section 2.5. We will also certify class 1 and the certification of class 0 is symmetric.

2.1. Mathematics of Randomized Smoothing

Randomized smoothing (Lecuyer et al., 2019) is a method that takes an arbitrary binary classifier $f : \mathbb{R}^d \to [0, 1]$ and a noise distribution q supported on $\mathbb{R}^d$; it produces a smoothed version h of the original classifier f:

$$h(x) = \mathop{\mathbb{E}}_{\varepsilon\sim q} f(x + \varepsilon). \tag{1}$$

For the thresholded classifier H defined as

$$H(x) = [\![h(x) > 0.5]\!] \tag{2}$$

we can certify adversarial robustness.

The intuition behind this is as follows: If the distribution q exhibits certain desirable characteristics (such as having a small total variation distance with its slightly shifted copy), for example, if it represents a uniform distribution within a hypercube with a radius larger than $\|\delta\|_1$ for some $\delta \in \mathbb{R}^d$, then $h(x)$ and $h(x + \delta)$ are both expectations of the same function under almost the same distribution; thus they should be similar. Therefore, if $h(x) \approx 1$, then also $h(x + \delta) > 0.5$ We formalize this intuition later in Proposition 2.1 for the case of ℓ_1 distance.

In (Yang et al., 2020), it has been argued and supported by both theoretical and experimental evidence that the optimal smoothing distribution for ℓ_1-robustness should have cubic superlevel sets. That is, a smoothing distribution with density q should satisfy that the set $U_t = \{x \in \mathbb{R}^d \mid q(x) \geq t\}$ is a hypercube for every t. In that case, we can express $q(x)$ as an (uncountable) mixture of uniform distributions supported in ℓ_∞-balls of specific radii and our proposed method can still be applied.

Proposition 2.1. *Let $f : \mathbb{R}^d \to [0, 1]$ and*

$$h(x) = \mathop{\mathbb{E}}_{\varepsilon\sim\mathcal{U}^d(\lambda)} f(x + \varepsilon).$$

Let B_1 and B_2 be the ℓ_∞-balls with radius λ centered at x, y respectively, then

$$h(y) \geq h(x) - 1 + \frac{Vol(B_1 \cap B_2)}{Vol(B_2)}.$$

Proof.

$$\begin{aligned} h(y) &= \frac{\int_{t\in B_2} f(t)dt}{Vol(B_2)} \geq \frac{\int_{t\in B_1\cap B_2} f(t)dt}{Vol(B_2)} \\ &= \frac{\int_{t\in B_1} f(t)dt - \int_{t\in B_1\setminus B_2} f(t)dt}{Vol(B_2)} \\ &\geq \frac{\int_{t\in B_1} f(t)dt - Vol(B_1 \setminus B_2)}{Vol(B_2)} \\ &= h(x) - 1 + \frac{Vol(B_1 \cap B_2)}{Vol(B_2)}, \end{aligned}$$

using $Vol(B_1 \setminus B_2) = Vol(B_2) - Vol(B_1 \cap B_2)$ and $Vol(B_1) = Vol(B_2)$. □

It remains to find a lower bound on the volume of intersection of two ℓ_∞-balls. For now, we present a simple bound and will return to a proper treatment later in Proposition 2.5 and Theorem 2.8.

Proposition 2.2. *Let B_1, B_2 be ℓ_∞-balls with radii λ centered at $x, y \in \mathbb{R}^d$ respectively; then*

$$\frac{Vol(B_1 \cap B_2)}{Vol(B_1)} \geq 1 - \frac{\|x - y\|_1}{2\lambda}.$$

Proof. The proof can be found in Appendix A.1. □

Now we have developed the intuition and tools, we are ready to state the foundational theorem of ℓ_1 robustness.

Theorem 2.3. *(Lee et al., 2019) Let $f : \mathbb{R}^d \to [0, 1]$ be a deterministic or random classifier. Then the smoothed classifier defined as:*

$$h(x) = \mathop{\mathbb{E}}_{\varepsilon\sim\mathcal{U}^d(\lambda)} [f(x + \varepsilon)]$$

is $1/(2\lambda)$-Lipschitz with respect to ℓ_1-norm.

Proof. Plugging the bound from Proposition 2.2 into Proposition 2.1 yields $h(y) - h(x) \leq \|x - y\|_1 / (2\lambda)$ for all $x, y \in \mathbb{R}^d$, thus it holds that $|h(y) - h(x)| \leq 1/(2\lambda) \|x - y\|_1$

□

Theorem 2.3 allows us to directly compute the radius λ of the ℓ_1 ball $\mathcal{B}_1(x, \lambda)$ such that it is classified by classifier H with the same label as $H(x)$.

Corollary 2.4 (of Theorem 2.3)**.** *Let h be a smoothed classifier as in Theorem 2.3. Then H (thresholding h at 0.5) is robust (for class 1) at x with certified radius*

$$r(x) = 2\lambda(h(x) - 1/2).$$

Proof. Using $h(y) \geq h(x) - \frac{\|x-y\|_1}{2\lambda} > \frac{1}{2}$ from Theorem 2.3 and solving for $\|x - y\|_1$ yields the result. □

2.2. Box Constraints

In the case of ℓ_1-robustness, the most successful methods use the uniform distribution in an ℓ_∞-ball as smoothing distribution. Thus, we focus on this case first and discuss the others later. In Proposition 2.1 we have established that the overlap of supports of the smoothing distributions is a crucial factor for the robustness certification. In the upcoming proposition, we show that considering box-constraints gives us a tighter upper bound on the minimal possible overlap.

Proposition 2.5. *Let B_1, B_2 be the ℓ_∞ balls with radii λ centered at $x, y \in [0,1]^d$; then*

$$\frac{Vol(B_1 \cap B_2)}{Vol(B_1)} \geq \left(1 - \frac{1}{2\lambda}\right)^{\lfloor \|x-y\|_1 \rfloor} \left(1 - \frac{\|x-y\|_1 - \lfloor \|x-y\|_1 \rfloor}{2\lambda}\right) \geq \left(1 - \frac{1}{2\lambda}\right)^{\|x-y\|_1}.$$

The very last inequality holds when $2\lambda \geq 1$. Both of the inequalities are attainable.

Proof. The proof can be found in Appendix A.2. □

The aim was to provide simple expressions in Proposition 2.5 so that we can express the certified radius in a closed form. Specifically, we can plug the result from Proposition 2.5 into Proposition 2.1, resulting in Theorem 2.6 and Corollary 2.7. In Figure 2, we can see the improvement achieved compared to the certificates based on Propositions 2.2 and 2.1 resulting in Corollary 2.4.

Theorem 2.6. *Let $f : \mathbb{R}^d \to [0,1]$ be a deterministic or random classifier. Then the smoothed classifier is defined as:*

$$h(x) = \mathop{\mathbb{E}}_{\varepsilon \sim \mathcal{U}^d(\lambda)} \left[f(x+\varepsilon) \right].$$

Then for $x, y \in [0,1]^d$ it holds that

$$|h(x) - h(y)| \leq 1 - \left(1 - \frac{1}{2\lambda}\right)^{\|x-y\|_1}.$$

Proof. Plugging the bound from Proposition 2.5 into Proposition 2.1. □

Corollary 2.7 (of Theorem 2.6)**.** *Let h be a smoothed classifier as in Theorem 2.6. Then H (thresholding h at 0.5) is robust (for class 1) at x with certified radius*

$$r(x) = \frac{\ln(1.5 - h(x))}{\ln(1 - \frac{1}{2\lambda})}.$$

Proof. Using $h(x) - \frac{1}{2} \leq 1 - \left(1 - \frac{1}{2\lambda}\right)^{\|x-y\|_1}$ from Theorem 2.6 and solving for $\|x-y\|_1$ yields the result. □

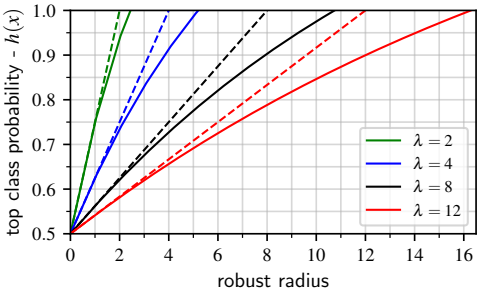


Figure 2: Conversion of top class probability of a smoothed classifier to the ℓ_1 certifiable robust radius for different noise magnitudes λ used for smoothing. The dashed lines are via Corollary 2.4 and the solid ones are via our Corollary 2.7.

We can further utilize the fact that the maximal possible difference between a potential adversarial example and the original image x at position i is at most $d_i = \max\{x_i, 1 - x_i\}$. In Proposition 2.5, we used an upper bound of $d_i \leq 1$. However, the following theorem establishes that for an image $x \in [0,1]^d$, we can find an image $y \in [0,1]^d$ that minimizes the intersection of $\mathcal{B}_\infty(x, \lambda)$ and $\mathcal{B}_\infty(y, \lambda)$ under the constraint $\|x-y\|_1 \leq c$ through a greedy coordinate-wise minimization approach. The improvements in certification are demonstrated in Example 2.9 and Figure 3.

Theorem 2.8. *Let $x \in [0,1]^d$. Let σ_i be an ordering induced by how far is x_i from boundary. That is:*

$$i \leq j \implies \min(x_{\sigma_i}, 1 - x_{\sigma_i}) \leq \min(x_{\sigma_j}, 1 - x_{\sigma_j}).$$

Then for any $c > 0$ such that there exists $y \in [0,1]^d$ with $\|x-y\|_1 = c$ it holds that

$$\inf_{y \in [0,1]^d \cap \mathcal{B}_1(x,c)} \frac{Vol(\mathcal{B}_\infty(x,\lambda) \cap \mathcal{B}_\infty(y,\lambda))}{Vol(\mathcal{B}_\infty(x,\lambda))} = \left(\prod_{i=1}^{T} \left(1 - \frac{\max\{x_{\sigma_i}, 1 - x_{\sigma_i}\}}{2\lambda}\right)\right)\left(1 - \frac{U}{2\lambda}\right)$$

where

$$T = \max_{k \in \mathbb{N}} \quad s.t. \sum_{i=1}^{i=k} \max(x_{\sigma_i}, 1 - x_{\sigma_i}) \leq c,$$

and

$$U = c - \sum_{i=1}^{i=T} \max(x_{\sigma_i}, 1 - x_{\sigma_i}).$$

Proof. The proof can be found in Appendix A.3 □

Theorem 2.8 is a clear generalization of Proposition 2.5 when we choose $x = \mathbf{0}$ in Theorem 2.8. Similarly, Proposition 2.2 can be seen as another corollary with a minor effort.

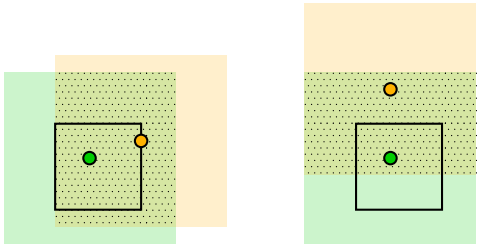


Figure 3: Effect of box constraints on the minimal overlap of two ℓ_∞ balls. The green point is at $(0.4, 0.6)$ while the orange one is at a distance 0.8 in ℓ_1-norm. On the left is depicted the minimal possible overlap considering box constraints (cf. Theorem 2.8), while on the right is the minimal possible overlap without considering box constraints (cf. Proposition 2.2). See Example 2.9 for further discussion.

Therefore, the proofs involve subtle variations of the same underlying idea.

Theorem 2.8 can be used for certification with Proposition 2.1. The certified radius cannot be expressed in a closed form but can be efficiently computed after sorting the coordinates since the volume of the intersection is an increasing piecewise-linear function of the robust radius.

Example 2.9. Consider a point $x = (0.4, 0.6)$ and a smoothing distribution uniform in $\mathcal{B}_\infty(0, 1)$, that is $\lambda = 1$. We want to certify the thresholded version of h when $h(x) = 0.88$. If we don't consider the box constraints, we can only certify robust radius via Corollary 2.4: $2\lambda(0.88 - 0.5) < 0.8$. However, if we consider the box constraints, we can certify robust radius 0.8 via Proposition 2.1 and Theorem 2.8. This is a consequence of the fact that the lower bound on the volume of intersections of two ℓ_∞ balls in Proposition 2.2 (in this case 0.6) is weaker than the (exact one) in Theorem 2.8 (in this case 0.63), see Figure 3 for the illustration.

2.3. Smoothing with Splitting Noise

An alternative ℓ_1-certification method proposed by Levine & Feizi (2021) uses a splitting noise. We show that if the splitting noise is independent in every dimension, then our certification from Corollary 2.7 can be directly applied here. For the simplicity, we introduce only the splitting noise with $\lambda \geq 0.5$ since the general version is more complicated and $0 < \lambda < 0.5$ can only be used to certify small radii; thus we do not improve the certification for that case. The fundamental concept behind the splitting noise is that, at every coordinate, we have two options: either we add uniform noise from the interval $[0, 1]$, or we set the value at that coordinate to 1. The strength of the noise λ determines the frequency at which each of these procedures occurs.

Theorem 2.10 (Theorem 2 of Levine & Feizi (2021))**.** *For any $f : \mathbb{R}^d \to [0, 1]$, and $\lambda \geq 0.5$ let $\mathbf{s} \in [0, 2\lambda]^d$ be a random variable whose (not necessarily independent) marginals follow the uniform distribution on $[0, 2\lambda]$. Then define*

$$\tilde{x}_i(s_i) = \min(s_i, 1) + [\![x_i > s_i]\!], \quad \forall i$$
$$h(x) = \mathop{\mathbb{E}}_{s}\left[f(\tilde{x}(s))\right].$$

Then, $h(\cdot)$ is $(1/2\lambda)$-Lipschitz with respect to the ℓ_1-norm.

Let us take a closer look at the distribution $\tilde{x}(s)$ for some $x \in [0, 1]$ and s uniformly drawn from interval $[0, 2\lambda]$. We split the inspection in three cases:

1. With probability $x/(2\lambda)$: $s \leq x$, then $s \leq 1$ and $\tilde{x}(s) = 1 + s$.
2. With probability $(1 - x)/(2\lambda)$: $x \leq s \leq 1$, then $\tilde{x}(s) = s$.
3. With probability $1 - 1/(2\lambda)$: $1 \leq s$, then $\tilde{x}(s) = 1$.

Thus, $\tilde{x}(s)$ is a mixture of a uniform distribution on $[x, 1+x]$ and a constant random variable at 1 with respective mixture coefficients $1/(2\lambda)$ and $1-1/(2\lambda)$ respectively. Thus, when $\lambda = 0.5$, the splitting noise distribution and uniform distribution in $\mathcal{B}_\infty(x, \lambda)$ are equal. Given this observation, it comes at no surprise that the techniques we used to improve the certification in the case of uniform noise in $\mathcal{B}_\infty(x, \lambda)$, resulting in Corollary 2.7, can be used also in the case of splitting noise.

Theorem 2.11. *Let the assumptions be as in Theorem 2.10 and additionally let the marginals of s be independent. Then Proposition 2.1 holds when the uniform noise is replaced by the splitting noise.*

Proof. Take $x, y \in \mathbb{R}^d$ and a noise sample $s \in \mathcal{U}^d(\lambda)$. At every position we have

$$\tilde{x}_i(s_i) = \min(s_i, 1) + [\![x_i > s_i]\!],$$

thus, in order to $\tilde{x}_i(s_i) \neq \tilde{y}_i(s_i)$, it has to be the case that

$$[\![x_i > s_i]\!] \neq [\![y_i > s_i]\!]$$

which happens with probability $\frac{|x_i - y_i|}{2\lambda}$. Let $R \subset [0, 2\lambda]^d$ such that for every $s \in R$ we have $\tilde{x}(s) = \tilde{y}(s)$. The probability that $\tilde{x}(s)$ and $\tilde{y}(s)$ are equal is exactly

$$\frac{Vol(R)}{(2\lambda)^d} = \prod_{i=1}^{d}\left(1 - \frac{|x_i - y_i|}{2\lambda}\right) = \frac{Vol(\mathcal{B}_\infty(x, \lambda) \cap \mathcal{B}_\infty(y, \lambda))}{Vol(\mathcal{B}_\infty(x, \lambda))}$$

since s has independent marginals. Then we mimick the proof of Proposition 2.1:

$$\begin{aligned} h(y) &= \frac{\int_{s\in[0,2\lambda]^d} f(\tilde{y}(s))ds}{(2\lambda)^d} \geq \frac{\int_{s\in R} f(\tilde{y}(s))ds}{(2\lambda)^d} \\ &= \frac{\int_{s\in R} f(\tilde{x}(s))ds}{(2\lambda)^d} \\ &= \frac{\int_{s\in[0,2\lambda]^d} f(\tilde{x}(s))ds - \int_{s\in[0,2\lambda]^d\setminus R} f(\tilde{x}(s))ds}{(2\lambda)^d} \\ &\geq h(x) - \frac{(2\lambda)^d - Vol(R)}{(2\lambda)^d} \\ &= h(x) - 1 + \frac{Vol(\mathcal{B}_\infty(x,\lambda)\cap\mathcal{B}_\infty(y,\lambda))}{Vol(\mathcal{B}_\infty(x,\lambda))}. \end{aligned}$$

□

Theorem 2.11 shows that smoothing with both uniform and splitting noise are captured by Proposition 2.1. Thus, the certification methods from Corollaries 2.4 and 2.7 developed for uniform noise can be also used for the splitting noise and for the clarity, we will keep using uniform noise in the discussions.

2.3.1. Deterministic Splitting Noise

The splitting noise has another useful property. If the noise in different coordinates is not independent, then we can evaluate the expectation exactly using $2q\lambda$ evaluations of the base classifier, where q here stands for the number of quantization levels which is commonly 256. We refer the reader to Levine & Feizi (2021) for the details. This has two benefits; one, the provided certificates are deterministic and second, they are faster to compute - although this is not inherent. We can use less samples to estimate the expectation of the base classifier under the smoothing noise. See Subsection 2.7 and Figure 4 for more details. Our method cannot be applied in this deterministic case because we can no longer rely on the independence of the splitting noise across different coordinates.

2.4. Multiclass Classification

We introduced the randomized smoothing machinery for the task of binary classification. In the K-class setting we define randomized smoothing as follows; let $f : \mathbb{R}^d \to \{e_1, e_2, \dots e_K\}$ where e_i are one-hot vectors with 1 at position i be the base classifier. The smoothed classifier is then

$$h(x) = \mathop{\mathbb{E}}_{\varepsilon\sim q} f(x+\varepsilon)$$

and we certify robustness for its thresholded version; i.e., for

$$H(x) = \arg\max_{i=1}^{K} h(x)_i.$$

A possible approach to the multiclass setting is straightforward; just consider all the other classes as a one big class. That is, treat the multiclass classifier $f(x)$ as if it would be a binary classifier $f(x)_y$ when certifying class y. This approach is commonly taken in the randomized smoothing literature (Salman et al., 2019; Cohen et al., 2019) to avoid problems with estimation of class probabilities. Thus, we can directly use all the theory we have developed so far for the binary classification. However, this simplification may come at a high cost. Consider for example a classification task with $k = 1000$ classes with $h(x)_1 = 0.4$ and $h(x)_i < 0.001$ for the other classes. Then with the discussed conversion we are not able to certify class 1. However, as we will see, H can be moderately robust at x.

Proposition 2.12. *Let $f : \mathbb{R}^d \to \{e_1, e_2, \dots e_K\}$ be a base classifier, its smoothed version be h and H be the thresholded version of h and the smoothing distribution be $\mathcal{U}^d(\lambda)$. Let also $H(x) = A$. Then for any point y it holds that*

$$\frac{h(y)_A - h(y)_B}{2} \geq \frac{h(x)_A - h(x)_B}{2} - 1 + \frac{Vol(B_1\cap B_2)}{Vol(B_2)}$$

where B is an arbitrary class and $B_1 = \mathcal{B}_\infty(x,\lambda)$ and $B_2 = \mathcal{B}_\infty(y,\lambda)$,

Proof. We subtract the inequalities from Proposition 2.1 applied to $h(\cdot)_A$ and $h(\cdot)_B$. □

In order to certify the thresholded classifier in the multiclass setting, we have to ensure that $\frac{h(y)_A - h(y)_B}{2} \geq 0$ in the notation of Proposition 2.12.

Corollary 2.13 (of Proposition 2.12 and 2.2)**.** *Let the notation be as in Proposition 2.12. Then H is certifiably robust at x with robust radius*

$$r(x) = \lambda\left(h(x)_A - h(x)_B\right),$$

where $H(x) = A$ and B is the runner-up class.

Corollary 2.14 (of Proposition 2.12 and 2.5)**.** *Let the notation be as in Proposition 2.12. Then H is certifiably robust at x with robust radius*

$$r(x) = \frac{\ln\left(1 - \frac{h(x)_A - h(x)_B}{2}\right)}{\ln\left(1 - \frac{1}{2\lambda}\right)}$$

where $H(x) = A$ and B is the runner-up class.

2.5. Algorithmic Implementation

We have covered the mathematical foundations of randomized smoothing. However, the exact expectation in Equation (1) is usually intractable to evaluate; therefore, Monte Carlo

sampling is used to estimate it (whence *randomized* smoothing). As a consequence, the technique is stochastic and we cannot guarantee that it will always produce a valid radius. Still, we can control the probability of bad luck during sampling and the certificates are (by design) computed so that they are correct with $1 - \alpha = 99.9\%$ probability. We discuss the procedure in Section 2.6. We emphasize that the certificates are for the actual classifier induced by h defined using an expectation. For the same reason, we should be careful with querying the classifier h - it brings another source of randomness. To control all this randomness, one commonly uses $n = 100\,000$ samples to estimate $h(x)$ in Equation (1) which leads to long certification and inference times. However, it is not necessary as we will discuss in Section 2.7

The choice of a smoothing distribution q is crucial for the performance. Some popular choices are normal distribution (for ℓ_2- and ℓ_∞-threat models) and uniform distributions in ℓ_p balls; see (Yang et al., 2020) for a thorough inspection of many smoothing distributions and respective certifications or (Dvijotham et al., 2020) for a general certification framework that is applicable for virtually any smoothing distribution. Specially, for smoothing with normal distribution with covariance matrix $\sigma^2 I_d$, it was shown by Cohen et al. (2019) that h is robust at a with radius $\sigma\Phi^{-1}(h(a))$ under the ℓ_2-threat model, where Φ^{-1} is the quantile function of standard normal distribution. For the ℓ_∞-threat model the radius $\sigma\Phi^{-1}(h(a))/\sqrt{d}$ can be certified as it fits into a ℓ_2-ball of radius $\sigma\Phi^{-1}(h(a))$.

2.6. Controlling Failure Probability

In order to provide high probability certificates, we need to estimate some of the class probabilities. A standard way to perform certification is by first evaluating a few (usually $n_0 = 64$, however, we use 256 in the experiments) noisy samples to estimate the top-1 class of $h(x)$ that is certified in a second step via one-versus-all approach as discussed in Section 2.4. This approach has the benefit that one only needs to estimate the parameter of a binomial distribution and one can easily control the failure probability via Clopper-Pearson tail bounds. However, this approach has its downsides pointed out in Subsection 2.4. Thus, we will follow Proposition 2.12 for the certification. We need to estimate not just top-1 class probability, but also the top-2 class probability. We use the standard Bonferroni correction to estimate them. That is, we estimate both, top-1 and top-2 class probability with allowed failure probability $\alpha/2$ via Clopper-Pearson tail bounds. Therefore, by a union bound, both of the estimated values are simultaneously correct with probability at least $1 - \alpha$.

See Algorithm 1 for the actual certification via Corollary 2.14. We note that there is no guarantee that $\hat{A}$ nor $\hat{B}$ correspond to the actual two most probable classes. However, it does not matter. The value p_A does not exceed $h(x)_A$ with probability at least $1 - \alpha/2$; thus, it also cannot exceed $\max_k h(x)_k$. Similarly, even if the actual top-2 class is not $\hat{B}$, then p_B is overestimating $h(x)_B$ with probability at least $1 - \alpha/2$. Thus, the failure probability is at most α. The function LOWCONFBOUND (resp. UPPCONFBOUND) computes lower (resp. upper) confidence interval for $h(x)_A$ (resp. $h(x)_B$). We discuss its implementation in Subsection 3.2.

Algorithm 1 Randomized Smoothing Certification

procedure SAMPLEUNDERNOISE(f, x, n, λ)
  `counts` $\leftarrow [0, 0, \ldots, 0]$
  for $i \leftarrow 1, n$ **do**
    $x' \leftarrow$ `noise`(x, λ)
    `counts` $\leftarrow$ `counts` $+ f(x')$
  return `counts`

procedure CERTIFY($f, x, n_0, n, \lambda, \alpha$)
  `counts0` $\leftarrow$ SAMPLEUNDERNOISE(f, x, n_0, λ)
  $\hat{A} \leftarrow$ top index in `counts0`
  `counts` $\leftarrow$ SAMPLEUNDERNOISE(f, x, n, λ)
  $\hat{B} \leftarrow$ top index in `counts` but not $\hat{A}$
  $p_A \leftarrow$ LOWCONFBOUND(`counts`$[\hat{A}], n, \alpha/2$)
  $p_B \leftarrow$ UPPCONFBOUND(`counts`$[\hat{B}], n, \alpha/2$)
  if $p_A > p_B$ **then**
    return prediction $\hat{A}$ and radius

$$\frac{\ln\left(1 - \frac{p_A - p_B}{2}\right)}{\ln\left(1 - \frac{1}{2\lambda}\right)}$$

  else
    return ABSTAIN

2.7. Influence of the Number of Samples

We discuss the influence of the number of samples used to estimate the output of a smoothed classifier and the required confidence on the certified accuracy. In the ℓ_1-case that we have covered, certification using Corollaries 2.4, 2.7 and their multiclass counterparts scales roughly linearly in the estimated probability

$$\ln\left(\frac{3}{2} - h(x) - \epsilon\right) \geq \ln\left(\frac{3}{2} - h(x)\right) - 2\epsilon$$

for $\epsilon > 0$ and $h(x) + \epsilon \leq 1$. We used that $\sup_{x \in [\frac{1}{2}, \frac{3}{2}]} \ln'(x) = 2$. The width of confidence intervals (for $p \geq 0.5$) when $p \approx 1$ scales roughly as $\frac{-\ln(\alpha)}{n}$ for confidence level α, while when $p \not\approx 1$, the width scales as $\sqrt{\frac{-\ln(\alpha)}{n}}$ as follows from the Bennett's inequality. A (sub

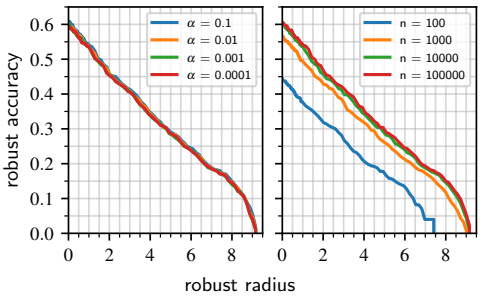


Figure 4: Effect of the choice of α (on the left; $n = 10\,000$) and of n (on the right; $\alpha = 0.001$) on the ℓ_1-robustness curve. Experiment is on CIFAR10 with $\lambda = 6.92$ and certification according to Corollary 2.7.

optimal) universal bound by Hoeffding's inequality yields

$$t = \sqrt{\frac{\ln(\alpha)}{-2n}}$$

as a width of the (one-sided) confidence interval; see Appendix B for the derivation. Thus, the estimation error of the robust radius gecreases at least as $\sqrt{\frac{-\ln \alpha}{n}}$ and significantly faster when the estimated probability is close to 1.

In practice, it might not be necessary to push for the largest n possible. See Figure 4 for certification with different choices of numbers of noise samples and choices of α to get an impression of how much the robustness curves are influenced by these hyperparameters.

We note that this finding does not transfer to ℓ_2- and ℓ_∞-smoothing where the normal distribution is dominantly used as a smoothing distribution. This is because for these cases the certificates are not linear in the estimated probability. They are computed as $\sigma\Phi^{-1}(p)$, where Φ^{-1} grows arbitrarily steeply as p approaches 1. For example, if we have a constant base classifier, $\sigma = 1$, and $n \in \{10\,000, 100\,000, 1\,000\,000\}$, then we can certify radii 3.20, 3.81 and 4.35 respectively which makes a huge difference. See Figure 8 of (Cohen et al., 2019) for more details. To conclude this subsection, we demonstrated that ℓ_1 certification via randomized smoothing requires significantly less samples than in the ℓ_2 and ℓ_∞ cases in order to reasonably controll the estimation error of a robust radius.

3. Experiments

We performed an extensive evaluation on CIFAR-10 (Krizhevsky et al., 2009) and ImageNet-1k (Russakovsky et al., 2015) and demonstrate the improvements comparing to (Levine & Feizi, 2021) and (Yang et al., 2020). To ensure a fairness of the evaluation, we follow the experimental setup that is identical in both mentioned papers but we perform it over a wider range of smoothing distribution parameters. Following previous work, we report standard deviations of the distribution instead of the λ parameter that we have used throughout the paper. The conversion is that σ corresponds to $\lambda := \sqrt{3}\sigma$. Namely, for CIFAR-10, we evaluated on $\sigma \in \{0.15, 0.25, \ldots 3.5, \ldots 8, 9, 10, 12\}$, where in the first gap, the spacing is 0.25 and in the second it is 0.5 and for ImageNet we used $\sigma \in \{0.5, 1.25, 2, 2.75, 3.5, 4.5, 5.5\}$.

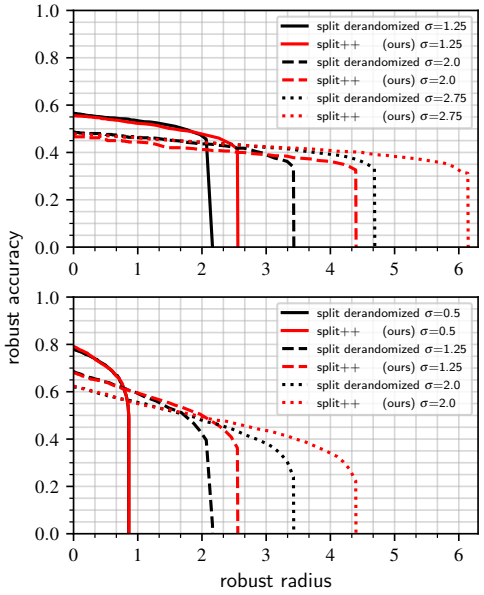


Figure 5: Comparison of certification using deterministic splitting noise and with independent splitting noise and our improved certification. The models were trained with stability training. ImageNet is shown in the top figure, CIFAR-10 in the bottom figure.

3.1. Training

For ImageNet we used a ResNet-50 trained for 30 epochs; and for CIFAR-10 we used a WideResNet-40-2 trained for 120 epochs. The optimizer is SGD with learning rate 0.1, Nesterov momentum 0.9 and weight decay 0.0001 with cosine annealing learning rate schedule and batch size is 64 for both models. We experimented with the two following types of training (with a slight abuse of notation in the case of splitting noise):

1. standard training with cross entropy loss

$$\mathcal{L}(x, y) = -\log\big(f(x+\delta)_y\big), \quad \delta \sim q$$

on noise-augmented data points as suggested in (Cohen et al., 2019).

2. stability training (Li et al., 2019) (is roughly twice as expensive) with loss

$$\begin{aligned}\mathcal{L}(x, y) =& C \cdot \mathrm{KL}\big(f(x+\delta_1) \parallel f(x+\delta_2)\big) \\ & - \log\big(f(x+\delta_1)_y\big), \quad \delta_1, \delta_2 \sim q\end{aligned}$$

where C is a hyperparameter chosen as $C = 6$ following (Carmon et al., 2019).

In the case when the smoothing distribution q is a uniform distribution, upgrading standard training to stability training helps significantly as observed by Levine & Feizi (2021) and Yang et al. (2020). However, for the splitting noise the benefits are less apparent and sometimes it even hurts. Nevertheless, according to our experiments, for every radius at which we evaluate robustness, the best performing model was trained with stability training.

3.2. Certification Results

Following the literature, we set the probability of certificate being incorrect to be at most $\alpha = 0.001$ for all methods. The only exception is smoothing with deterministic splitting noise which is always correct. Unless stated otherwise, we use $10\,000$ noise samples for the certification and 256 to estimate the top-1 class. The confidence intervals are computed using Python function `proportion_confint` from package `statsmodels.stats.proportion` implementing the Clopper-Pearson method. We call the method with $\alpha = 0.002$ because the confidence interval returned is central and the coverage is $1 - \alpha/2$ in both tails. For CIFAR-10 dataset we certify $2\,000$ images from the test set, while for ImageNet we certify the same subset of 500 images as Cohen et al. (2019) and Levine & Feizi (2021).

In Figure 5 we empirically demonstrate that with the improved certification we are able to certify significantly larger ℓ_1-radii both on ImageNet and CIFAR-10. In Appendix C, there is an additional extensive comparison of the proposed method with the current state of the art.

In Figure 6 we show how the choice of the certification scheme (binary or multiclass) affects the robustness curve. This explains why we observe a similar performance of splitting noise method with its derandomized counterpart, while Levine & Feizi (2021) (who used the binary certification scheme) observed significantly weaker performance.

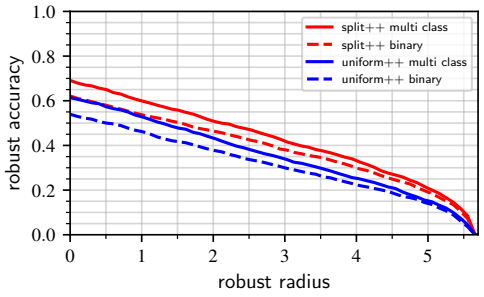


Figure 6: Comparison of the robustness curves for the binary (Corollary 2.7) and multiclass (Corollary 2.14) certification approach. Note that the multiclass approach (almost strictly) outperforms the binary one. Certification was done by our methods, but the conclusion holds for the standard bounds from Corollary 2.4 and 2.13 as well. The setting is standard training, CIFAR-10, $\sigma = 2.5$.

4. Conclusions

In this paper we have shown that incorporating the constraint of image classifiers that input points have to lie in the image domain $[0, 1]^d$ leads to significantly improved certified ℓ_1-radii. The application of our framework is essentially for free and can be directly applied to randomized smoothing using uniform or splitting noise. Our experiments show that we can certify significantly larger ℓ_1-radii than previous work but there still remains a gap to what seems possible in empirical ℓ_1-robustness which is an interesting question for future research for both empirical and certified robustness.

Acknowledgements

The authors thank the anonymous reviewers for their comments, which helped improve the quality of the manuscript. The authors acknowledge support from the DFG Cluster of Excellence "Machine Learning – New Perspectives for Science", EXC 2064/1, project number 390727645 and the Carl Zeiss Foundation in the project "Certification and Foundations of Safe Machine Learning Systems in Healthcare". The authors are thankful for the support of Open Philanthropy.

References

Athalye, A., Carlini, N., and Wagner, D. A. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *ICML*, 2018.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks

against machine learning at test time. In *ECML*.

Brown, T. B., Mané, D., Roy, A., Abadi, M., and Gilmer, J. Adversarial patch. *arXiv preprint arXiv:1712.09665*, 2017.

Carlini, N., Athalye, A., Papernot, N., Brendel, W., Rauber, J., Tsipras, D., Goodfellow, I., Madry, A., and Kurakin, A. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019.

Carmon, Y., Raghunathan, A., Schmidt, L., Duchi, J. C., and Liang, P. S. Unlabeled data improves adversarial robustness. In *NeurIPS*, 2019.

Cohen, J. M., Rosenfeld, E., and Kolter, J. Z. Certified adversarial robustness via randomized smoothing. In *NeurIPS*, 2019.

Croce, F. and Hein, M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

Croce, F. and Hein, M. Mind the box: l_1-apgd for sparse adversarial attacks on image classifiers. In *ICML*, 2021.

Croce, F. and Hein, M. Adversarial robustness against multiple ℓ_p-threat models at the price of one and how to quickly fine-tune robust models to another threat model. In *ICML*, 2022.

Dvijotham, K. D., Hayes, J., Balle, B., Kolter, Z., Qin, C., Gyorgy, A., Xiao, K., Gowal, S., and Kohli, P. A framework for robustness certification of smoothed classifiers using f-divergences. In *ICLR*, 2020.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., and Kohli, P. On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*, 2018.

Kantchelian, A., Tygar, J., and Joseph, A. Evasion and hardening of tree ensemble classifiers. In *ICML*, 2016.

Katz, G., Barrett, C., Dill, D. L., Julian, K., and Kochenderfer, M. J. Reluplex: An efficient smt solver for verifying deep neural networks. In *CAV*, 2017.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Laidlaw, C., Singla, S., and Feizi, S. Perceptual adversarial robustness: Defense against unseen threat models. In *ICLR*, 2021.

Lecuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., and Jana, S. Certified robustness to adversarial examples with differential privacy. In *IEEE Symposium on Security and Privacy (SP)*, 2019.

Lee, G.-H., Yuan, Y., Chang, S., and Jaakkola, T. Tight certificates of adversarial robustness for randomly smoothed classifiers. In *NeurIPS*, 2019.

Leino, K., Wang, Z., and Fredrikson, M. Globally-robust neural networks. In *ICML*, 2021.

Levine, A. and Feizi, S. Wasserstein smoothing: Certified robustness against wasserstein adversarial attacks. In *AISTATS*, 2020.

Levine, A. J. and Feizi, S. Improved, deterministic smoothing for ℓ_1 certified robustness. In *ICML*, 2021.

Li, B., Chen, C., Wang, W., and Carin, L. Certified adversarial robustness with additive noise. In *NeurIPS*, 2019.

Lin, W., Lucas, K., Bauer, L., Reiter, M. K., and Sharif, M. Constrained gradient descent: A powerful and principled evasion attack against neural networks. In *ICML*, 2022.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Metzen, J. H. and Yatsura, M. Efficient certified defenses against patch attacks on image classifiers. In *ICLR*, 2021.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

Salman, H., Li, J., Razenshteyn, I., Zhang, P., Zhang, H., Bubeck, S., and Yang, G. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, 2019.

Salman, H., Jain, S., Wong, E., and Madry, A. Certified patch robustness via smoothed vision transformers. In *CVPR*, 2022.

Saralajew, S., Holdijk, L., and Villmann, T. Fast adversarial robustness certification of nearest prototype classifiers for arbitrary seminorms. In *NeurIPS*, 2020.

Singla, S., Singla, S., and Feizi, S. Improved deterministic l_2 robustness on CIFAR-10 and CIFAR-100. In *ICLR*, 2022.

Steele, J. M. The cauchy-schwarz master class: An introduction to the art of mathematical inequalities, 2004.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *ICLR*, 2014.

Tjeng, V. and Tedrake, R. Verifying neural networks with mixed integer programming. preprint, arXiv:1711.07356v1, 2017.

Tramer, F., Carlini, N., Brendel, W., and Madry, A. On adaptive attacks to adversarial example defenses. In *NeurIPS*, 2020.

Trockman, A. and Kolter, J. Z. Orthogonalizing convolutional layers with the cayley transform. In *ICLR*, 2021.

Voráček, V. and Hein, M. Provably adversarially robust nearest prototype classifiers. In *ICML*, 2022.

Wang, L., Liu, X., Yi, J., Zhou, Z.-H., and Hsieh, C.-J. Evaluating the robustness of nearest neighbor classifiers: A primal-dual perspective. *arXiv preprint, arXiv:1906.03972*, 2019.

Wong, E., Schmidt, F., Metzen, J. H., and Kolter, J. Z. Scaling provable adversarial defenses. In *NeurIPS*, 2018.

Wong, E., Schmidt, F., and Kolter, Z. Wasserstein adversarial examples via projected sinkhorn iterations. In *ICML*, 2019.

Yang, G., Duan, T., Hu, J. E., Salman, H., Razenshteyn, I., and Li, J. Randomized smoothing of all shapes and sizes. In *ICML*, 2020.

Zhang, B., Jiang, D., He, D., and Wang, L. Boosting the certified robustness of l-infinity distance nets. In *ICLR*, 2022a.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. Efficient neural network robustness certification with general activation functions. In *NeurIPS*, 2018.

Zhang, H., Wang, S., Xu, K., Li, L., Li, B., Jana, S., Hsieh, C.-J., and Kolter, J. Z. General cutting planes for bound-propagation-based neural network verification. In *NeurIPS*, 2022b.

A. Proofs of Proposition 2.2,2.5 and Theorem 2.8

The central quantity in the proofs is

$$\frac{Vol(\mathcal{B}_\infty(0,\lambda)\cap\mathcal{B}_\infty(z,\lambda))}{Vol(\mathcal{B}_\infty(0,\lambda))}=\prod_{i=1}^{d}\left(\frac{2\lambda-z_i}{2\lambda}\right)=\prod_{i=1}^{d}\left(1-\frac{z_i}{2\lambda}\right) \tag{3}$$

for $z\in[0,2\lambda]^d$. We show that (3) is a Schur-concave function and is therefore minimized by a maximal element w.r.t. the majorization order. The following definitions and propositions are adopted from Steele (2004).

Definition A.1. Let $x,y\in\mathbb{R}^d$. We write $x\succeq y$ (x *weakly majorizes* y) if for all $1\leq k\leq d$ it holds that

$$\sum_{i=1}^{k}x_i\geq\sum_{i=1}^{k}y_i.$$

If further $\sum_{i=1}^{d}x_i=\sum_{i=1}^{d}y_i$, we write $x\succ y$ (x majorizes y).

Definition A.2. A function $f:\mathcal{X}\to\mathbb{R}$ is said to be Schur-concave if for all $x,y\in\mathcal{X}$ such that $x\succ y$ it holds that $f(x)\leq f(y)$.

Proposition A.3. *Let $f:\mathcal{X}\to\mathbb{R}$ be a differentiable symmetric function. Then it is Schur-concave if*

$$(x_i-x_j)\left(\frac{\partial f}{\partial x_i}-\frac{\partial f}{\partial x_j}\right)\leq 0.$$

Proposition A.4. *Function (3) is Schur-concave. It further holds that for all x,y such that $x\succeq y$, $f(x)\geq f(y)$.*

Proof. Let $X=(0,2\lambda)^d$. Function f is positive; thus is Schur-concave on X if and only if $g(x)=\log(f(x))$ is Schur-concave since log is an increasing function. Then

$$(x_i-x_j)\left(\frac{\partial g}{\partial x_i}-\frac{\partial g}{\partial x_j}\right)=(x_i-x_j)\left(\frac{1}{x_i-2\lambda}-\frac{1}{x_j-2\lambda}\right)\leq 0$$

because $x_i\geq x_j\iff 0>x_i-2\lambda\geq x_j-2\lambda\iff\frac{1}{x_i-2\lambda}\leq\frac{1}{x_j-2\lambda}$.

Since f is continuous on X and symmetric, f is Schur-concave on $[0,2\lambda]^d$. The second claim follows since f is a decreasing function in every coordinate on $[0,2\lambda]^d$. □

A.1. Proof of Proposition 2.2

Proposition A.5. *Let B_1,B_2 be ℓ_∞-balls with radii λ centered at $x,y\in\mathbb{R}^d$ respectively; then*

$$\frac{Vol(B_1\cap B_2)}{Vol(B_1)}\geq 1-\frac{\|x-y\|_1}{2\lambda}.$$

Proof. Let $x,y\in\mathbb{R}^d$ and $c=\|x-y\|_1$. It holds that

$$\begin{aligned}
&\inf_{z\in\mathbb{R}^d_+\cap\langle\mathbf{1},z\rangle\leq c}\prod_{i=1}^{d}\left(1-\frac{z_i}{2\lambda}\right)\\
=&\inf_{u,v\in\mathbb{R}^d\cap\|u-v\|_1\leq c}\prod_{i=1}^{d}\left(1-\frac{|u_i-v_i|}{2\lambda}\right)\\
\leq&\prod_{i=1}^{d}\left(1-\frac{|x_i-y_i|}{2\lambda}\right)\\
=&\frac{Vol(\mathcal{B}_\infty(x,\lambda)\cap\mathcal{B}_\infty(y,\lambda))}{Vol(\mathcal{B}_\infty(x,\lambda))}\\
=&\frac{Vol(B_1\cap B_2)}{Vol(B_1)}
\end{aligned}$$

so it is sufficient to show that

$$\inf_{z\in\mathbb{R}^d_+\cap\langle\mathbf{1},z\rangle\leq c}\prod_{i=1}^{d}\left(1-\frac{z_i}{2\lambda}\right)=1-\frac{\|x-y\|_1}{2\lambda}. \tag{4}$$

The objective of optimization problem in (4) equals to (3) and is Schur-concave according to Proposition A.4. Thus is minimized by (e.g.,) $z=(c,0,0,\dots)$. In that case, the value of the objective is $1-\frac{c}{2\lambda}=1-\frac{\|x-y\|_1}{2\lambda}$. □

A.2. Proof of Proposition 2.5

Proposition A.6. *Let B_1, B_2 be the ℓ_∞ balls with radiii λ centered at $x, y\in[0,1]^d$; then*

$$\begin{aligned}&\frac{Vol(B_1\cap B_2)}{Vol(B_1)}\geq\\&\left(1-\frac{1}{2\lambda}\right)^{\lfloor\|x-y\|_1\rfloor}\left(1-\frac{\|x-y\|_1-\lfloor\|x-y\|_1\rfloor}{2\lambda}\right)\\&\geq\left(1-\frac{1}{2\lambda}\right)^{\|x-y\|_1}.\end{aligned}$$

The very last inequality holds when $2\lambda\geq 1$. Both of the inequalities are attainable.

Proof. Let $x, y\in[0,1]^d$ and $c=\|x-y\|_1$. It holds that

$$\begin{aligned}&\inf_{z\in[0,1]^d\cap\langle\mathbf{1},z\rangle\leq c}\prod_{i=1}^{d}\left(1-\frac{z_i}{2\lambda}\right)\\=&\inf_{u,v\in[0,1]^d\cap\|u-v\|_1\leq c}\prod_{i=1}^{d}\left(1-\frac{|u_i-v_i|}{2\lambda}\right)\\\leq&\prod_{i=1}^{d}\left(1-\frac{|x_i-y_i|}{2\lambda}\right)\\=&\frac{Vol(\mathcal{B}_\infty(x,\lambda)\cap\mathcal{B}_\infty(y,\lambda))}{Vol(\mathcal{B}_\infty(x,\lambda))}\\=&\frac{Vol(B_1\cap B_2)}{Vol(B_1)}\end{aligned}$$

so it is sufficient to show that

$$\inf_{z\in\mathbb{R}^d_+\cap\langle\mathbf{1},z\rangle\leq c}\prod_{i=1}^{d}\left(1-\frac{z_i}{2\lambda}\right)=\left(1-\frac{1}{2\lambda}\right)^{\lfloor\|x-y\|_1\rfloor}\left(1-\frac{\|x-y\|_1-\lfloor\|x-y\|_1\rfloor}{2\lambda}\right). \tag{5}$$

The objective of optimization problem in (5) equals to (3) and is Schur-concave according to Proposition A.4. Thus, it is minimized by a vector z such that $z_i=1$ at $\lfloor c\rfloor$ positions and $z_i=c-\lfloor c\rfloor$ at another position which is clearly a maximal element w.r.t. the majorization order. In that case, the value of the objective is

$$\left(1-\frac{1}{2\lambda}\right)^{\lfloor c\rfloor}\left(1-\frac{c-\lfloor c\rfloor}{2\lambda}\right).$$

If $x=\mathbf{0}$ and $y=z$, the inequality is tight. Furthermore, due to the convexity of the exponential function, we have for $a\geq 1$ and $0\leq x\leq 1$ that

$$\left(1-\frac{1}{a}\right)^{x}\leq 1-\frac{x}{a}.$$

Thus, we can simplify (5) to

$$\inf_{z\in\mathbb{R}^d_+\cap\langle\mathbf{1},z\rangle\leq c}\prod_{i=1}^{d}\left(1-\frac{z_i}{2\lambda}\right)\geq\left(1-\frac{1}{2\lambda}\right)^{\|x-y\|_1}$$

finishing the proof. □

A.3. Proof of Theorem 2.8

Theorem A.7. *Let $x \in [0,1]^d$. Let σ_i be an ordering induced by how far is x_i from boundary. That is;*

$$i \leq j \implies \min(x_{\sigma_i}, 1 - x_{\sigma_i}) \leq \min(x_{\sigma_j}, 1 - x_{\sigma_j}).$$

Then for any $c > 0$ such that there exists $y \in [0,1]^d$ with $\|x - y\|_1 = c$ it holds that

$$\inf_{y \in [0,1]^d \cap \mathcal{B}_1(x,c)} \frac{Vol(\mathcal{B}_\infty(x,\lambda) \cap \mathcal{B}_\infty(y,\lambda))}{Vol(\mathcal{B}_\infty(x,\lambda))} = \left(\prod_{i=1}^{T} \left(1 - \frac{\max\{x_{\sigma_i}, 1 - x_{\sigma_i}\}}{2\lambda}\right)\right)\left(1 - \frac{U}{2\lambda}\right)$$

where

$$T = \max_{k \in \mathbb{N}} \quad s.t. \sum_{i=1}^{i=k} \max(x_{\sigma_i}, 1 - x_{\sigma_i}) \leq c,$$

and

$$U = c - \sum_{i=1}^{i=T} \max(x_{\sigma_i}, 1 - x_{\sigma_i}).$$

Proof. We equivalently rewrite the problem as

$$\inf_{z \in \mathcal{X} \cap \langle \mathbf{1}, z \rangle = c} \frac{Vol(\mathcal{B}_\infty(x,\lambda) \cap \mathcal{B}_\infty(y,\lambda))}{Vol(\mathcal{B}_\infty(x,\lambda))} \tag{6}$$

where $\mathcal{X} = \bigtimes_{i=1}^{d} [0, \max\{x_i, 1 - x_i\}]$ so that from z we recover y as $y_i = x_i \pm z_i$ where either $0 \leq x_i + z_i \leq 1$ or $0 \leq x_i - z_i \leq 1$.

The objective of the optimization problem (6) is again (3). Thus, we only need to find a maximizing element w.r.t. the majorization order and the Theorem describes how to find it. To see that, we notice that every $1 \leq k \leq d$, the sequence

$$z_i = \begin{cases} \max\{x_i, 1 - x_i\}, & \text{if } \sigma_i^{-1} < T + 1 \\ U, & \text{if } \sigma_i^{-1} = T + 1 \\ 0, & \text{if } \sigma_i^{-1} > T + 1 \end{cases}$$

where σ_i^{-1} is the inverse ordering, that is, $\sigma_{\sigma_i^{-1}} = i$, maximizes $\sum_{i=1}^{k} z_i$ under the constraint $\sum_{i=1}^{d} z_i = c$. □

B. Hoeffding's bound

Theorem B.1 (Hoeffding's inequality)**.** *Let $X_1, \dots X_n$ be random variables with $\frac{1}{n}\mathbb{E}[\sum_{i=1}^{n} X_i] = \mu$ and $0 \leq X_i \leq 1$. Then it holds that*

$$\mathbb{P}\left(\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) - \mu \geq t\right) \leq e^{-2t^2 n}.$$

for any $t \geq 0$.

We rewrite the inequality as

$$\mathbb{P}\left(\left(\frac{1}{n}\sum_{i=1}^{n} X_i\right) - t \geq \mu\right) \leq e^{-2t^2 n} \leq \alpha.$$

We want to compute t - that is, how much do we need to subtract from the average so that the probability that the result of the subtraction will be larger than the mean is small.

$$e^{-2t^2 n} = \alpha \quad \implies \quad t = \sqrt{\frac{\ln(\alpha)}{-2n}}$$

C. Ablations

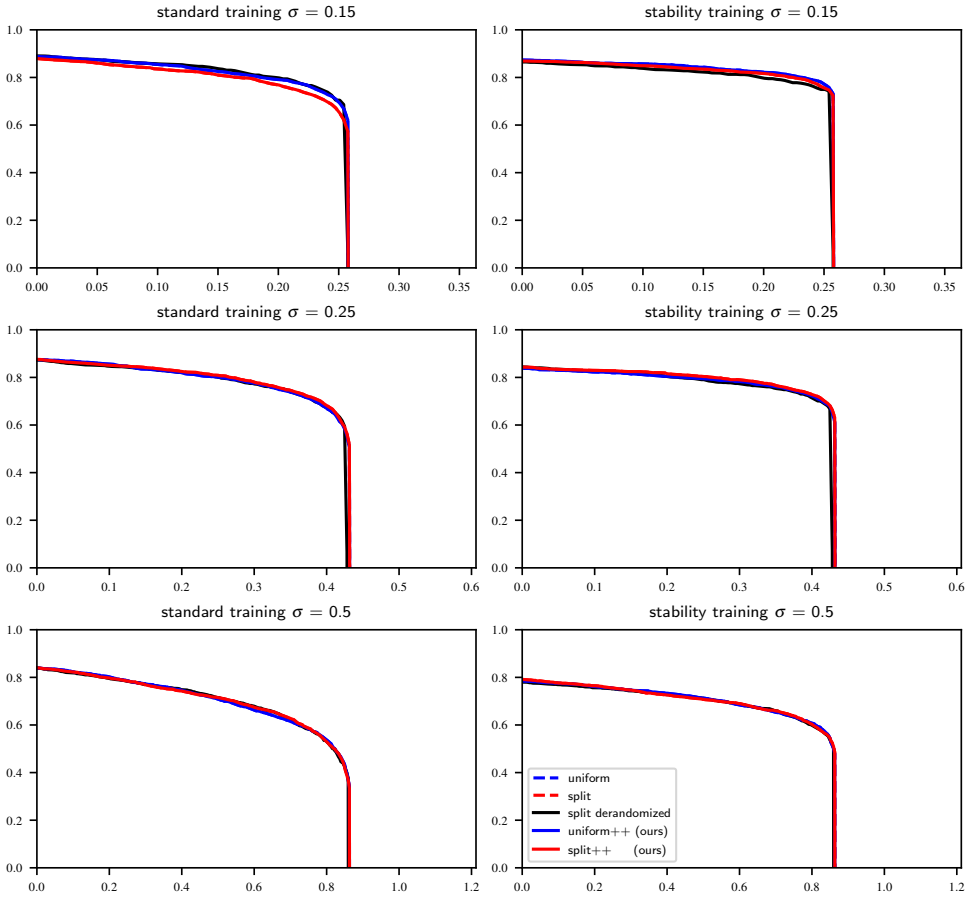


Figure 7: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns.

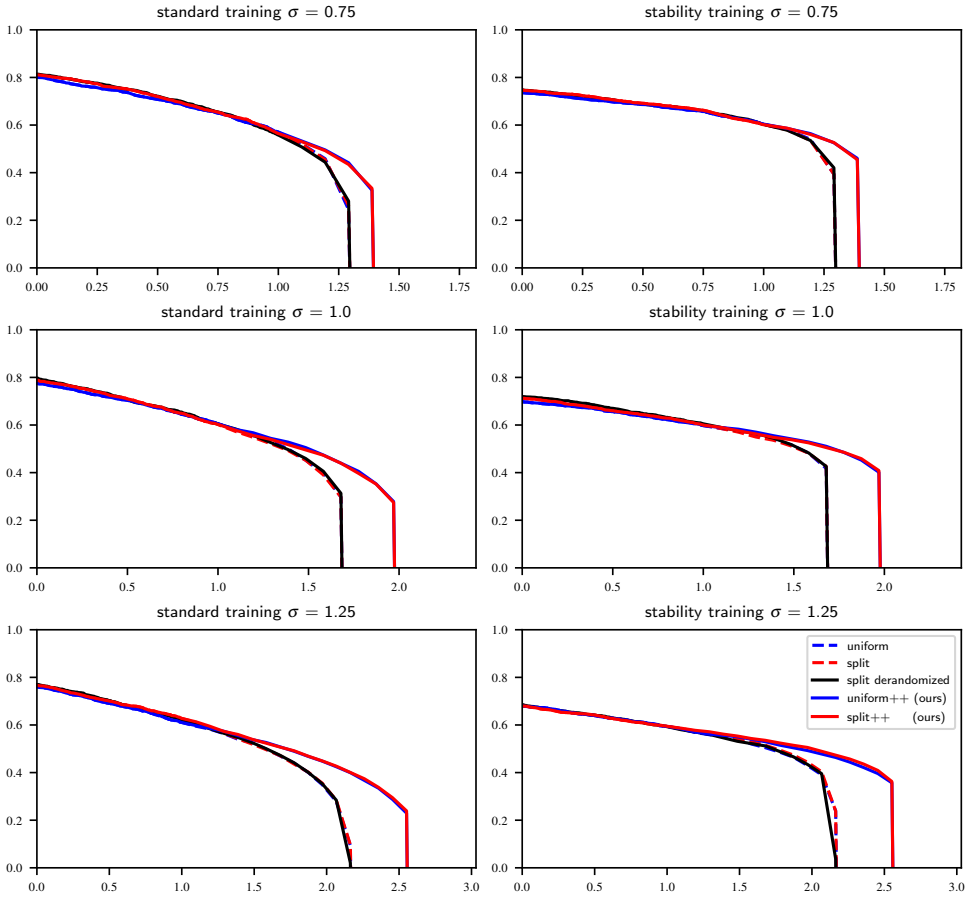


Figure 8: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns.

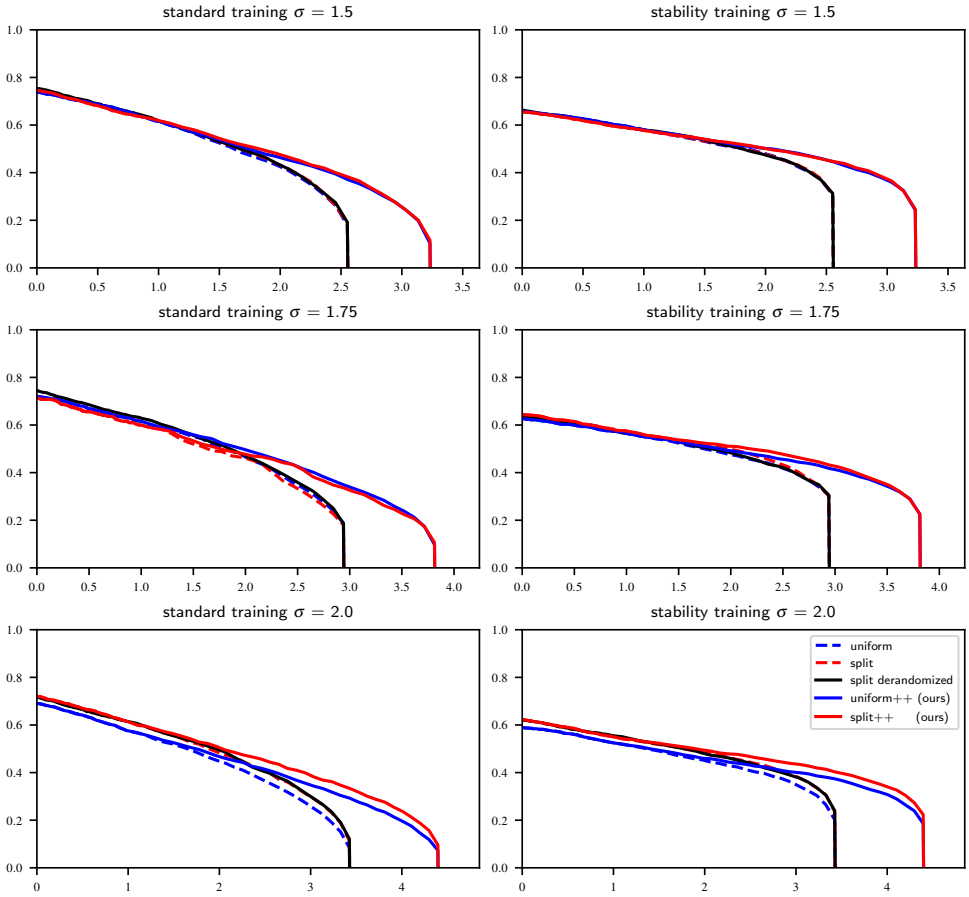


Figure 9: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns.

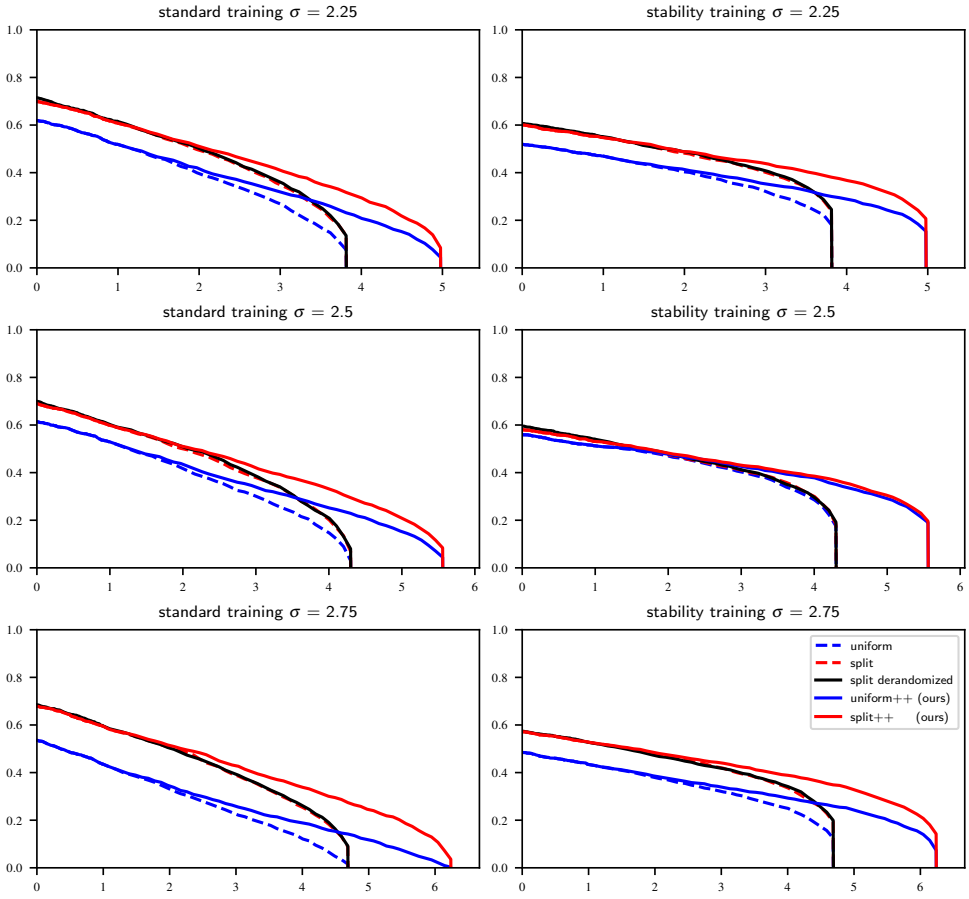


Figure 10: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns.

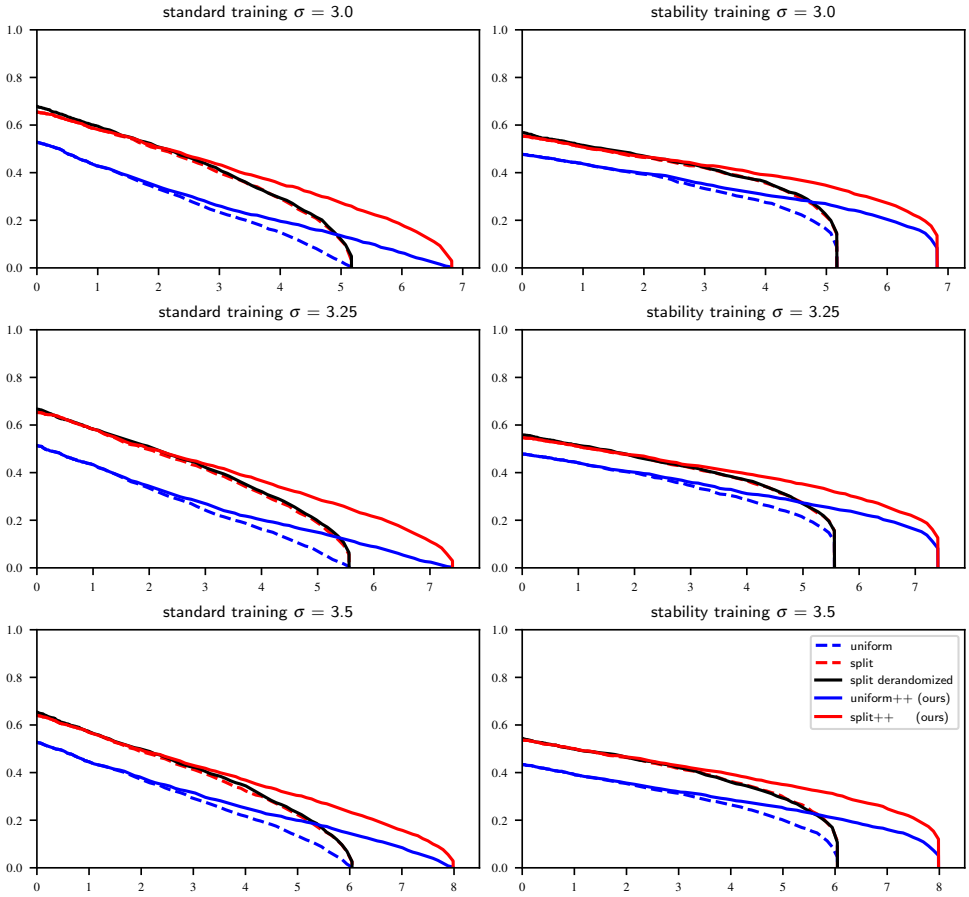


Figure 11: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns.

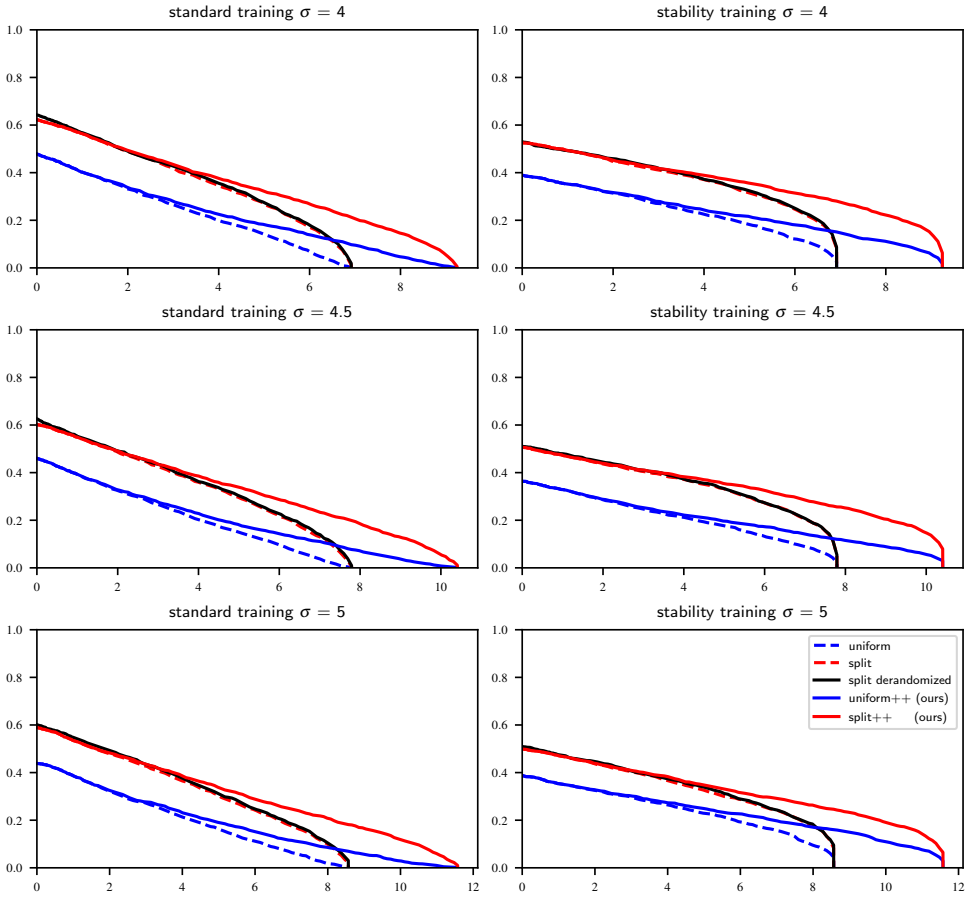


Figure 12: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns.

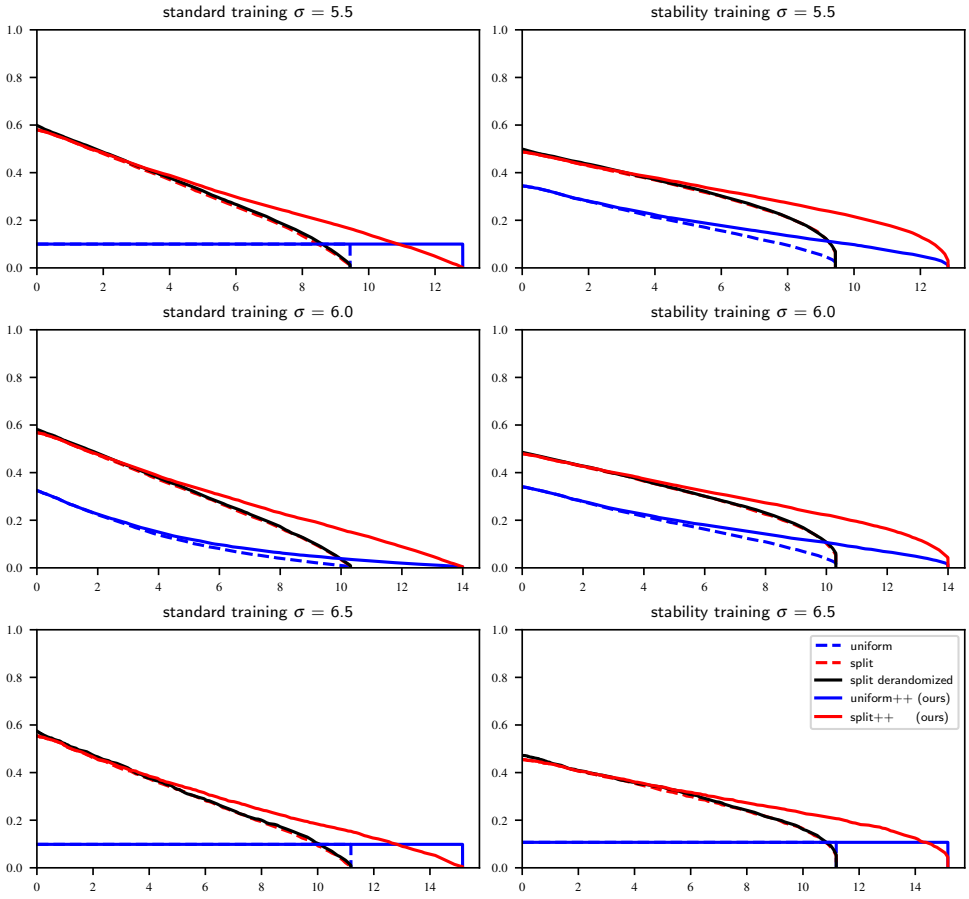


Figure 13: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns. Note that the uniform noise training sometimes converges to an (apparently) constant classifier

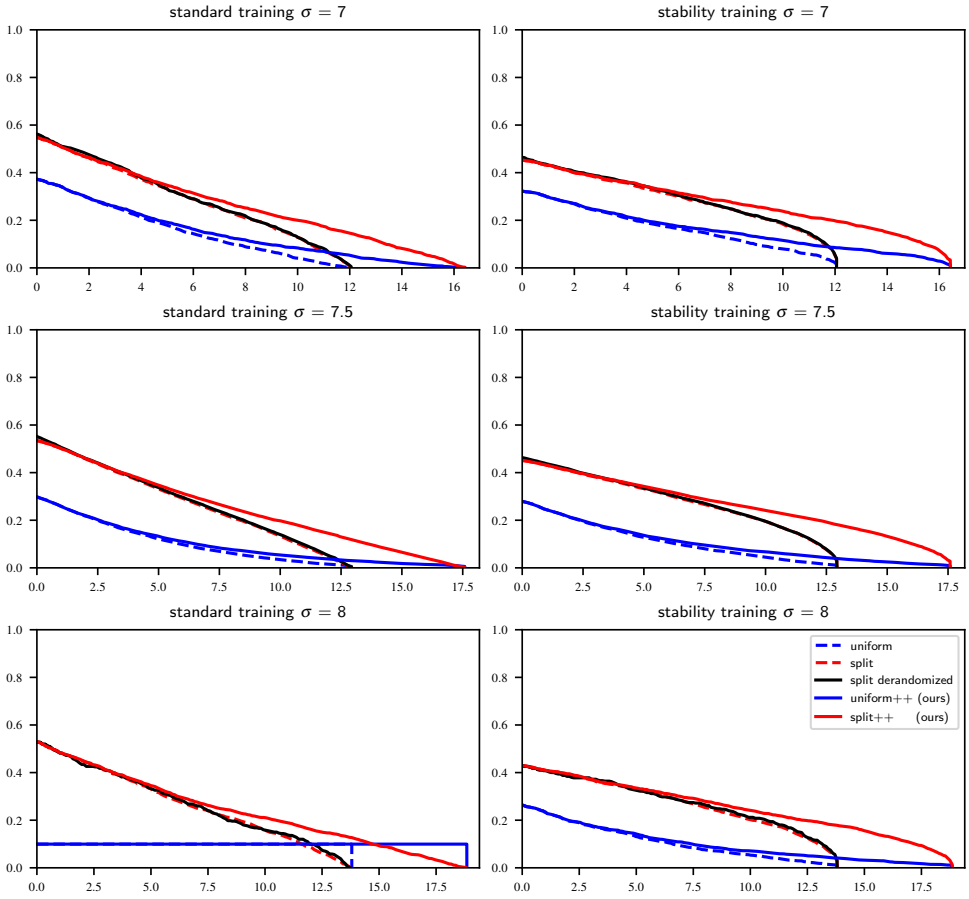


Figure 14: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns. Note that the uniform noise training sometimes converges to an (apparently) constant classifier

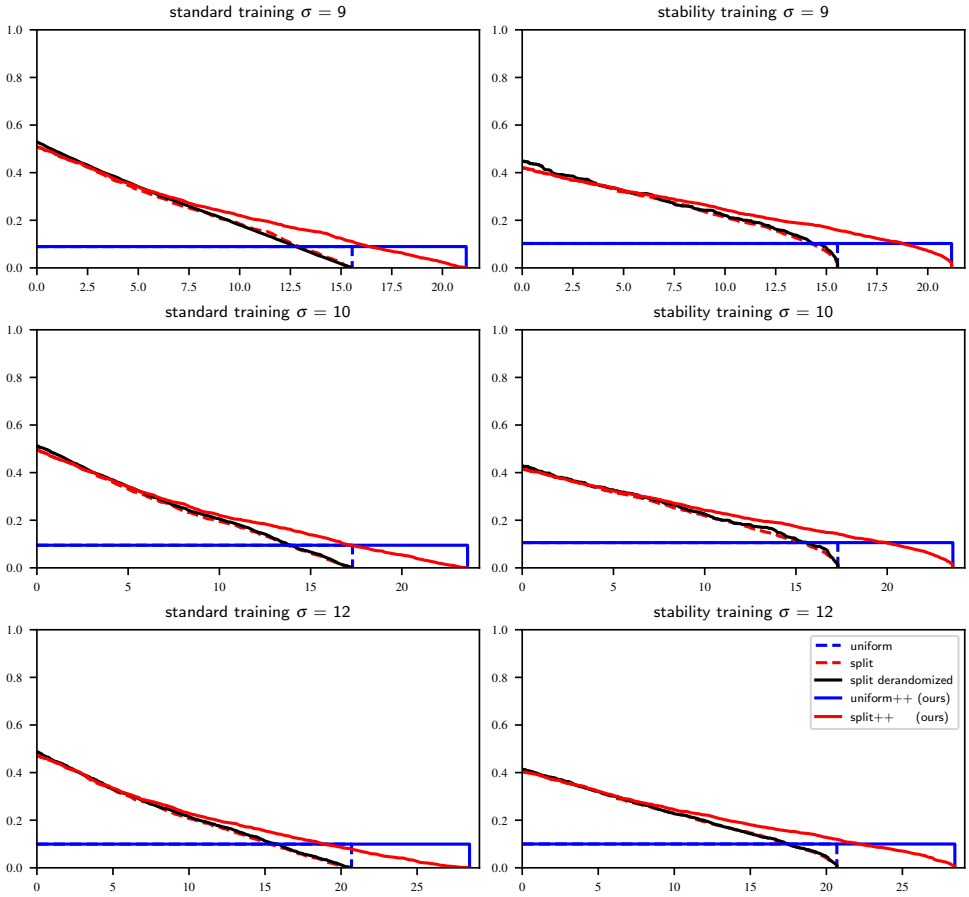


Figure 15: Robustness curves on CIFAR-10 for different methods. The noise magnitudes differ in rows and the training method differ in columns. Note that the uniform noise training converges to an (apparently) constant classifier

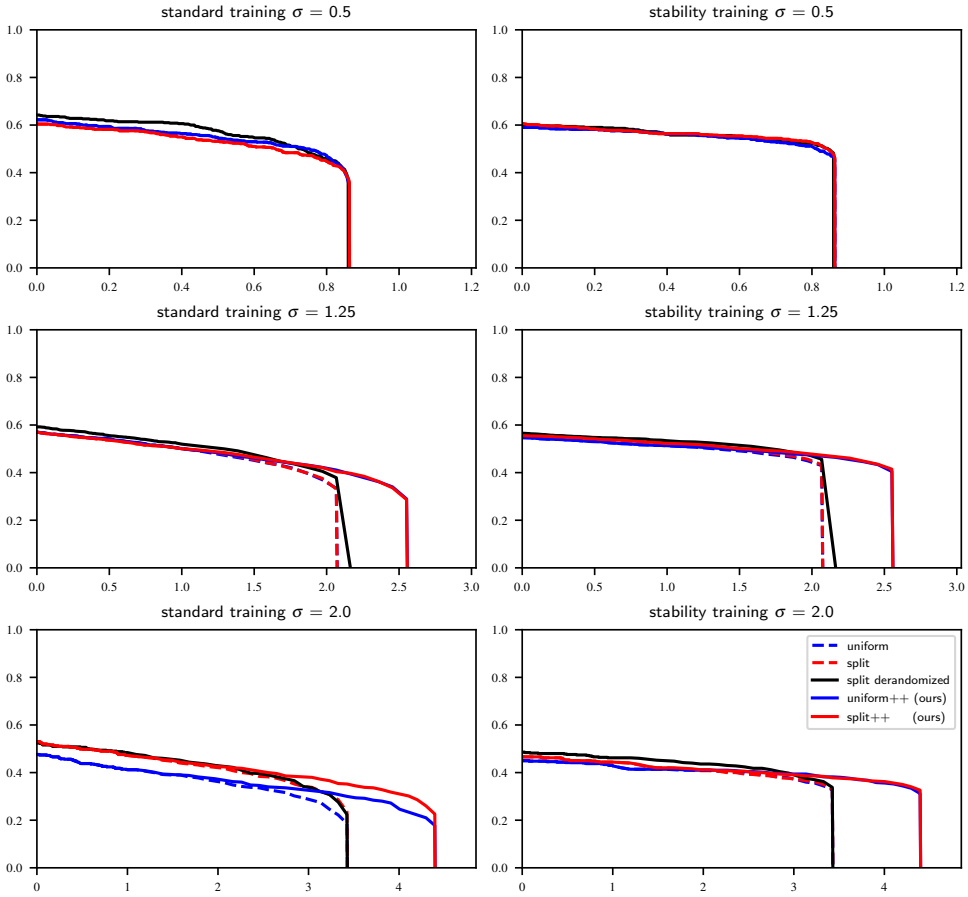


Figure 16: Robustness curves on ImageNet for different methods. The noise magnitudes differ in rows and the training method differ in columns.

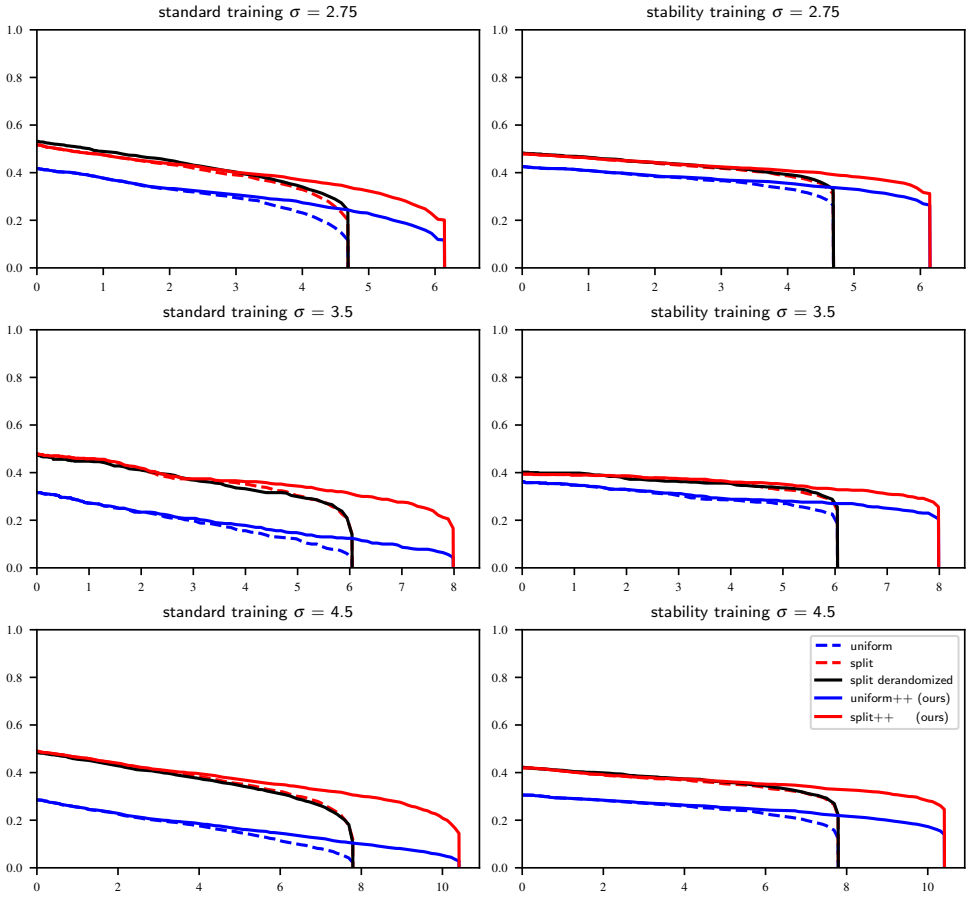


Figure 17: Robustness curves on ImageNet for different methods. The noise magnitudes differ in rows and the training method differ in columns.

7.4 Treatment of Statistical Estimation Problems in Randomized Smoothing

Treatment of Statistical Estimation Problems in Randomized Smoothing for Adversarial Robustness

Václav Voráček
Tübingen AI center, University of Tübingen
vaclav.voracek@uni-tuebingen.de

Abstract

Randomized smoothing is a popular certified defense against adversarial attacks. In its essence, we need to solve a problem of statistical estimation which is usually very time-consuming since we need to perform numerous (usually 10^5) forward passes of the classifier for every point to be certified. In this paper, we review the statistical estimation problems for randomized smoothing to find out if the computational burden is necessary. In particular, we consider the (standard) task of adversarial robustness where we need to decide if a point is robust at a certain radius or not using as few samples as possible while maintaining statistical guarantees. We present estimation procedures employing confidence sequences enjoying the same statistical guarantees as the standard methods, with the optimal sample complexities for the estimation task and empirically demonstrate their good performance. Additionally, we provide a randomized version of Clopper-Pearson confidence intervals resulting in strictly stronger certificates. The code can be found at `https://github.com/vvoracek/RS_conf_seq`.

We encourage the readers only interested in statistics to start at Subsection 2.1.

1 Introduction

Adversarial robustness: It is well known that a tiny, adversarial, perturbation of the input can change the output of basically any undefended machine learning model (Biggio et al., 2013; Szegedy et al., 2014); this is unpleasant and we continue in the mitigation of the problem. There are two main lines of work tackling this problem: (1) Empirical: the standard approach here is to use adversarial training (Madry et al., 2018; Goodfellow et al., 2014) where the model is trained on adversarial examples. This approach does not provide guarantees, only empirical evidence suggesting that the model may be robust. With stronger attacks, we might (yet again) realize it is not the case. (2) Certified: with formal robustness guarantees for the model. We will focus on this, and in particular on *randomized smoothing* (Lecuyer et al., 2019) which is currently the strongest certification method[1]. We will not cover other certification methods and we refer the reader to the survey Li et al. (2023) instead. We consider the standard task of certified robustness; the goal is to decide if the decision of a classifier F at a particular input x is robust against additive perturbations δ such that $\|\delta\| \leq r$ for some norm $\|\cdot\|$. Formally, we ask if $F(x) = F(x')$ whenever $\|x - x'\| \leq r$.

Randomized smoothing is a framework providing state of the art formal guarantees on the adversarial robustness for many datasets. One of its benefits lies in the fact that there are no assumptions on the model, making it possible to readily transfer the methods from defending image classifiers against sparse pixel changes to different modalities; e.g., defending large language models against change of

[1] see leaderboard `https://sokcertifiedrobustness.github.io/leaderboard/`

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

some letters/words/tokens. Randomized smoothing transforms any undefended classifier $F : \mathbb{R}^d \to \mathcal{Y}$ by a smoothing distribution φ into a smoothed classifier $H_\varphi(x) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}_{\delta \sim \varphi}[\![F(x+\delta) = y]\!]$ for which robustness guarantees exist. We postpone the details for later. During the certification process, we need to estimate the maximum probability of a multinomial distribution from samples as the exact computation is intractable. This statistical estimation problem is the focus of this paper.

Speed issues: The main weakness of randomized smoothing is the extensive time required for both prediction and certification, making it troublesome for real-world applications. There is an inherent trade-off between the allowed probability of incorrectly claiming robustness[2] (type-1 error, α), the probability of incorrectly claiming non-robustness (type-2 error, β), and the number of samples used n. The standard practice is to set $\alpha = 0.001$, $n = 100\ 000$ and the value of β is then implicit. it might not be the most practically relevant setting since the implicitly set value of β is exponentially small in n when the sample is not close to the threshold. The claim is made precise in Example A.1.

The arguably more relevant setting is to set the values of α and β and leave n implicit. This is much more challenging since it is no longer possible to draw the predetermined number of samples and use a favourite concentration inequality. We propose a new certification procedure using confidence sequences to adaptively (and optimally) deciding how many samples to draw addressing the problem.

Contributions:

- We introduce a new, strictly better version of Clopper-Pearson confidence intervals for estimating the class probabilities in Subsection 2.1. The presented interval is optimal, and thus is the ultimate solution to the canonical statistical estimation of randomized smoothing.
- We propose new methods for the certification utilizing confidence sequences (instead of confidence intervals) in Subsection 2.2. This allows us to draw *just enough* samples to certify robustness of a point; greatly decreasing the number of samples needed.
- We provide a complete theoretical analysis of the proposed certification procedures. In particular, we provide matching (up to a constant factor) lower-bounds and upper-bounds for the width of the respective confidence intervals. We invert the bound and show that the certification procedure has the optimal sample-complexity in an adaptive estimation task.
- We provide empirical validation of the proposed methods confirming the theory.

Notation: Bernoulli random variable with mean p is denoted as $\mathcal{B}(p)$ and binomial random variable is $\mathcal{B}(n, p)$. Random variables are in capitals (X) and the realizations are lowercase (x). We type sequences in bold and denote $\mathbf{x}_{:t}$ first t elements of $\mathbf{x}$. We write $a \lesssim b$ if there exists a universal constant $C > 0$ such that $a \leq Cb$. If $a \lesssim b$ and $b \lesssim a$, then we write $a \asymp b$. Iverson bracket $[\![\Phi]\!]$ evaluates to 1 if Φ is true and to 0 otherwise.

1.1 Paper organization

First, in Section 2 we introduce randomized smoothing, then, in Subsection 2.1, we introduce Clopper-Pearson confidence intervals, show that they are conservative and propose their improved (optimal) randomized version. In Subsection 2.2 we discuss shortcomings of confidence intervals and introduce confidence sequences and provide lower and upper bounds for their performance. We use the confidence sequences in Section 3 and benchmark them on a sequential estimation task.

2 Randomized Smoothing

As outlined in Introduction, consider a classifier $F : \mathbb{R}^d \to \mathcal{Y}$ and let the class probabilities under additive noise φ be $h_\varphi(x)_y = \mathbb{P}_{\delta \sim \varphi}[F(x + \delta) = y]$. Denote the highest probability (breaking ties arbitrarily) class in the original point $A = \arg\max_{y \in \mathcal{Y}} h_\varphi(x)_y$ and the second-highest probability class $B = \arg\max_{y \in \mathcal{Y} \setminus A} h_\varphi(x)_y$. Let the corresponding probabilities be p_A and p_B respectively. Recalling that $H_\varphi(x) = \arg\max_{y \in \mathcal{Y}} h_\varphi(x)_y$, then for a certain function $r : [0, 1]^2 \to \mathbb{R}_+$ we have

$$\|x - x'\| \leq r(p_A, p_B) \implies H_\varphi(x) = H_\varphi(x').$$

[2] of an input for a model at a certain radius

This r depends on the smoothing distribution φ and the considered norm. For example, if the considered norm is ℓ_2 and φ is isotropic Gaussian with standard deviation σ, then $r(p_A, p_B) = \frac{\sigma}{2}(\Phi^{-1}(p_A) - \Phi^{-1}(p_B))$ where Φ^{-1} is Gaussian quantile function Cohen et al. (2019). Note that in general, $r(\cdot, \cdot)$ is increasing in the first coordinate and decreasing in the second one. The intuition is that the larger the value of p_A at x, the larger it will be also in the neighborhood of x; similarly for p_B. It is common in the literature to use the bound $p_B \leq 1 - p_A$ and thus certify $r(p_A, 1 - p_A)$. We stick to the convention in the paper and discuss the topic in more details in Appendix B.

Statistical estimation: The crux of the paper lies in the statistical estimation problems for randomized smoothing. We consider the abstract framework for randomized smoothing, so the proposed techniques can be used as a drop-in replacement in all randomized smoothing works with a statistical-estimation component (i.e., not in the de-randomized ones such as Levine & Feizi (2021)). We do not only propose methods that work good empirically, we also provide theoretical analysis suggesting that we solve the problems optimally in a certain strong sense. The main focus is on the following two constructs.

1. Confidence intervals: A standard component of randomized smoothing pipelines is the Clopper-Pearsons confidence interval. It is known to be conservative[3]; thus, the certification procedures are unnecessarily underestimating the certified robustness. We provide the *optimal* confidence interval for binomial random variables, resolving this issue completely.
2. Confidence sequences: In the standard randomized smoothing practice, we draw a certain, predetermined, number of samples and then we compute the certified radius on a confidence level $1 - \alpha$. We improve on this by allowing for adaptive estimation procedures employing confidence sequences; We demonstrate the performance in the standard task of adversarial robustness, where we want to decide if a point is robust at radius r with type-1 (resp. 2) error rates α (resp. β) using as few samples as possible.

Literature review of randomized smoothing: The most relevant related works are Horváth et al. (2022); Chen et al. (2022) and they are discussed in Section 3. Here we briefly summarize literature relevant to randomized smoothing in general. The choice of the smoothing distribution φ is a crucial decision determined mainly by the threat model with respect to which we want to be robust. For example, if we are after certifying ℓ_1 robustness, we choose a uniform distribution in a $d-$dimensional ℓ_∞ ball (Lee et al., 2019; Yang et al., 2020), or better, splitting noise (Levine & Feizi, 2021), but we do not go into details here. Alternatively, for $p-$norms, $p \geq 2$ one would usually use a $d-$dimensional normal distribution Lecuyer et al. (2019); Cohen et al. (2019). The variance of the distribution based on how large perturbations do we allow in our threat model. We refer the reader to Yang et al. (2020) for a broader discussion on the smoothing distributions. It is possible to use methods in the spirit of randomized smoothing to certify other threat models, such as patch attacks Levine & Feizi (2020) and sparse attack Bojchevski et al. (2020), which can be readily extended to other modalities. Sometimes, the "smoothing" distribution can be made supported on a small, discrete set and then we can evaluate the expectation exactly, yielding deterministic certification (often called de-randomized smoothing (Levine & Feizi, 2021)). See also Kumari et al. (2023) for a survey on randomized smoothing containing examples of when the certification is beyond additive ℓ_p-norm threat model; even such techniques use the Bernoulli estimation subroutine.

2.1 Confidence Intervals

We do not have access to the class A probability p_A and only have to estimate it from binomial samples; hence, the name *randomized* smoothing. Because of the randomness, we can only provide probabilistic statements about the robustness of a classifier in the following spirit "with probability at least $1 - \alpha$, robust radius is at least r" for a small α, usually 0.001. This failure probability corresponds to the event of overestimating p_A and we control it with the help of confidence intervals.

The standard choice for calculating the upper confidence interval is the Clopper-Pearson interval, sometimes called *exact*. Regardless of this pseudonym, it is in reality conservative. In this subsection, we introduce the Clopper-Pearson confidence interval for the mean of binomial random variables, demonstrate its limitations and introduce its (better) randomized version.

[3] See `https://en.wikipedia.org/wiki/Binomial_proportion_confidence_interval`.

Definition 2.1 (Confidence interval for binomials)**.** Let u, v map sample to a real number. They form a (possibly randomized) confidence interval $I(x) = [u(x), v(x)]$ with coverage $1 - \alpha$ if for any $p \in [0, 1]$ it holds that

$$\mathbb{P}_{X \sim \mathcal{B}(n,p), I} \left(p \in I(X) \right) \geq 1 - \alpha.$$

We will mainly use one-sided confidence intervals; that is, $u(\cdot) = 0$ (lower confidence interval) or $v(\cdot) = 1$ (upper confidence interval). When we will talk about probability of confidence interval containing the parameter, it will be in the frequentist sense, keeping in mind that confidence intervals provide no guarantees post-hoc for any individual estimation.

Clearly, if $I(x) = [0, 1]$ regardless of x, it will be a valid confidence interval but rather useless; thus we aim for short intervals. Ideally it would hold for every p that $\mathbb{P}_X(p \notin I(X)) = \alpha$, otherwise, some values are included in the confidence intervals unnecessarily often they can be shortened. In the following we introduce the standard Clopper-Pearson confidence intervals Clopper & Pearson (1934).

Definition 2.2 (Clopper-Pearson intervals)**.** One sided upper interval is defined as $v(x) = 1$ and

$$u(x) = \inf\{p \,|\, \mathbb{P}(\mathcal{B}(n, p) \geq x) > \alpha\}.$$

The lower one is defined as $u(x) = 0$ and

$$v(x) = \sup\{p \,|\, \mathbb{P}(\mathcal{B}(n, p) \leq x) > \alpha\}.$$

Amongst the deterministic confidence intervals, they are the shortest possible; however, they are in general conservative. In the binomial case $\mathcal{B}(n, p)$, there are only $n + 1$ possible outcomes; and thus only $n + 1$ possible confidence intervals suggesting that the actual coverage can be $1 - \alpha$ only for at most $n + 1$ values of p. The problem strikingly arises for upper confidence interval for large values of p. When we sample from $\mathcal{B}(n, p)$, regardless of the outcome, all values larger than $\sqrt[n]{\alpha}$ are contained in the confidence interval. This is a usual problem in the context of randomized smoothing, leading to sharp drops towards the end of robustness curves. We demonstrate this sub-optimality in the first part of Example A.2. We mitigate this problem by introducing randomness into the confidence intervals. They will still have the desired coverage level $1 - \alpha$, but will be shorter. Intuitively, we do so by "interpolating" between the deterministic confidence intervals in the spirit of Stevens (1950).

Definition 2.3 (Randomized Clopper-Pearson intervals)**.** Let W be uniform on the interval $[0, 1]$. The randomized one sided upper interval is defined as $v_r(x) = 1$ $u_r(x) = u'_r(x, W)$ where

$$u'_r(x, w) = \inf\{p \,|\, \mathbb{P}(\mathcal{B}(n, p) > x) + w\mathbb{P}(\mathcal{B}(n, p) = x) > \alpha\}.$$

The lower one is defined as $u_r(x) = 0$ and $v_r(x) = v'_r(x, W)$ where

$$v'_r(x, w) = \sup\{p \,|\, \mathbb{P}(\mathcal{B}(n, p) < x) + w\mathbb{P}(\mathcal{B}(n, p) = x) > \alpha\}.$$

Proposition 2.4. *Randomized Clopper-Pearson interval (I_{rCP}) have coverage exactly $1 - \alpha$. Furthermore, for any confidence interval I at level $1 - \alpha$, and any $p \geq q \in [0, 1]$ it holds that*

$$\mathbb{P}_{X \sim \mathcal{B}(n,p)}(q \in I(X)) \geq \mathbb{P}_{X \sim \mathcal{B}(n,p)}(q \in I_{rCP}(X)).$$

The proof is in the Appendix D and we remark that the interval can be efficiently found by binary search. Proposition 2.4 implies that the randomized Clopper-Pearson bounds are optimal and all the other confidence intervals for binomial random variables are more conservative. It remains to demonstrate the advantage of the randomized confidence intervals. We refer to Figure 1 for the comparison of the randomized and deterministic Clopper-Pearson confidence intervals and how they affect the robustness. We note that the most significant difference is towards the high values of p and for certification functions r such that $\lim_{p \to 1} r(p) = \infty$, such as when smoothing with normal distribution. This explains the common sharp drop by the end of the robustness curve.

Width of confidence intervals: For the simplicity of exposition, let the width of a confidence interval at level $1-\alpha$ with n samples be $\asymp \sqrt{\log(1/\alpha)/n}$. This way, we hide the dependency on p into $\asymp$. In the full generality, the width of the confidence intervals exhibits many decay regimes between the rates $\sqrt{p(1-p)\log(1/\alpha)/n}$ (when $np(1-p) \gg 1$) and $\log(1/\alpha)/n$ (when $np(1-p) \asymp 1$). Our algorithms capture the correct scaling of the confidence intervals. See Boucheron et al. (2013) and the discussion on Bennett's inequality which captures the correct rates for Bernoulli mean estimation.

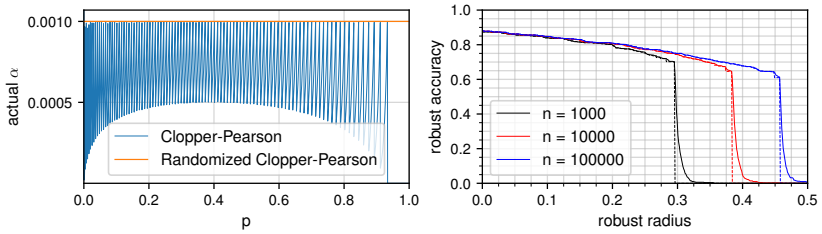


Figure 1: **left**: Comparison of coverages of confidence intervals for the mean estimation of $\mathcal{B}(100, p)$ when $\alpha := 0.001$. Note that for $p > \sqrt[n]{\alpha} \sim 0.93$, the coverage is 1. **right**: Comparison of ℓ_2-robustness curves with the standard (dashed) or the randomized (solid) Clopper-Pearson bounds on a CIFAR-10 dataset under the standard setting. The experimental details are in Appendix C.

2.2 Confidence Sequences

Limitations of confidence intervals: In order to compute the confidence intervals presented in the previous subsection, we need to collect samples and then run an estimation procedure once which brings certain limitations. Consider the following two scenarios: (1) It might be the case that we do not need all $100\,000$ samples and after only 10 it would be enough for our purposes because we could already conclude that the point cannot be certified here; thus, we wasted $99\,990$ samples. (2) Alternatively, we could see that even 10^5 samples are not enough, and we need to draw more samples. However, we have already spent our failure budget α, so we cannot even carry another test at all.

This motivates the introduction of confidence sequences. They generalize confidence intervals in the way that they provide a confidence interval after every received sample such that we control the probability that the underlying parameter is contained in *all* the confidence intervals *simultaneously*.

Definition 2.5 (Confidence sequence)**.** Let $\{u_t, v_t\}_{t=1}^{\infty}$ be mappings from a sequence of observations to a real number. They form a confidence sequence $I_t(\mathbf{x}_{:t}) = [u_t(\mathbf{x}_{:t}), v_t(\mathbf{x}_{:t})]$ for all $t \geq 1$ with confidence level $1 - \alpha$ if

$$\mathbb{P}\left(p \in I_t(\mathbf{X}_{:t}), \forall t > 1\right) \geq 1 - \alpha$$

for any $p \in [0, 1]$, where $\mathbf{X}$ is an infinite sequence of Bernoulli random variables $\mathcal{B}(p)$.

Remark 2.6. Since we want the estimated parameter to be contained in all the confidence intervals simultaneously, we will have by convention that $I_{t+1}(x_{:t+1}) \subseteq I_t(x_{:t})$.

To simplify presentation, we would consider the symmetric ones; i.e., those where we consider the two possible failures - when we overestimate or underestimate the mean - to be equal. However, they will be constructed from two one-sided bounds, so the generalization is straightforward and will not be discussed.

Related work: The construction of confidence sequences based on union bound employs the doubling trick which is widely used in online learning to convert fixed-horizon algorithms to anytime algorithms. In this direction, we refer to Mnih et al. (2008) as the direct predecessor of this work, where they used similar techniques but did not explicitly construct confidence sequences. In the confidence sequence literature, this technique is similar to stitching of Howard et al. (2020). The stitched confidence intervals of Howard et al. (2020) are generally shorter by a small constant factor, but the analysis and generalization become complicated, contrasting with our approach.

The construction based on betting is in the spirit of Orabona & Jun (2023) (the reference contains an excellent survey on the topic). The difference mainly lies in the fact that we are interested in Bernoulli random variables, which allows us to use specialized tools at places, as opposed to the referred work, which considers bounded random variables. As an analogy to confidence intervals, the previous work constructed Bernstein-type inequalities, while we constructed Clopper-Pearson-like bounds. In Ryu & Bhatt (2024), the authors improved over Orabona & Jun (2023) in certain aspects in the case of $[0, 1]$-valued random variables. Notably, in Section 3, they considered the special case - $\{0, 1\}$-valued random variables and their results are greatly overlapping those in Section 2.4.

2.3 Union bound confidence sequence

A natural way how to extend the confidence intervals to confidence sequences is to construct a confidence interval at every time step and use a union bound to control the total failure probability. In the following, we first show that a naive application of this approach is asymptotically suboptimal, and then we provide a way how to construct optimal confidence intervals in a certain strong sense.

Intuition on the width of confidence sequences: For any random variable with finite variance, the optimal width of the confidence interval for the mean parameter scales as $\sqrt{\log(1/\alpha)/t}$ with the increasing number of samples t at confidence level $1-\alpha$ Lugosi & Mendelson (2019). On the other hand, it is well known that the width of the optimal confidence sequence scales as $\sqrt{(\log(1/\alpha)+\log\log t)/t}$ as t increases due to the law of iterated logarithms Ledoux & Talagrand (1991). A naive use of union bound, computing a confidence interval using failure probability at time step t, $\alpha_t = \frac{\alpha c}{t^\gamma}$ for some c and $\gamma > 1$ such that $\sum_{i=1}^{\infty} \alpha_t = \alpha$ yields a confidence sequence whose width scales as $\sqrt{\log(1/\alpha_t)/t} \approx \sqrt{(\log(t)+\log(1/\alpha))/t}$. We cannot choose any monotonous α_t schedule decaying slower because even for $\gamma = 1$ we still keep the $\log$ factor while the sum $\sum_{i=1}^{\infty} \delta_t$ diverges.

Now consider non-monotonous schedules of α_t, two key ideas follows. (1) In order to have the optimal rate $\log(1/\alpha_t) \approx \log(1/\alpha) + \log\log t$, we need $\alpha_t \asymp \alpha/\log t$. Clearly, if this holds for all t, then $\sum_{t=1}^{\infty} \alpha_t$ diverges. (2) A confidence interval at time t is also a valid confidence interval for all $t' > t$. Furthermore, if t' is not much larger than t, then it may still asymptotically have the optimal width up to a multiplicative constant. Thus, updating the confidence sequence when t is a power of (say) 2 result in the optimal width. This reasoning is formalized in the following theorem.

Theorem 2.7. *Fix $\alpha > 0$. Consider a sequence*

$$\alpha_t = \begin{cases} \frac{\alpha}{k(k+1)} & \text{if } t = 2^k \text{ for integer } k, \\ 0 & \text{otherwise.} \end{cases}$$

Then Algorithm 1 produces a confidence sequence at level $1-\alpha$ of the following width which is attained in the worst case

$$\varepsilon_t \lesssim \sqrt{\frac{\log(1/\alpha) + \log\log(t)}{t}}$$

where ε_t is $U - L$ at time t, and the confidence intervals are randomized Clopper-Pearson intervals.

The proof is in Appendix E. We remark that the statement of Theorem 2.7 prioritizes simplicity over its full generality. The generalization to other schedules of α_t from the second bullet point as described in the previous paragraph is routine and described in detail in Appendix C.2.

Corollary 2.8. *The asymptotic rate of 2.7 is optimal due to law-of-iterated-logarithm (Ledoux & Talagrand, 1991) and even in the finite sample regime due to Balsubramani (2014)[Theorem 2].*

2.4 Confidence sequences based on betting

A recent alternative approach to confidence sequences is based on a hypothetical betting game. For the illustration, consider a fair sequential game; e.g., sequentially betting on outcomes of a coin. If we guess the outcome correctly, we win the staked amount, otherwise we lose it. If the coin is fair, in expectation, our wealth stays the same. On the other hand, if the game is not fair and the coin is biased, we can win money. For example, if the true head-probability is 0.51, we start increasing our wealth in an exponential fashion, see Example A.3; thus, if we win lots of money, we can conclude that the game is not fair. We instantiate a betting game for every possible mean $0 \leq p \leq 1$ that would be fair if the true mean is p. Then we observe samples of the random variable and as soon as we win enough money, we drop that particular p from the confidence sequence. To make things formal, we introduce the necessary concepts from probability theory. The evolution of our wealth throughout a fair game is modeled by martingales[4], sequences of random variables for which, independently of the past, the expected value stays the same.

[4]It would be historically accurate to say that martingales actually model fair games.

Definition 2.9 (Martingale)**.** A sequence of random variables $W_1, W_2, \ldots$ is called a *martingale* if for any integer $n > 0$, we have $\mathbb{E}(|W_n|) < \infty$ and $\mathbb{E}(W_{n+1}|W_1, \ldots, W_n) = W_n$. If we instead have $\mathbb{E}(W_{n+1}|W_1, \ldots, W_n) \leq W_n$, then the sequence is called a *supermartingale*.

Algorithm 1 Union-Bound Confidence Sequence

$t, H, K, L, U \leftarrow 0, 0, 0, 0, 1$
loop
  Obtain random x
  $H \leftarrow H + x$
  $t \leftarrow t + 1$
  if $t = 2^K$ **then**
    $K \leftarrow K + 1$
    $\alpha_t \leftarrow \alpha/(K(K+1))$
    $L \leftarrow \max\{L, \text{LowConfInt}(H, t, \alpha_t)\}$
    $U \leftarrow \min\{U, \text{UppConfInt}(H, t, \alpha_t)\}$
  end if
end loop

Algorithm 2 Betting Confidence Sequence

$\text{LOGQ}, t, H, L, U \leftarrow 0, 0, 0, 0, 1$
loop
  $\hat{q} \leftarrow (H + 1/2)/(t + 1)$
  Obtain random x
  $H \leftarrow H + x$
  $t \leftarrow t + 1$
  $\text{LOGQ} \leftarrow \text{LOGQ} + x \log(\hat{q}) + (1-x)\log(1-\hat{q})$
  $\text{LOGP}(p) := H \log(p) + (t - H)\log(1 - p)$
  $I_p \leftarrow \{p | \text{LOGQ} - \text{LOGP}(p) \leq \log(1/\alpha)\}$
  $L \leftarrow \max\{L, \min I_p\}$
  $U \leftarrow \min\{U, \max I_p\}$
end loop

In the coin-betting example, $W_1, W_2, \ldots$ is a martingale where W_n represents our wealth after playing the game for n rounds. We stress that $W_t \geq 0$ for all $t > 0$. By convention, we will also have $W_1 = 1$. We further need a time-uniform generalization of Markov's inequality.

Proposition 2.10 (Ville's inequality Durrett (2010))**.** *Let $W_1, W_2, \ldots$ be a non-negative supermartingale. then for any real $a > 0$*

$$\mathrm{P}\left[\sup_{n \geq 1} W_n \geq a\right] \leq \frac{\mathbb{E}\left[W_1\right]}{a}.$$

Thus, whenever we play a game and earn a lot, we can — with high probability — rule out the possibility that the game is fair. So far, this is still an abstract framework. We have yet to design the betting game and the betting strategy and describe how to run the infinite number of games.

Betting game: Let[5] $0 < p < 1$. Consider a coin-betting game where we win $1/p$ (resp. $1/(1-p)$) multiple of the staked amount if we correctly predicted heads (resp. tails). If the underlying heads probability is p, then regardless of our bet - in expectation - we still have the same amount of money; thus, this game is fair. We identify heads and tails with outcomes $1, 0$ respectively.

Betting strategy: We deconstruct the betting strategy into the two sub-tasks: (1) If we know the underlying heads probability, we can design the optimal betting strategy for any criterion. (2) Estimate the heads probability. **First sub-task:** Let p define the betting game from the previous paragraph and q be the true heads probability; Optimally, bet q-fraction of wealth to heads and $1 - q$ fraction to tails. It maximizes the expected log-wealth, or equivalently, the expected growth-rate of our wealth and is also known as the Kelly Criterion. This is known to be optimal for the adaptive estimation task, see Wald (1947) under the name sequential-probability-ratio-test (SPRT). One might have expected the optimal criterion to optimize to be the expected wealth; however, the betting strategy maximizing the expected wealth suggest to bet all the money on one of the outcomes. This strategy, however, leads to an eventual bankruptcy almost surely and is not recommended in this context. **Second sub-task:** A natural choice is to use the running sample mean of the observations as the estimator of q. Unfortunately, this estimator would be either 0 or 1 after the first observations, so we go bankrupt whenever the observed sequence contains both outcomes. Thus, we use a "regularized" sample mean and after observing H times heads in a sequence of length t, we estimate $\hat{q} = (H + 0.5)/(t + 1)$. This is the MAP estimate of the mean with $\text{Beta}(1/2, 1/2)$ prior and is known as Krichevsky–Trofimov estimator (Cesa-Bianchi & Lugosi (2006) Section 9.7), Krichevsky & Trofimov (1981) and is proven to be successful for building confidence sequences beyond the Bernoulli case Orabona & Jun (2023).

Parallel betting games: We have described a betting game for a certain p and a betting strategy. Employing Ville's inequality we can possibly reject the hypothesis that the true sampling distribution

[5]For simplicity of exposition, similar arguments hold when $p \in \{0, 1\}$.

follows p. However, we need to run the game for all values of $p \in [0,1]$ and after every observation report the smallest interval containing all the values of p that were not rejected so far. This is clearly impossible to do explicitly, but it turns out that the non-rejected values of p form an interval and we can use binary search twice to find the end-points of the interval. This is a non-trivial result and generally does not need to hold for confidence intervals constructed by betting. The first key observations is that the betting strategy does not depend on p, so we can "play the game" just once. The second observation is that the resulting wealth is convex in p. To see why, let $\hat{q}_\tau$ (resp. x_τ) be our estimate of q (resp. the coin-toss outcome, for brevity $1/0$ corresponds to heads/tails respectively and $H = \sum_{\tau=1}^{t} x_\tau$), then our log-wealth at time t can be written as a function of p.

$$\begin{aligned}\log W_t(p) &= \log \prod_{\tau=1}^{t} \left(\left(\frac{\hat{q}_\tau}{p}\right)^{x_\tau} \left(\frac{1-\hat{q}_\tau}{1-p}\right)^{1-x_\tau} \right) \\ &= \underbrace{\sum_{\tau=1}^{t} x_\tau \log(\hat{q}_\tau) + (1-x_\tau)\log(1-\hat{q}_\tau)}_{\text{LOGQ}} \underbrace{-H\log(p) - (t-H)\log(1-p)}_{\text{LOGP}(p)}.\end{aligned}$$

Therefore, at every time-step, we can compute the interval of values of p for which the betting game has not concluded yet and thus form the current confidence interval. The proof is in Appendix F.

Theorem 2.11. *Algorithm 2 produces a valid confidence sequence at confidence level $1-\alpha$, where we interpret the interval $[L,U]$ at iteration T of the algorithm as the confidence interval at time t with width ε at most (which is attained in the worst case):*

$$\varepsilon_t \lesssim \sqrt{\frac{\log(1/\alpha) + \log(t)}{t}}.$$

This is not asymptotically optimal; still, empirically it performs well, and the same techniques can be used to obtain a confidence sequence that follows law-of-iterated-logarithms Orabona & Jun (2023). We present a comparison of the confidence sequences in Figure 2 and conclude this subsection by an implementation remark.

Remark 2.12. Wealth $W(\mathbf{x}_{:t})$ does not depend on the order of $\mathbf{x}_{:t}$, so we write it as $W(h,t)$, meaning wealth after observing h heads in the first t tosses. Now, for every time t, we can compute what is the minimal number of observed 1 (heads) (call it $H(t)$) so that p is outside of the lower-confidence interval. $H(t)$ is clearly non-decreasing in t; also, $W(h,t))$ can be easily computed from $W(h-1,t)$ and from $W(h,t-1)$ in constant time. The whole dynamic programming approach can be summarized in the following scheme which is repeatedly executed starting from $h=0, t=0$.

- If $W(h,t) \geq \frac{1}{\alpha}$: $H(t) := h$, $t := t+1$, compute $W(h,t+1)$
- Else: $h := h+1$, compute $W(h+1,t)$.

Both lines are executed in constant time. Also, we start from $W(0,0)$ and $h,t < N+1$. Executing a line, either h or t increases and thus to compute the thresholds for sequence of length up to N, we need to execute at most $2N$ lines of the scheme. We note that for the simplicity, we did not handle the case when $W(h,t) < 1/\alpha$. In that case we would just set $H(t) = t+1$ and increase t.

3 Experiments

Now we shall provide experimental evidence for the performance of the proposed methods. We benchmark the confidence sequences on the Sequential decision making task, where we try to certify a certain radius at given confidence level with as few samples as possible; the definition that follows is general beyond randomized smoothing. We emphasize that this setting of certifying a certain radius is by far the most common one in the robustness literature. We stress that the comparison of the robustness curves (e.g., as in Figure 1) is vacuous, since in the adaptive task, we do not spend samples to *improve* the robustness curves, beyond the certified level.

Definition 3.1 (Sequential decision making task)**.** Let $\frac{1}{2} \leq p, q \leq 1$ and only p being known. Receive samples from $\mathcal{B}(q)$. After every sample, either halt and declare that $p > q$, $p < q$, or request another sample. The task is to minimize the number of samples while being wrong with frequency at-most α.

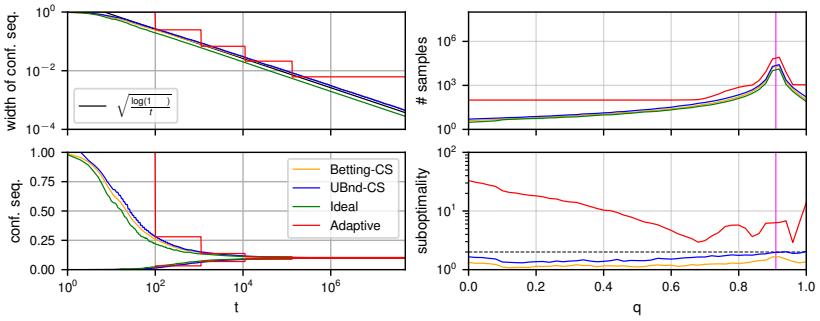


Figure 2: **left:** Comparison of widths of confidence sequences for the mean of Bernoulli $\mathcal{B}(0.1)$ with $\alpha = 0.001$. The width is on top and the actual confidence sequence on the bottom. In the notation of Algorithms 1 and 2, the sequence of $U - L$ is in the top figure, while both sequences U and L are in the bottom figure. Note the log-scale for t (and width on top). **right** : Instantiation of Task 3.1. The goal is to decide if $p = 0.91$ (vertical magenta line) or not with $\alpha = 0.001$. On top are the numbers of samples requested for the individual methods averaged over 1000 trials for 51 equally spaced values of $p \in [0, 1]$; on the bottom is the relative suboptimality of the individual methods; i.e., how many times more samples did they request compared to the ideal method. Note log scales on the $y-$axis. **methods:** UBnd-CS and Betting-CS are from Algorithm 1 and 2 respectively. Adaptive is from Horváth et al. (2022). The ideal is the unattainable lower-bound for the two tasks. On the LHS, it is a confidence interval on level $1 - \alpha$ computed independently at every time step. On the RHS, it is SPRT knowing both p, q which is optimal due to Wald (1947).

3.1 Related work

We identified Horváth et al. (2022) as the most relevant work. They distinguish between samples for which a predetermined radius r can be certified, and the samples for which it cannot. They use s values $n_1 < \cdots < n_s$ ($[10^2, 10^3, 10^4, 1.2 \cdot 10^5]$) sequentially as the number of samples. They try to certify radius r with n_1 samples; if it fails, then they try n_2 samples etc. They employ Bonferroni correction (union bound) and every sub-certification is allowed to fail with probability only $\frac{\alpha}{s}$. The key differences (details in Appendix C.2.1) to our method are that (1) It always abstains for hard tasks. (2) Splitting the α budget evenly degrades performance for small n. (3) method is only a heuristics. See Figure 5 and Tables 1, 3, 4. For the empirical comparison.

Another relevant work is Chen et al. (2022). Here, the certification is split in two phases. (1) Mean is crudely estimated. (2) The crude estimate selects the number of samples drawn so that the decrease (either multiplicative or absolute) in the certified radius is heuristically approximately at most a predetermined constant. We note that this heuristic for distributing samples can be made rigorous in a certain sense (see Dagum et al. (1995)). This is trivial for confidence sequences, as one can stop the estimation only as soon as they short enough and solve the task of Chen et al. (2022) with guarantees (instead of just heuristic). In this sense, we see our methods to be more general. We benchmark this in Table 2.

The works Seferis et al. (2023); Ugare et al. (2024) also address the speed issues of randomized smoothing, however, they are orthogonal to our directions. In particular, Ugare et al. (2024) uses an auxiliary network for which the certification is faster and transfer the certificates to the original model. Seferis et al. (2023) observes that few samples are sufficient for non-trivial certificates.

4 Conclusion

In this paper, we investigated the statistical estimation procedures related to randomized smoothing and improved them in the following two ways: (1) We have provided a strictly stronger version of confidence intervals than the Clopper-Pearson confidence interval. (2) We have developed confidence sequences for sequential estimation in the framework of randomized smoothing, which will greatly

	r=0.5	r=1.25 ,	r=2
Adaptive Horváth et al. (2022)	1976 ± 41	3593 ± 574	4623 ± 47
Betting CS 2	531 ± 157	2169 ± 257	2130 ± 339
Union bound CS 1	635 ± 157	2557 ± 234	2670 ± 271
Adaptive Horváth et al. (2022)	0.13 ± 0.006s	0.23 ± 0.036s	0.3 ± 0.003s
Betting CS 2	0.05 ± 0.012s	0.17 ± 0.019s	0.17 ± 0.02s
Union bound CS 1	0.05 ± 0.006s	0.19 ± 0.018s	0.21 ± 0.02s

Table 1: Comparison of the average number of samples (resp. time) needed to decide if a point is certifiably robust with given radius. Cifar10, ℓ_2, details are in Appendix C.2.1

ε	0.01	0.02	0.03
UB-CS	197 628	49 198	21 513
Betting-CS	199 771	47 215	20 918
Horvath	768 560	94 900	81 080

Table 2: We run the confidence sequences until the width is smaller than ε on a (both sided) confidence level 0.999. That way we can certify certain radius knowing that the true probability is at most ε larger. We used the same network as for the ℓ_2 experiment in Table 1 (WideResnet-40 on CIFAR10, $\sigma = 1$). We report the average number of samples required over 500 images.

reduce the number of samples needed for adaptive estimation tasks. Additionally, we provided matching algorithmic upper bounds with problem lower bounds for the relevant statistical estimation task.

5 Broader Impact Statement

We hope that this paper enlarges the interest in statistical estimation within the ML community.

Acknowledgments

The author was supported by the DFG Cluster of Excellence "Machine Learning – New Perspectives for Science", EXC 2064/1, project number 390727645 and is thankful for the support of Open Philanthropy.

References

Balsubramani, A. Sharp finite-time iterated-logarithm martingale concentration. *arXiv preprint arXiv:1405.2639*, 2014.

Biggio, B., Corona, I., Maiorca, D., Nelson, B., Šrndić, N., Laskov, P., Giacinto, G., and Roli, F. Evasion attacks against machine learning at test time. In *ECML PKDD*, 2013.

Bojchevski, A., Gasteiger, J., and Günnemann, S. Efficient robustness certificates for discrete data: Sparsity-aware randomized smoothing for graphs, images and more. In *ICML*, 2020.

Boucheron, S., Lugosi, G., and Massart, P. Concentration inequalities. 2013.

Cesa-Bianchi, N. and Lugosi, G. *Prediction, learning, and games*. Cambridge university press, 2006.

Chen, R., Li, J., Yan, J., Li, P., and Sheng, B. Input-specific robustness certification for randomized smoothing. In *AAAI*, 2022.

Clopper, C. J. and Pearson, E. S. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 1934.

Cohen, J., Rosenfeld, E., and Kolter, Z. Certified adversarial robustness via randomized smoothing. In *ICML*, 2019.

Dagum, P., Karp, R., Luby, M., and Ross, S. An optimal algorithm for monte carlo estimation. In *FOCS*, 1995.

Durrett, R. *Probability: Theory and Examples*. Cambridge Series in Statistical and Probabilistic Mathematics, 2010.

Goodfellow, I. J., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. *ICLR*, 2014.

Horváth, M. Z., Müller, M. N., Fischer, M., and Vechev, M. Boosting randomized smoothing with variance reduced classifiers. *ICLR*, 2022.

Howard, S. R., Ramdas, A., McAuliffe, J., and Sekhon, J. Time-uniform chernoff bounds via nonnegative supermartingales. *Probability Surveys*, 2020.

Krichevsky, R. and Trofimov, V. The performance of universal encoding. *IEEE Transactions on Information Theory*, 1981.

Kumari, A., Bhardwaj, D., Jindal, S., and Gupta, S. Trust, but verify: A survey of randomized smoothing techniques. *arXiv preprint arXiv:2312.12608*, 2023.

Lecuyer, M., Atlidakis, V., Geambasu, R., Hsu, D., and Jana, S. Certified robustness to adversarial examples with differential privacy. In *S&P*, 2019.

Ledoux, M. and Talagrand, M. *The Law of the Iterated Logarithm*, pp. 196–234. 1991.

Lee, G.-H., Yuan, Y., Chang, S., and Jaakkola, T. Tight certificates of adversarial robustness for randomly smoothed classifiers. *NeurIPS*, 2019.

Levine, A. and Feizi, S. (de) randomized smoothing for certifiable defense against patch attacks. *NeurIPS*, 2020.

Levine, A. J. and Feizi, S. Improved, deterministic smoothing for ℓ_1 certified robustness. In *ICML*, 2021.

Li, L., Xie, T., and Li, B. Sok: Certified robustness for deep neural networks. In *S&P*. IEEE, 2023.

Lugosi, G. and Mendelson, S. Mean estimation and regression under heavy-tailed distributions: A survey. *Foundations of Computational Mathematics*, 2019.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

Mnih, V., Szepesvári, C., and Audibert, J.-Y. Empirical bernstein stopping. In *Proceedings of the 25th international conference on Machine learning*, pp. 672–679, 2008.

Orabona, F. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

Orabona, F. and Jun, K.-S. Tight concentrations and confidence sequences from the regret of universal portfolio. *IEEE Transactions on Information Theory*, 2023.

Ryu, J. J. and Bhatt, A. On confidence sequences for bounded random processes via universal gambling strategies. *IEEE Transactions on Information Theory*, 2024.

Ryu, J. J. and Wornell, G. W. Gambling-based confidence sequences for bounded random vectors, 2024. URL `https://arxiv.org/abs/2402.03683`.

Salman, H., Li, J., Razenshteyn, I., Zhang, P., Zhang, H., Bubeck, S., and Yang, G. Provably robust deep learning via adversarially trained smoothed classifiers. In *NeurIPS*, 2019.

Seferis, E., Burton, S., and Kollias, S. Randomized smoothing (almost) in real time? In *ICMLw The Many Facets of Preference-Based Learning*, 2023.

Stevens, W. L. Fiducial limits of the parameter of a discontinuous distribution. *Biometrika*, 1950.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *ICLR*, 2014.

Ugare, S., Suresh, T., Banerjee, D., Singh, G., and Misailovic, S. Incremental randomized smoothing certification. In *ICLR*, 2024.

Voracek, V. and Hein, M. Improving l1-certified robustness via randomized smoothing by leveraging box constraints. In *International Conference on Machine Learning*, 2023.

Wald, A. *Sequential analysis*. 1947.

Yang, G., Duan, T., Hu, J. E., Salman, H., Razenshteyn, I., and Li, J. Randomized smoothing of all shapes and sizes. In *ICML*, 2020.

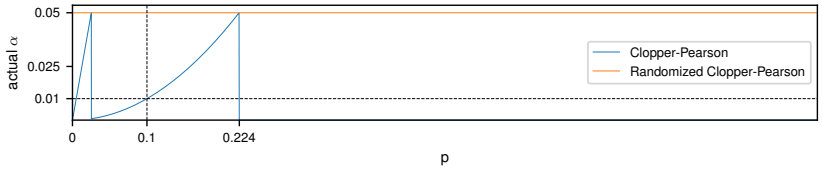


Figure 3: Actual coverages for (randomized) Clopper-Pearson confidence intervals for $\mathcal{B}(2, p)$.

A Deferred examples

Example A.1. [implicit type-2 error is exponentially small] Let us sample from $X \sim \mathcal{B}(p)$ to decide if the mean is smaller or larger than $p - \varepsilon$ using $n = 100\,000$ samples. We use Hoeffdings' inequality to bound the probability that p is incorrectly estimated to be lower than $p - \varepsilon$, i.e.,

$$\mathbb{P}\left[\frac{1}{n}\sum_{i=1}^{n} X_i \leq \mathbb{E}[X] - \varepsilon\right] \leq e^{-2n\varepsilon^2}.$$

Considering ε to be a constant, we see that this probability scales as e^{-n}. For example, when the true probability is 0.5 and we want to decide if it is smaller or larger than 0.4, already 1000 samples make the probability of incorrectly decision to be roughly $2 \cdot 10^{-9}$.

Example A.2. [suboptimality of Clopper-Pearson confidence interval and optimality of the randomized one] Recall that the coverage for p is the probability that p is included in the confidence interval when it is the true parameter and α is the allowed type-1 error and $1 - \alpha$ should be the coverage. Consider samples from $X \sim \mathcal{B}(2, p)$ and $\alpha = 0.05$. By definition, the Clopper-Pearson upper intervals are $[0, 1]$, $[0.025, 1]$, $[0.224, 1]$ for observations $x = 0, 1, 2$ respectively. Coverage for $p = 0.224$ is 0.95 because the event that p is outside of the confidence interval is $\mathbb{P}(X \in \{0, 1\}) = 1 - p^2 \approx 0.95$. On the other hand, coverage for $p = 0.1$ is $\mathbb{P}(X \in \{0, 1\}) = 1 - p^2 = 0.99$ and for all $p > 0.224$ it is 1; see Figure A for the coverages. Now we turn on to the randomized Clopper-Pearson interval for $p = 0.5$. Recall the definition of the upper interval,

$$u'_r(x, w) = \inf\{p \,|\, \mathbb{P}(\mathcal{B}(n, p) > x) + w\mathbb{P}(\mathcal{B}(n, p) = x) > \alpha\}.$$

The randomized confidence interval for some value x interpolates between the confidence intervals for x and $x + 1$ when $x < n$. Thus, when $x \neq 2$, $p > 0.224$ is always in the confidence interval (this happens with probability $1 - p^2 = 0.75$. Otherwise, we solve the following for w (because we set $n = 2$, $x = 2$):

$$\begin{aligned}\mathbb{P}(\mathcal{B}(2, p) > 2) + w\mathbb{P}(\mathcal{B}(2, p) = 2) &= \alpha,\\ 0 + p^2 w &= \alpha,\\ v &= \alpha/p^2.\end{aligned}$$

Thus, $p > 0.224$ is not contained in the randomized confidence interval iff $W \leq \alpha/p^2$ and $X = 2$. Since these random variables are independent, the resulting probability is $\mathbb{P}(W \leq 0.2)\mathbb{P}(X = 2) = \alpha/p^2 \cdot p^2 = \alpha$, as desired.

Example A.3 (Exponential increase of wealth for a biased coin)**.** For simplicity of exposition we assume that the coin falls on head 51 times from 100 tosses. To get a high probability statement is straightforward. Let us always bet 0.51 fraction of our money to to heads and 0.49 to tails (equivalently, just bet 0.02 of the money to the heads). If we win, we win 2% of our money, otherwise we lose 2%. Then, our wealth after 100 tosses will be $1.02^{51} \cdot 1.02^{-49} \sim 1.04$. Thus, every 100 tosses we multiply our wealth by the factor of 1.04 which is the desired exponential function.

B Binary or multiclass certification

Although all the standard benchmarking datasets for randomized smoothing are multiclass (cifar10 and imagenet), the commonly used randomized smoothing certification protocol is for the binary

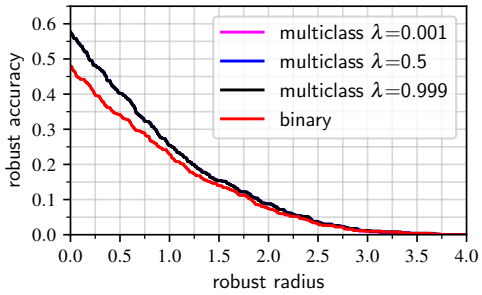


Figure 4: Comparison of the robustness curves for binary and multiclass certification. In the binary case, all the failure budget $\alpha = 0.001$ was spent on controlling the top-1 class probability. In the multiclass setting, we spend λ fraction of the budget in bounding p_A and the remaining $1 - \lambda$ part on bounding p_B. Note that this has no significant effect. The average certified radius for binary certification is 0.50, while for the multiclass it is 0.61. The experimental details are in Appendix C. The only difference is that now $\sigma = 1$.

setting, where we certify class A against all the other classes merged in a super class. In that case, the certification is done using the formula $r(\hat{p}_A, 1 - \hat{p}_A)$, where we have to guarantee that $p_A \geq \hat{p}_A$ at confidence level at least $1 - \alpha$. The alternative is to use multiclass certification; here, the certification is done via formula $r(\hat{p}_A, \hat{p}_B)$ ensuring that $p_A \geq \hat{p}_A$ and $p_B \leq \hat{p}_B$ at the same time at confidence level at least $1 - \alpha$. The difference between these two bounds naturally manifests in the regime when p_A is small. Strikingly, when $p_A < 0.5$, the binary certification approach cannot certify any robustness, while the multiclass one possible can. The cost for the multiclass procedure is only that we have to divide the failure budget between the the two estimation procedures. This is usually insignificant. In the ℓ_2 (and thus ℓ_∞ case), the role of α is rather minor, see Cohen et al. (2019) Figure 8. This is even more pronounced in the ℓ_1 case. Here α plays an absolutely negligible role in the resulting certified radius, see Voracek & Hein (2023), Subsection 2.7 for the discussion and Figures 4, 6. While this might be known to many, we believe that some readers may benefit from reading this argument. We demonstrate the difference in certification power in Figure 4. This multiclass certification fits in our setting effortlessly. We can run one confidence sequence for p_A, and another for p_B. We do not even need to know p_B and we can run it for all of them at the same time. This means, that we run it only for the second most observed class. This second most observed class does not need to be the actual runner-up class, but since it was possibly observed more times than the actual runner-up class, it will also provide a wider confidence interval, so the statistical estimation is still correct.

C Experimental details

C.1 Figure 1

The model in Figure 1 is the pretrained cifar10 model (Exactly the same model/setting from the example in README) of Salman et al. (2019), `https://github.com/Hadisalman/smoothing-adversarial`; in particular, it was ResNet-110 smoothed with Gaussian noise $\sigma = 0.12$ for ℓ_2 robustness. We set $\alpha = 0.001$ as usual and skip every 20 images of the test dataset (using 500 images, as is the standard practice).

C.2 Parameters of union bound confidence sequences

We were enlarging the sample size by a factor of $\beta = 1.1$ between estimations (that is, the condition $T = 2^K$ is replaced by $T > \beta^K = 1.1^K$ and our schedule is $\alpha_k = 5\alpha/((t+4)(t+5))$ The method is not sensitive to the choice of hyperparameters, see 5. In fact, the hyperparameters β, γ should be selected based on what is the "interesting" regime. There is an inherent tradeoff between a good asymptotical performance ($\beta, \gamma \sim 1$) and low-sample performance ($\beta, \gamma > 1$). This is confirmed in

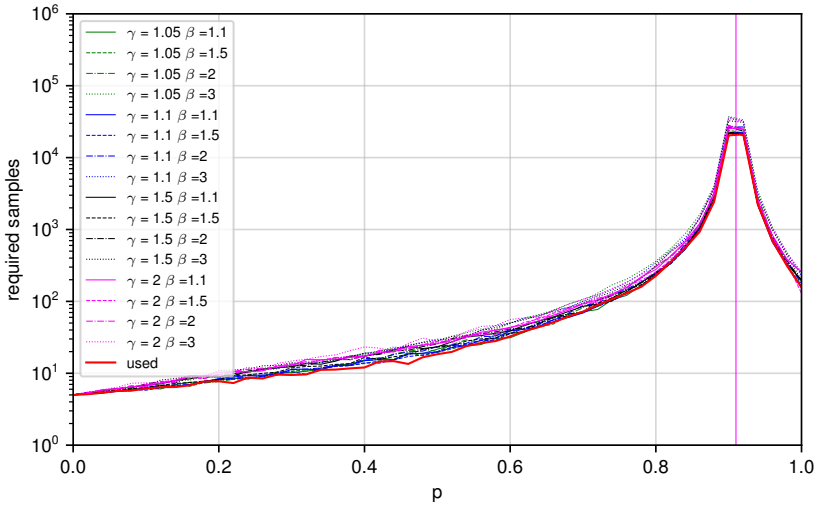


Figure 5: Samples needed for the adaptive estimation task as in Figure 2 for different hyperparameters. β is the factor by which we enlarge the sample size before computing new confidence interval, γ is the scaling of α as described in the main text. I.e., $k-$th estimation will have $\alpha_k = \frac{\alpha c}{k^\gamma}$ where c is the normalization constant such that $\sum_{k=1}^{\infty} \alpha_k = \alpha$.

Figure 5. The width of the confidence interval scales as:

$$\sqrt{\frac{\beta(\log(1/(\gamma-1)) + \log(1/\alpha) + \gamma \log\log_\beta t)}{t}}.$$

This is because $\sum_{t=1}^{\infty} \frac{1}{t^\gamma} \asymp \frac{1}{\gamma-1}$, $\alpha_k \asymp \frac{\alpha(\gamma-1)}{k^\gamma}$ and $t \asymp \beta^k$, then $k \asymp \log_\beta t$. Plugging in these identites in $\sqrt{\frac{\log(\alpha_t)}{t}}$, and remembering that the confidence interval is recomputed only after elarging the sample size by β factor, then for time t, the actual t we use in the formula can be as small as t/β.

C.2.1 Comparison with Horváth et al. (2022)

The method from Horváth et al. (2022) uses a finite collection of values of $n = n_1 < \cdots < n_s$ for which the confidence intervals are computed. A direct consequence is that the hard examples for which more than n_s samples are needed cannot be certified. Additionally, due to Bonferroni correction, with large s, the term $\log(s/\alpha)$ appearing in the confidence interval becomes large compared to t for small values of t (compare with the polynomial scaling that we propose where this is not the case). Consider $n_i = n_1^i$, then the width of the confidence interval scales as (not considering the regime when $t > n_s$ where the width is constant):

$$\sqrt{\frac{n_1(\log s + \log 1/\alpha)}{t}}.$$

In particular, when we want to have confidence sequence up $n_s = N$ samples, then the width becomes

$$\sqrt{\frac{\sqrt[s]{N}(\log s + \log 1/\alpha)}{t}}$$

and we either pay for the fact that we have large differences between the steps ($\sqrt{s}N$), or for the fact that we have lot of steps ($\log(s)$) which is detrimental for small values of t.

C.3 Details for 1, 3, 4

We had WideResNet-40-2 for CIFAR-10 trained for 120 epochs with SGD and learning rate 0.1, Nesterov momentum 0.9, weight decay 0.0001 and cosine annealing. batch size 64. The loss was the standard, noise-augmented training using the same noise as for the certification. We either used Gaussian smoothing for ℓ_2 robustness of uniform in ℓ_∞ box for ℓ_1 robustness.

For the certification we used batch size 100 (natural for Horváth et al. (2022)). For our method, we had a mixed batch of data points so the data points for which we have the least amount of samples and are not decided yet are put in the batch.

D Proof of Proposition 2.4

Proof. We show it for the upper interval, the lower is analogical. First, we note that $f(p) = \mathbb{P}(\mathcal{B}(n,p) \geq a)$ is non-decreasing in p for any n, a; Thus, $u_r(X) \leq p$ if and only if $\mathbb{P}(\mathcal{B}(n,p) > x) + w\mathbb{P}(\mathcal{B}(n,p) = x) > \alpha$. Now, we show that $\mathbb{P}(u_r(X) \leq p) = 1 - \alpha$ for $X \sim \mathcal{B}(n,p)$. To shorten the notation, let $\alpha_1 = \mathbb{P}(X \geq a)$ and $\alpha_2 = \mathbb{P}(X > a)$ for such a that $\alpha_1 \leq \alpha \leq \alpha_2$. We have

$$\begin{aligned}\mathbb{P}(u_r(X) \leq p) = \mathbb{P}(u_r(X) \leq p \mid X < a)\mathbb{P}(X < a) \\ + \mathbb{P}(u_r(X) \leq p \mid X = a)\mathbb{P}(X = a) \\ + \mathbb{P}(u_r(X) \leq p \mid X > a)\mathbb{P}(X > a),\end{aligned}$$

which we evaluate to $\mathbb{P}(u_r(X) \leq p) = (1 - \alpha_1) + (\alpha_1 - \alpha_2)\mathbb{P}(u_r(X) \leq p \mid X = a) + 0$. We note that given the event $X = a$, it holds $u_r(X) \leq p \iff \alpha_2 + W(\alpha_1 - \alpha_2) > \alpha$, so $\mathbb{P}(u_r(X) \leq p \mid X = a) = \mathbb{P}(\alpha_2 + W(\alpha_1 - \alpha_2) > \alpha) = \mathbb{P}\left(W > \frac{\alpha - \alpha_2}{\alpha_1 - \alpha_2}\right) = \frac{\alpha_1 - \alpha}{\alpha_1 - \alpha_2}$. Overall, $\mathbb{P}(u_r(X) \leq p) = (1 - \alpha_1) + (\alpha_1 - \alpha_2)\frac{\alpha_1 - \alpha}{\alpha_1 - \alpha_2} = 1 - \alpha$.

The second part of the statement follows from Neymann-Pearson lemma. Concretely, we consider the following binary hypothesis testing problem from sample $\mathcal{B}(n,\theta)$, and we decide if $\theta = p$ or $\theta = q$. Both confidence intervals has size α and can be interpreted as binary tests – just return the indicator function of $q \in I(x)$. Neymann-Pearson lemma states that the (unique) uniformly most powerful test is the likelihood ratio test, which is implemented by the randomized Clopper-Pearson interval.

□

E Proof of Theorem 2.7

Proof of Theorem 2.7. First, $\sum_{i=1}^{\infty} \alpha_t = \sum_{k=1}^{\infty} \frac{\alpha}{k(k+1)} = \alpha$; thus, by union bound, all the computed confidence intervals are simultaneously correct at confidence level $1 - \alpha$. Next we show that the width is as claimed. When $t = 2^k$, we directly have from Hoeffding's inequality

$$\varepsilon \lesssim \sqrt{\frac{\log \frac{1}{\alpha_t}}{t}} \asymp \sqrt{\frac{\log \frac{1}{\alpha} + \log\log t}{t}}.$$

Otherwise, we would use a confidence interval of some previous t' such that $t' < t < 2t'$ with width

$$\varepsilon \lesssim \sqrt{\frac{\log \frac{1}{\alpha} + \log\log t'}{t'}} \asymp \sqrt{\frac{\log \frac{1}{\alpha} + \log\log t}{t}},$$

Noting that Clopper-Pearson's confidence intreval is shorter than Hoeffding's finishes the proof. □

F Proof of Theorem 2.11

Proof. First we verify that everything in the algorithm is well defined and the logarithms take positive inputs. Now let $W_t = \exp\left(\frac{\text{LOGQ}_t}{\text{LOGP}_t}\right)$ where subscript t denotes iteration of the algorithm. We show that it is a martingale when $X \sim \mathcal{B}(p)$ for $0 < p < 1$. In that case,

$$\mathbb{E}_X[W_t] = W_{t-1}\mathbb{E}_X\left[\left(\frac{\hat{q}_t}{p}\right)^X \left(\frac{1 - \hat{q}_t}{1 - p}\right)^{1-X}\right] = W_{t-1}\left(p\frac{\hat{q}}{p} + (1-p)\frac{1-\hat{q}}{1-p}\right) = W_{t-1}.$$

	r=0.5	r=1.25 ,	r=2
Adaptive Horváth et al. (2022)	1976 ± 41	3593 ± 574	4623 ± 47
Betting CS 2	531 ± 157	2169 ± 257	2130 ± 339
Union bound CS 1	635 ± 157	2557 ± 234	2670 ± 271
Adaptive Horváth et al. (2022)	0.13 ± 0.006s	0.23 ± 0.036s	0.3 ± 0.003s
Betting CS 2	0.05 ± 0.012s	0.17 ± 0.019s	0.17 ± 0.02s
Union bound CS 1	0.05 ± 0.006s	0.19 ± 0.018s	0.21 ± 0.02s
	r=0.5	r=1.25 ,	r=2
Adaptive Horváth et al. (2022)	4150 ± 523	2266 ± 640	4760 ± 131
Betting CS 2	2206 ± 150	1665 ± 84	3932 ± 199
Union bound CS 1	2665 ± 78	1674 ± 37	3717 ± 361
Adaptive Horváth et al. (2022)	0.27 ± 0.04s	0.15 ± 0.04s	0.3 ± 0.009s
Betting CS 2	0.17 ± 0.011s	0.13 ± 0.006s	0.3 ± 0.014s
Union bound CS 1	0.2 ± 0.005s	0.13 ± 0.002s	0.28 ± 0.02s

Table 3: Comparison of the average number of sample needed to decide if the point is certifiably robust with given radius in the top. The time needed is on the bottom. The upper table was in the main paper, the bottom one is the exact same experiment but with a retrained model for ℓ_2 robustness robustness with $\sigma = 1$.

	r=0.5	r=1 ,	r=1.5
Adaptive Horváth et al. (2022)	423 ± 38	3520 ± 82	3823 ± 127
Betting CS 2	138 ± 28	2680 ± 221	3007 ± 122
Union bound CS 1	150 ± 3	2926 ± 18	3144 ± 347
Adaptive Horváth et al. (2022)	0.04 ± 0.006s	0.23 ± 0.006s	0.25 ± 0.009s
Betting CS 2	0.19 ± 0.002s	0.21 ± 0.017s	0.23 ± 0.01s
Union bound CS 1	0.016 ± 0.006s	0.22 ± 0.009s	0.25 ± 0.03s
	r=0.5	r=1 ,	r=1.5
Adaptive Horváth et al. (2022)	1370 ± 596	4790 ± 50	8463 ± 714
Betting CS 2	592 ± 94	4055 ± 459	5822 ± 81
Union bound CS 1	806 ± 113	4327 ± 106	5795 ± 321
Adaptive Horváth et al. (2022)	0.1 ± 0.04s	0.30 ± 0.003s	0.53 ± 0.05s
Betting CS 2	0.05 ± 0.007s	0.31 ± 0.03s	0.44 ± 0.005s
Union bound CS 1	0.06 ± 0.008s	0.33 ± 0.008s	0.44 ± 0.02s

Table 4: Comparison of the average number of samples needed to decide if the point is certifiably robust with given radius in the top. The time needed is on the bottom. top and bottom are again the exact same experiment but with a retrained model for ℓ_1 robustness with $\sigma = 1$.

If $p \in \{0, 1\}$, then we would have a deterministic sequence and $W_t \leq W_{t-1}$. Thus, W_t is a supermartingale. It is also output of the exponential function and so is non-negative. Therefore, the assumptions of Ville's inequality are satisfied and can be applied. Whenever p is excluded from the confidence interval, it happened that $\text{LOGQ}_t - \text{LOGP}_t \geq \log(1/\alpha)$, or equivalently, $W_t \geq 1/\alpha$ which can only happen with probability α and it is thus a valid confidence sequence. We also recall that in the main text we have shown that I_p is a sub-level set of a convex function and is thus convex and can be efficiently found by binary search. The width of the confidence interval follows from the standard regret bounds for the Krichevsky–Trofimov estimator Cesa-Bianchi & Lugosi (2006) Section 9.7; Krichevsky & Trofimov (1981). The result then follows from Orabona (2019), Subsection 12.7. It was also derived in Ryu & Bhatt (2024); see Ryu & Wornell (2024) for generalization to vector-valued random variables.

□