Provable Adversarial Robustness for Fractional ℓ_p Threat Models

Alexander Levine
University of Maryland

Soheil Feizi
University of Maryland

Abstract

In recent years, researchers have extensively studied adversarial robustness in a variety of threat models, including ℓ_0, ℓ_1, ℓ_2, and ℓ_∞-norm bounded adversarial attacks. However, attacks bounded by fractional ℓ_p "norms" (quasi-norms defined by the ℓ_p distance with $0 < p < 1$) have yet to be thoroughly considered. We proactively propose a defense with several desirable properties: it provides provable (certified) robustness, scales to ImageNet, and yields deterministic (rather than high-probability) certified guarantees when applied to quantized data (e.g., images). Our technique for fractional ℓ_p robustness constructs expressive, deep classifiers that are globally Lipschitz with respect to the ℓ_p^p metric, for any $0 < p < 1$. However, our method is even more general: we can construct classifiers which are globally Lipschitz with respect to any metric defined as the sum of concave functions of components. Our approach builds on a recent work, Levine and Feizi (2021), which provides a provable defense against ℓ_1 attacks. However, we demonstrate that our proposed guarantees are highly non-vacuous, compared to the trivial solution of using (Levine and Feizi, 2021) directly and applying norm inequalities.

1 Introduction

Adversarial attacks (Szegedy et al., 2013; Goodfellow et al., 2014; Athalye et al., 2018; Carlini and Wagner, 2017; Tramèr et al., 2017) represent a significant security vulnerability in deep learning. In these attacks, small (often imperceptible) perturbations of the input to a machine-learning system (such as a classifier) are made which change the behavior of the system in an undesirable way. Concretely, for example, a small change to an image belonging to one class (e.g., an image of a cat) can be crafted in order to cause a classifier to misclassify the image as belonging to a different class (e.g., the class 'dogs').

One line of work to ameliorate this threat has been to propose certifiably (provably) robust classifiers, where each classification is paired with a certificate, specifying a radius in input space (with respect to some distance function) around the input in which the classification is guaranteed to be constant. Of these certification techniques, in general, *randomized smoothing* approaches (Cohen et al. (2019), among others) have shown to be uniquely promising for large-scale tasks in the scale of ImageNet. However, these techniques also have drawbacks: they provide only probabilistic, rather than deterministic, certificate results, and rely on Monte-Carlo sampling at test time, requiring a large number of evaluations of the "base classifier" neural network.

Recently, Levine and Feizi (2021) proposed a randomized smoothing-inspired technique for certifying robustness against the ℓ_1 threat model, which provides deterministic certificates at ImageNet scale. That work demonstrates that the averaged output of a bounded function (i.e., a classifier logit) over a specially-designed *finite, tractable* set of noise samples must be Lipschitz with respect to the ℓ_1-norm. Because the averaged logits are Lipschitz, a robustness certificate can be computed simply by dividing the difference between the top logit and the runner-up logit by the Lipschitz constant, and dividing by 2 (this gives the minimum radius required for the runner-up logit to overtake the top logit, and hence to change the classification.)

In this work, we extend the results of Levine and Feizi (2021) to cover ℓ_p "norms" for $p < 1$. More precisely, we develop a deterministic smoothing method that guarantees Lipschitzness with respect to the ℓ_p^p

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

metric for $p \in (0, 1)$, defined as:

$$\ell_p^p(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{d} |x_i - y_i|^p \tag{1}$$

This immediately provides ℓ_p^p-metric certificates, which can be converted to ℓ_p certificates by simply raising the radius to the power of $1/p$. Our technique is in fact more general than this, and can be applied to ensure the Lipschitz continuity of a function to a larger family of "elementwise-concave metrics" defined as the sum of concave functions of coordinate differences.

While not as frequently encountered as other ℓ_p norms, ℓ_p "norms" for $p \in (0, 1)$ (which are in fact quasi-norms, because they violate the triangle inequality) are used in several machine-learning applications (see Related Works, below). While ℓ_p, $p \in (0, 1)$ adversarial attacks have yet to emerge in practice, Wang et al. (2021) have recently proposed an algorithm for ℓ_p-constrained optimization with $p < 1$: the authors mention that this could be used to generate adversarial examples. This suggests that developing defenses to such attacks is a valuable exercise. Fractional ℓ_p threat models can also be thought of as "soft" versions of the widely-considered ℓ_0 threat model, allowing the attacker, in addition to entirely changing some pixels, to slightly impact additional pixels at a "discount", without paying the full price in perturbation budget for modifying them. This may be relevant, for example, in physical ℓ_0 attacks. Furthermore, readers may find other uses for ensuring that a trained function is ℓ_p^p-Lipschitz for $p < 1$.

Our technique inherits some of the limitations of Levine and Feizi (2021): notably that the deterministic variant applies exclusively to bounded, quantized input domains: that is, inputs where the value in each dimension only assumes values in $[0, 1]$ which are multiples of $1/q$, for some quantization parameter q. However, this applies to many domains of practical interest in machine learning, such as image classification, which typically uses $q = 255$. Because the image domain is perhaps the most widely-studied domain of adversarial robustness, this restriction does not pose a significant limitation in practice. (Even in their randomized variants, both Levine and Feizi (2021) and this work assume bounded input domains: that is, inputs $\boldsymbol{x} \in [0, 1]^d$.)

In Appendix D, we also consider the $p = 0$ limit of our algorithm. In that case, we show that our method simplifies to essentially a deterministic variant of the "randomized ablation" ℓ_0 smoothing defense proposed by Levine and Feizi (2020a). In fact, this deterministic variant was already implicitly discussed in Levine and Feizi (2020b), where a specialized form of it was used to provably defend against poisoning attacks. Here, we apply it to evasion attacks directly. While this simplified ℓ_0 defense somewhat under-performs the randomized variant, it provides deterministic certificate results at greatly reduced runtime.

In summary, in this work, we propose a novel, deterministic method for ensuring that a trained function on bounded, quantized inputs is Lipschitz with respect to any ℓ_p^p metric for $p \in (0, 1)$. This has immediate applications to provable adversarial robustness: we use our method to generate robustness certificates for fractional ℓ_p quasi-norms on CIFAR-10 and ImageNet.

2 Related Works

Many prior works have proposed techniques for certifiable robust classification, under various ℓ_p norms, This includes many techniques that provide deterministic certification results, for small-scale image classification tasks. (Wong and Kolter, 2018; Gowal et al., 2018; Raghunathan et al., 2018; Tjeng et al., 2019; Zhang et al., 2018; Li et al., 2019a; Anil et al., 2019; Jordan et al., 2019; Singla and Feizi, 2021, 2020; Trockman and Kolter, 2021). As mentioned in the introduction, randomized smoothing approaches (Salman et al., 2019; Cohen et al., 2019; Lecuyer et al., 2019; Li et al., 2019b; Lee et al., 2019; Yang et al., 2020; Zhai et al., 2019; Jeong and Shin, 2020) are the only certification approaches that are practical at ImageNet scale. However, in general, these techniques do not produce deterministic certificates: for ℓ_p norms, the only ImageNet-scale deterministic certification result is Levine and Feizi (2021).

Some known applications of ℓ_p, $(p < 1)$ "norms" in machine learning include clustering (Aggarwal et al., 2001; Datar et al., 2004), dimensionality reduction (Chachlakis and Markopoulos, 2021), and image retrieval (Howarth and Rüger, 2005).

3 Notation and Preliminaries

We first specify some notation. Let $\mathcal{U}(a, b)$ represent the uniform distribution on the range $[a, b]$, and let $\text{Beta}(\alpha, \beta)$ represent the beta distribution with parameters α, β. Let $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ represent the floor and ceiling functions. Let $[d]$ be the set $1, ..., d$. Let $\mathbf{1}_{(\text{condition})}$ be the indicator function. Following Levine and Feizi (2021), we use '$a \pmod b$' with real-valued a, b to indicate $a - b\lfloor a/b \rfloor$.

Next, we define the general set of metrics our technique applies to, of which ℓ_p^p metrics are an example.

Definition 1 (Elementwise-concave metric (ECM))**.** *For any $\boldsymbol{x}, \boldsymbol{y}$, let $\delta_i := |x_i - y_i|$. An elementwise-*

concave metric (ECM) is a metric on $[0,1]^d$ *in the form:*

$$d(\boldsymbol{x},\boldsymbol{y}) := \sum_{i=1}^{d} g_i(\delta_i), \tag{2}$$

where $g_1, ..., g_d \subset [0,1] \to [0,1]$ *are increasing, concave functions with* $g_i(0) = 0$.

Note that the ℓ_p^p metrics with $p \leq 1$ are ECM's, with $\forall i,\ g_i(z) = z^p$. Note also that any distance function meeting the definition of an ECM is in fact a metric, unless some g_i is the zero function.

We also introduce the main theorem from Levine and Feizi (2021), which our work extends upon.

Theorem 1 (Levine and Feizi (2021))**.** *For any* $f : [0,1]^d \times [0,1]^d \to [0,1]$, *and* $\Lambda > 0$, *let* $\boldsymbol{s} \in [0,\Lambda]^d$ *be a random variable, with a fixed distribution such that:*

$$s_i \sim \mathcal{U}(0,\Lambda), \quad \forall i. \tag{3}$$

Note that the components $s_1, ..., s_d$ *are* ***not*** *required to be distributed independently from each other. Then, define:*

$$x_i^{upper} := \min(\Lambda\lceil\frac{x_i - s_i}{\Lambda}\rceil + s_i, 1)\ , \quad \forall i \tag{4}$$

$$x_i^{lower} := \max(\Lambda\lceil\frac{x_i - s_i}{\Lambda}\rceil + s_i - \Lambda, 0)\ , \quad \forall i \tag{5}$$

$$p(\boldsymbol{x}) := \mathop{\mathbb{E}}_{\boldsymbol{s}}\left[f(\boldsymbol{x}^{lower}, \boldsymbol{x}^{upper})\right]. \tag{6}$$

Then, $p(.)$ *is* $1/\Lambda$*-Lipschitz with respect to the* ℓ_1 *norm.*

We provide a visual explanation of this theorem in Figure 1. The basic intuition is that the $[0,1]$ domain of each dimension is divided into "bins", with dividers at each value $s_i + n\Lambda, \forall n \in \mathbb{N}$. Then, x_i^{lower} and x_i^{upper} are the lower- and upper-limits of the bin which x_i is assigned to. For two points $\boldsymbol{x}$ and $\boldsymbol{y}$, let $\delta_i := |x_i - y_i|$: the probability of a divider separating x_i and y_i is $\min(\delta_i/\Lambda, 1)$. this means that the probability that $(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}})$ differs from $(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}})$ is at most $\|x - y\|_1/\Lambda$. The Lipschitz property follows from this.

Note that we have modified the notation from the original statement of the theorem: in particular, we use Λ instead of 2λ. Additionally, we pass both $\boldsymbol{x}^{\text{lower}}$ and $\boldsymbol{x}^{\text{upper}}$ to the base classifier f, even though these are redundant when Λ is fixed: this is because we are about to break this assumption. (We include a proof sketch in the modified notation in Appendix A.1.) Note also that this is a *randomized* algorithm; we will discuss the derandomization in Section 5, where we present the derandomization of our proposed method.

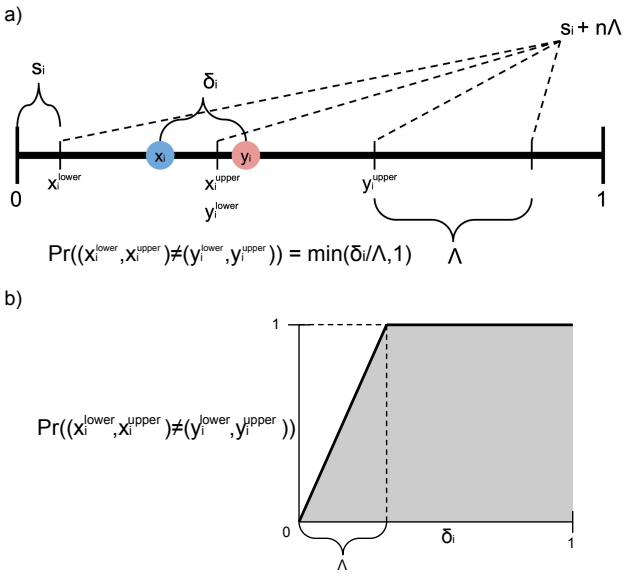

Figure 1: A visual explanation of Theorem 1 from Levine and Feizi (2021). (a) Whether x_i and y_i belong to the same bin depends on the value of the bin-divider offset s_i. However, because this is uniformly distributed, the probability that they are mapped to different bins is simply δ_i/Λ, if $\delta_i < \Lambda$, and 1 otherwise. (b) Graph of the probability that x_i and y_i are assigned to different bins, as a function of their difference δ_i.

4 Proposed Method

In this paper, we modify the algorithm described in Theorem 1 by allowing Λ itself to vary randomly in each dimension, according to a fixed distribution $\mathcal{D}_i$:

$$\begin{aligned} \Lambda_i &\sim \mathcal{D}_i \\ s_i &\sim \mathcal{U}(0, \Lambda_i) \end{aligned} \tag{7}$$

The reason for doing this is that, by mixing the smoothing distributions for various Λ in each dimension, the probability that x_i and y_i are assigned to different "bins" assumes a concave relationship to their difference δ_i, as illustrated in Figure 2. In fact, by doing this, we are able to make the probability of splitting x_i and y_i to be any arbitrary smooth concave increasing function of δ_i, allowing us to enforce Lipschitzness with respect to arbitrary ECMs, as shown in the upcoming Theorem 2.

Note that, if we allow the support of $\mathcal{D}_i$ to be $(0,\infty)$, there is some redundancy in the noise model specified by Equation 7: in particular, whenever $\Lambda_i > 1$, there are at most two bins, with the single divider s_i uniformly on the range $[0,1]$ with probability $1/\Lambda_i$ and otherwise with the entire range $[0,1]$ falling into one bin. For simplicity, therefore, we can allow the support of $\mathcal{D}_i$ to be $(0,1] \cup \{\infty\}$, where $\Lambda_i = \infty$ signifies to consider the entire domain as one bin. Formally, our noise process is defined as follows:

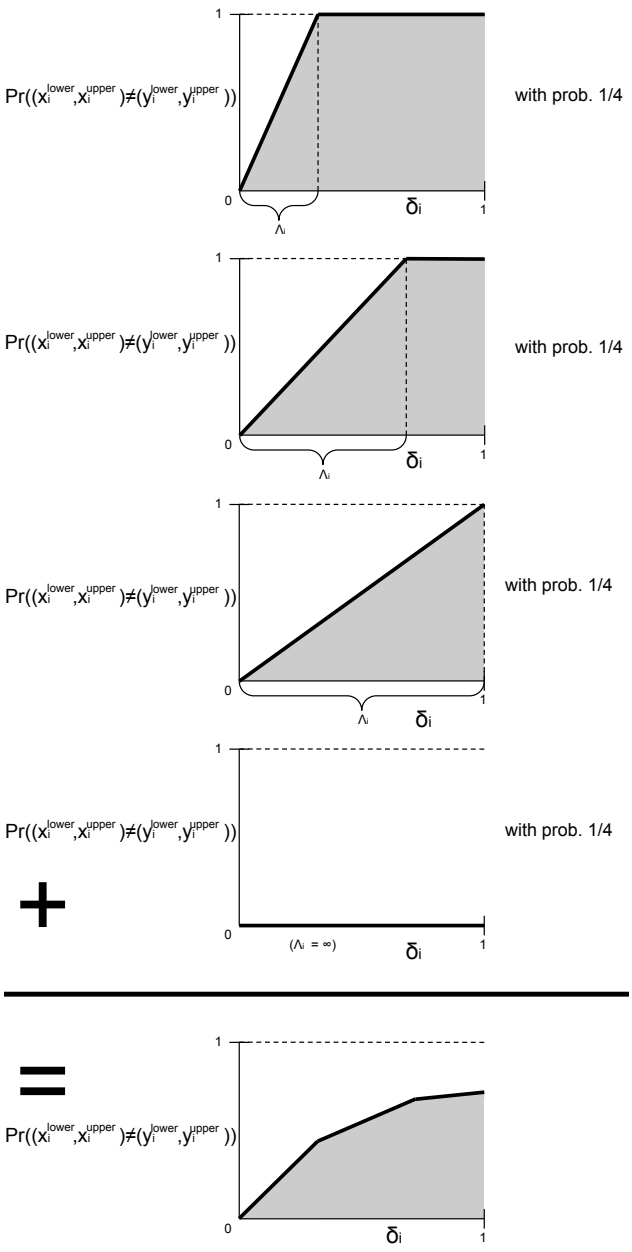

Figure 2: A mixture of various values of Λ creates a concave relationship between the probability that x_i and y_i are distinguishable to the base classifier and the difference δ_i between their values. This is because the slope of each of the curves in the mixture goes to zero at $\delta_i = \Lambda_i$.

Definition 2 (Variable-Λ smoothing)**.** *For any $f : [0,1]^d \times [0,1]^d \to [0,1]$, and distribution $\mathcal{D} = \{\mathcal{D}_1, ...\mathcal{D}_d\}$, such that each $\mathcal{D}_i$ has support $(0,1] \cup \{\infty\}$, let:*

$$\Lambda_i \sim \mathcal{D}_i \tag{8}$$

If $\Lambda_i = \infty$, then $x_i^{upper} := 1$, $x_i^{lower} := 0$, otherwise:

$$s_i \sim \mathcal{U}(0, \Lambda_i) \tag{9}$$

$$x_i^{upper} := \min(\Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i, 1) \tag{10}$$

$$x_i^{lower} := \max(\Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i - \Lambda_i, 0) \tag{11}$$

$$\tag{12}$$

The smoothed function is defined as:

$$p_{\mathcal{D},f}(\boldsymbol{x}) := \mathop{\mathbb{E}}_{\boldsymbol{s}} \left[f(\boldsymbol{x}^{lower}, \boldsymbol{x}^{upper}) \right]. \tag{13}$$

Note that we make no assumptions about the joint distributions of Λ or of $\boldsymbol{s}$.

We can now present our main theorem, describing how to ensure Lipschitzness with respect to an ECM:

Theorem 2. *Let $d(\cdot,\cdot)$ be an ECM defined by concave functions $g_1, ..., g_d$. Let $\mathcal{D}$ and $f(\cdot)$ be the Λ-distribution and base function used for Variable-Λ smoothing, respectively. Let $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^d$ be two points. For each dimension i, let $\delta_i := |x_i - y_i|$. Then:*

(a) The probability that $(x_i^{lower}, x_i^{upper}) \neq (y_i^{lower}, y_i^{upper})$ is given by $\Pr_i^{split}(\delta_i)$, where:

$$\Pr_i^{split}(z) := \Pr_{\mathcal{D}_i}(\Lambda_i \leq z) + z \mathop{\mathbb{E}}_{\mathcal{D}_i} \left[\frac{\mathbf{1}_{(\Lambda_i \in (z,1])}}{\Lambda_i} \right] \tag{14}$$

(b) If $\forall i \in [d]$ and $\forall z \in [0,1]$,

$$\Pr_i^{split}(z) \leq g_i(z), \tag{15}$$

then, the smoothed function $p_{\mathcal{D},f}(\cdot)$ is 1-Lipschitz with respect to the metric $d(\cdot,\cdot)$.

(c) Suppose g_i is continuous and twice-differentiable on the interval $(0,1]$. Let $\mathcal{D}_i$ be constructed as follows:

- *On the interval $(0,1)$, Λ_i is distributed continuously, with pdf function:*

$$pdf_{\Lambda_i}(z) = -z g_i''(z) \tag{16}$$

- $\Pr(\Lambda_i = 1) = g_i'(1)$
- $\Pr(\Lambda_i = \infty) = 1 - g_i(1)$

then,

$$\Pr_i^{split}(z) = g_i(z) \quad \forall z \in [0,1]. \tag{17}$$

If all $\mathcal{D}_i$ are constructed this way, then the conclusion of part (b) above applies.

Here, $\Pr_i^{\text{split}}(\delta_i)$ represents the probability that the base classifier is given the information necessary to distinguish x_i from y_i; in order to ensure that the base classifier receives as much information as possible, we would like to design $\mathcal{D}_i$ to make $\Pr_i^{\text{split}}(z)$ as large as possible, for all $z \in [0,1]$. However, if we want our smoothed classifier to have the desired Lipschitz property, $\Pr_i^{\text{split}}(z)$ can be no larger than $g_i(z)$, as stated in part (b) of the theorem. Part (c) of the theorem shows how to design $\mathcal{D}_i$ such that $\Pr_i^{\text{split}}(z)$ takes exactly its maximum allowed value, $g_i(z)$, everywhere.

We can apply part (c) of Theorem 2 to derive smoothing distributions for Lipschitzness on fractional-p ℓ_p^p metrics, simply by taking $g_\cdot(z) = \frac{z^p}{\alpha}$:

Corollary 1. *For all $p \in (0,1]$, $\alpha \in [1,\infty)$, if we perform Variable-Λ smoothing with all Λ_i's distributed identically (but not necessarily independently) as follows:*

$$\begin{aligned} \Lambda_i \sim & Beta(p,1), \ \textit{with prob.} \ \frac{1-p}{\alpha} \\ \Lambda_i = & 1, \ \textit{with prob.} \ \frac{p}{\alpha} \\ \Lambda_i = & \infty, \ \textit{with prob.} \ 1 - \frac{1}{\alpha} \end{aligned} \tag{18}$$

then, the resulting smoothed function will be $1/\alpha$-Lipschitz with respect to the ℓ_p^p metric[1].

We use the fact that 1-Lipschitzness with respect to $d(\cdot,\cdot)/\alpha$ is equivalent to $1/\alpha$-Lipschitzness with respect to $d(\cdot,\cdot)$. We can verify that taking $p=1$, $\alpha = \Lambda$ recovers Theorem 1 for $\Lambda \geq 1$.

5 Quantization and Derandomization

While the previous section describes a *randomized smoothing* scheme for guaranteeing ℓ_p^p-Lipschitz behavior of a function, in this section, we would like to derandomize this algorithm to ensure an exact, rather than high-probability, guarantee. For the fixed-Λ case, Levine and Feizi (2021) derives such a derandomization in a two-step argument. First, a *quantized* form of Theorem 1 is proposed. To explain this, we introduce some notation from Levine and Feizi (2021). Let q be the number of quantizations (e.g., 255 for images). Let

$$[a,b]_{(q)} := \left\{ i/q \;\middle|\; \lceil aq \rceil \leq i \leq \lfloor bq \rfloor \right\}. \tag{19}$$

[1]If we desire *weaker* Lipschitz guarantees, i.e., with $\alpha < 1$, the assumptions of Theorem 2-c no longer hold. We deal with this case in Appendix A.3.1, but it is not particularly relevant for our application: note that, as long as the classifier's accuracy remains high, then certificates scale with $1/\alpha$, so larger α is generally desirable. In our experiments, we find that the classifier's accuracy remains high even for α much greater than 1.

For example, $[0,1]_{(q)}$ represents the set $\{0, \frac{1}{q}, \frac{2}{q}, ..., \frac{q-1}{q}, 1\}$. Departing slightly from Levine and Feizi (2021), we define $\mathcal{U}_{(q)}(a,b)$ as the uniform distribution on the set $[a, b-\frac{1}{q}]_{(q)} + \frac{1}{2q}$. (e.g., $\mathcal{U}_{(q)}(0,1)$ is uniform on $\{\frac{1}{2q}, \frac{3}{2q}, ... \frac{2q-1}{2q}\}$: these are the *midpoints between* the quanizations in $[0,1]_{(q)}$).

Levine and Feizi (2021) show that Theorem 1 applies essentially unchanged in the quantized case: in particular, if the domain of $p(\cdot)$ is restricted to $[0,1]_{(q)}$, (and assuming that Λ is a multiple of 1/q) then the theorem still applies when Equation 3 is replaced with:

$$s_i \sim \mathcal{U}_{(q)}(0,\Lambda), \quad \forall i. \tag{20}$$

When this quantized form is used, there are only a discrete number of outcomes ($= \Lambda q$) for each s_i.

Building on this, the second step in the argument is to leverage the fact that Theorem 1 makes no assumption on the joint distribution of s_i's to *couple* all of the elements of $\boldsymbol{s}$. In particular, s_i's are set to have fixed offsets from one another (mod Λ). In other words, the outcomes for each s_i are *cyclic permutations* of each other. This preserves the property that each s_i is uniformly distributed, while also ensuring that there are now only Λq outcomes of the smoothing process *in total*. Then expectation in Equation 13 can be evaluated exactly and efficiently (See Figure 3-a.)

For our Variable-Λ method, we use a similar strategy for derandomization: We quantize the smoothing process in a similar way, modifying Definition 2 by redefining the support of $\mathcal{D}_i$ as $[\frac{1}{q}, 1]_{(q)} \cup \{\infty\}$ and replacing Equation 10 with:

$$s_i \sim \mathcal{U}_{(q)}(0, \Lambda_i). \tag{21}$$

We also define a quanitized version of ECM's as a metric on $[0,1]^d_{(q)}$ where the domain of each g_i is restricted to $[0,1]_{(q)}$. This yields a quantized version of Theorem 2 which we spell out fully in Appendix A.4. The most significant difference occurs in part (c) where we use quantized forms of derivatives:

Theorem 3 (c)**.** *If $\mathcal{D}_i$ is constructed as follows:*

- *On the interval $[\frac{1}{q}, \frac{q-1}{q}]_{(q)}$, Λ_i is distributed as:*

$$\begin{aligned} \Pr(\Lambda_i = z) = -qz \Big[& g_i(z - \frac{1}{q}) \\ & + g_i(z + \frac{1}{q}) - 2g_i(z) \Big] \quad \forall z \in [\frac{1}{q}, \frac{q-1}{q}]_{(q)} \end{aligned} \tag{22}$$

- $\Pr(\Lambda_i = 1) = q \left[g_i(1) - g_i(\frac{q-1}{q}) \right]$
- $\Pr(\Lambda_i = \infty) = 1 - g_i(1)$

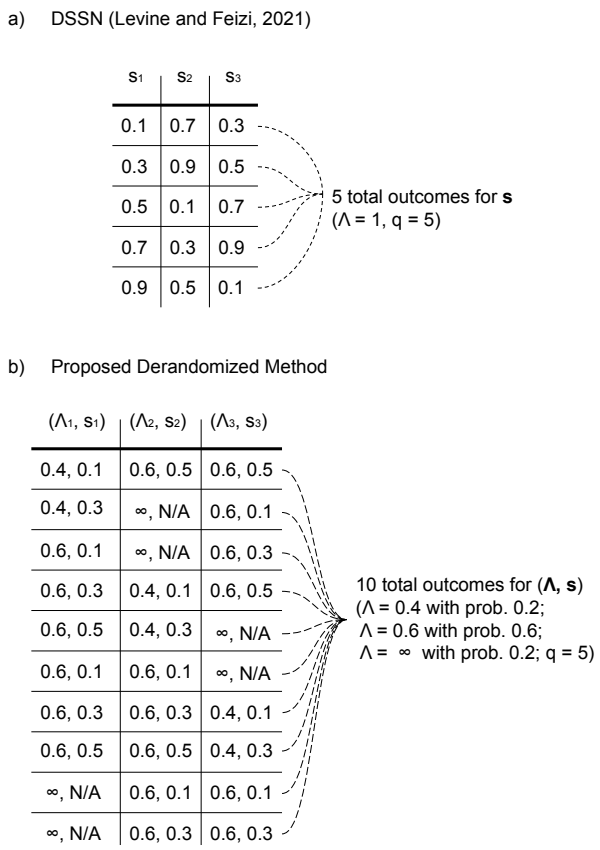

Figure 3: (a) "Fixed offset" method of sampling outcomes in Levine and Feizi (2021). In this case, the fixed offset is that $s_1 = s_2 - 0.6 = s_3 - 0.2 \,(\text{mod } \Lambda)$ Note that in the sample of 5 outcomes, each s_i is uniform on $\{0.1, 0.3, 0.5, 0.7, 0.9\}$, which is to say, $s_i \sim \mathcal{U}_{(5)}(0, 1)$, as desired. (b) Fixed-offset sampling applied to variable-Λ smoothing, for a given distribution of Λ. For each (Λ_i, s_i), we list out each possible outcome, in some cases repeated in order to achieve the desired distribution over Λ. Outcomes are then cyclically permuted for each dimension i to define the coupling. As in Levine and Feizi (2021), the offsets for the cyclic permutations are arbitrary, but fixed throughout training and testing. Specifically, the offsets are chosen pseudorandomly using a fixed seed. We use a seed of 0 for experiments in the main text; other values are explored on CIFAR-10 in Appendix G. Note that Theorem 3 does not require cyclic permutations: choosing arbitrary permutations for each s_i would work. However, storing such arbitrary permutations for each dimension would be highly memory-intensive. In Appendix H, we show (at small-scale: CIFAR-10) that using arbitrary (pseudorandom) permutations as opposed to cyclic permutations confers no practical benefit.

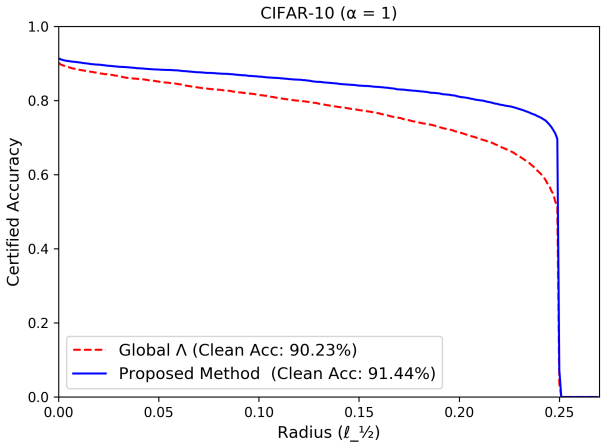

Figure 4: Using a global value for Λ as suggested in Equation 24 leads to suboptimal certified robustness.

then

$$\Pr_i^{split}(z) = g_i(z), \quad \forall z \in [0, 1]. \tag{23}$$

Now that we have defined a quantized version of our smoothing method, we attempt the coupling step (using the fact that Theorem 2 also makes no assumptions about joint distributions of $\boldsymbol{s}$ or Λ). However, this presents greater challenges than the ℓ_1 case. In the ℓ_1 case, all outcomes for each s_i occur with equal probability $1/(\Lambda q)$ so we can arbitrarily associate each outcome for s_1 with a unique outcome for s_2, and so on (for example using the fixed offset method described above). However, Equation 22 assigns real-number probabilities to each value of Λ_i. This means that the outcomes (Λ_i, s_i) for each dimension occur with non-uniform probabilities, making the coupling process more difficult.

One naive solution (at least in the case where g_i's are all the same function, for example for ℓ_p^p metrics) is to couple the Λ_i's such that they are all equal to one another; in other words, the sampling process becomes:

$$\begin{aligned} &\Lambda \sim \mathcal{D}. \\ &s_i \sim \mathcal{U}(0, \Lambda) \quad \forall i \end{aligned} \tag{24}$$

We can then apply the fixed-offset coupling of $\boldsymbol{s}$ for each possible value of Λ, evaluating $q\Lambda$ outcomes for each value. We then exactly compute the final expectation $p(\cdot)$ as the *weighted* average of $f(\cdot)$ over these outcomes, with the weights for each Λ being determined by Theorem 3-c. However, this naive "Global Λ" method underperforms in practice (see Figure 4) and has significant theoretical drawbacks (e.g., notice that this method simply produces the average of several ℓ_1-Lipschitz functions). We explain this further in Appendix B.

What we do instead is to design $\mathcal{D}$ such that, for some constant integer B, all outcomes for (Λ_i, s_i) each have probability in the form n/B, where $n \in \mathbb{N}$. By repeating each outcome n times, this allows us to generate a list of B total outcomes which occur with uniform probability. We then couple these using cyclic permutations as in Levine and Feizi (2021), so that we require a total of B smoothing samples (See Figure 3-b.)

Note that the distribution $\mathcal{D}$ given by Equation 22 is not necessarily of this form. However, even though the distribution given by Equation 22 is in some sense "optimal" in that it causes $\Pr^{\text{split}}(z)$ to perfectly match $g(z)$, thereby providing the most information to the base classifier, it is only necessary for the Lipschitz guarantee that $\Pr^{\text{split}}(z)$ is nowhere greater than $g(z)$. It turns out (as explained in full detail in Appendix C) that for a fixed budget B, finding a distribution over Λ with all outcomes in the form n/B such that $\Pr^{\text{split}}(z)$ approximates but never exceeds a given $g(z)$ can be formulated as a mixed integer linear program. Solving these MILP's yields distributions for Λ that cause $\Pr^{\text{split}}(z)$ to satisfyingly approximate $g(z)$ (see Figure 5.) We also show in the appendix that an arbitrarily close approximation can always be obtained with B sufficiently large.

6 Results

Our results are presented in Table 1 and Figure 3.

In Table 1, we present certificates that our algorithm generates on CIFAR-10 for $\ell_{1/2}$ and $\ell_{1/3}$ quasi-norms. As a baseline, we compare to Levine and Feizi (2021)'s certificates for ℓ_1, using norm inequalities to derive certificates for ℓ_p ($p < 1$). In particular we use the standard norm inequality:

$$\ell_p(\boldsymbol{x}, \boldsymbol{y}) \geq \ell_1(\boldsymbol{x}, \boldsymbol{y}), \ \forall p \in (0, 1) \tag{25}$$

Moreover, since our domain is $[0, 1]$, we have:

$$\begin{aligned} \ell_p(\boldsymbol{x}, \boldsymbol{y}) = \left(\sum_{i=1} \delta_i^p\right)^{1/p} \geq \\ \left(\sum_{i=1}^{d} \delta_i\right)^{1/p} = (\ell_1(\boldsymbol{x}, \boldsymbol{y}))^{1/p}, \\ \forall p \in (0, 1) \end{aligned} \tag{26}$$

This means that we can compute the baseline, ℓ_1-based certificate as:

$$\text{Cert.}(\ell_p) = \max(\text{Cert.}(\ell_1), \text{Cert.}(\ell_1)^{1/p}) \tag{27}$$

Table 1 shows that our method outperforms this baseline significantly on CIFAR-10 at a wide range of scales. For example, at an $\ell_{1/3}$ radius of 720, the proposed method has over 14 percentage points higher certified-robust accuracy, using a model that also has over 14 percentage points higher clean accuracy.

In Figure 6, we present certificate results of our method on ImageNet-1000, showing that our method scales to high-dimensional datasets, with one model able to certify many samples as robust to an $\ell_{1/2}$ radius close to 80 while maintaining over 50% clean accuracy.

Our architecture and training settings were largely borrowed from Levine and Feizi (2021), using WideResNet-40 for CIFAR-10 and ResNet-50 for ImageNet. One additional challenge was the presence of both $\boldsymbol{x}^{\text{lower}}$ and $\boldsymbol{x}^{\text{upper}}$ as inputs to the base classifier $f(\cdot)$. Levine and Feizi (2021) does not need this, because when Λ is fixed, $\boldsymbol{x}^{\text{lower}}$ and $\boldsymbol{x}^{\text{upper}}$ can both be computed from their mean. In order to use the extra information, we doubled the input channels to the first convolutions layer, and represented the two images in different channels. We explore other representation techniques in Appendix F. An explicit description of our certification procedure is provided in Appendix E.

We use 1000α smoothing samples to approximate the metric functions $g = \frac{z^p}{\alpha}$, where $1/\alpha$ is the Lipschitz constant. Note that, unlike in randomized smoothing, this "sampling" does not mean that our final certificates are non-deterministic or approximate: they are exact certificates for the fractional ℓ_p-quasi-norm.

7 Societal Impacts, Limitations, and Environmental Impact

Adversarial attacks represent an important potential security concerns in machine learning: although fractional-ℓ_p attacks have not yet emerged as threats themselves, we consider defending against these attacks proactively as a responsible step. However, it is imperative that practitioners using any defense understand its limitations: we acknowledge that the proposed defense provides provable robustness to only a narrow threat model. We note that we do not present any new attacks or security vulnerabilities in this work.

While our technique does rely on (non-random) sampling to compute certificates, the 1000α smoothing samples we used, with α at most 18, is significantly fewer than the 100,000 smoothing samples typically used in randomized smoothing results (e.g. Cohen et al. (2019); Salman et al. (2019); Yang et al. (2020)). The sampling involved in randomized smoothing is computationally expensive and thus environmentally impactful as a forward-pass is computed on each sample: this means that de-randomization of smoothing techniques can have a positive environmental impact.

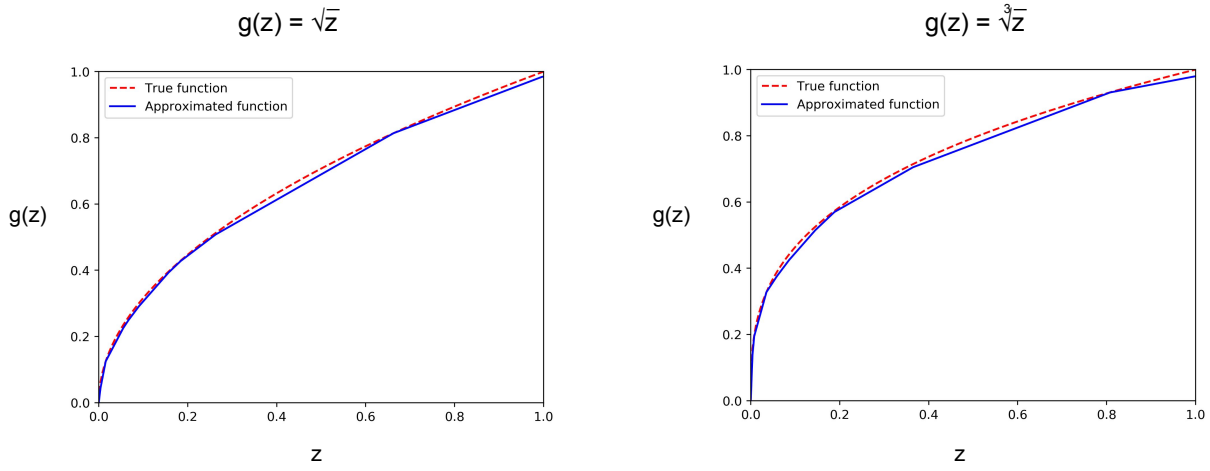

Figure 5: Approximations of g for $p = \frac{1}{2}$ and $p = \frac{1}{3}$ using a budget of $B = 1000$ smoothing samples. In both cases, the true and approximated functions differ by at most 0.02.

$\ell_{1/2}$								
ρ	10	20	30	40	50	60	70	80
L&F (2021) (From ℓ_1)	42.69% (60.42% @ α=18)	35.04% (60.42% @ α=18)	28.89% (60.42% @ α=18)	23.46% (60.42% @ α=18)	18.81% (60.42% @ α=18)	13.76% (60.42% @ α=18)	8.38% (60.42% @ α=18)	1.27% (60.42% @ α=18)
L&F (2021) (From ℓ_1) (Stab. Training)	41.32% (55.38% @ α=12)	35.56% (50.11% @ α=18)	32.07% (50.11% @ α=18)	28.70% (50.11% @ α=18)	24.95% (50.11% @ α=18)	20.79% (50.11% @ α=18)	16.20% (50.11% @ α=18)	6.98% (50.11% @ α=18)
Variable-Λ	**56.74%** (73.22% @ α=15)	**49.80%** (70.57% @ α=18)	43.60% (70.57% @ α=18)	37.97% (70.57% @ α=18)	32.37% (70.57% @ α=18)	25.83% (70.57% @ α=18)	18.19% (70.57% @ α=18)	5.02% (70.57% @ α=18)
Variable-Λ (Stab. Training)	55.21% (69.87% @ α=9)	48.72% (62.74% @ α=15)	**45.05%** (60.44% @ α=18)	**42.26%** (60.44% @ α=18)	**38.62%** (60.44% @ α=18)	**34.42%** (60.44% @ α=18)	**29.01%** (60.44% @ α=18)	**16.28%** (60.44% @ α=18)
$\ell_{1/3}$								
ρ	90	180	270	360	450	540	630	720
L&F (2021) (From ℓ_1)	34.98% (60.42% @ α=18)	27.86% (60.42% @ α=18)	22.69% (60.42% @ α=18)	18.49% (60.42% @ α=18)	14.32% (60.42% @ α=18)	10.37% (60.42% @ α=18)	5.99% (60.42% @ α=18)	0.89% (60.42% @ α=18)
L&F (2021) (From ℓ_1) (Stab. Training)	35.54% (50.11% @ α=18)	31.30% (50.11% @ α=18)	28.06% (50.11% @ α=18)	24.75% (50.11% @ α=18)	21.33% (50.11% @ α=18)	18.27% (50.11% @ α=18)	13.97% (50.11% @ α=18)	6.07% (50.11% @ α=18)
Variable-Λ	**55.66%** (74.57% @ α=18)	49.04% (74.57% @ α=18)	43.27% (74.57% @ α=18)	38.21% (74.57% @ α=18)	33.37% (74.57% @ α=18)	27.17% (74.57% @ α=18)	20.27% (74.57% @ α=18)	6.87% (74.57% @ α=18)
Variable-Λ (Stab. Training)	54.63% (70.21% @ α=12)	**49.88%** (64.30% @ α=18)	**46.92%** (64.30% @ α=18)	**44.11%** (64.30% @ α=18)	**41.03%** (64.30% @ α=18)	**37.56%** (64.30% @ α=18)	**32.46%** (64.30% @ α=18)	**20.84%** (64.30% @ α=18)

Table 1: Certified accuracy as a function of fractional ℓ_p distance ρ, for $p = 1/2$ and $1/3$, on CIFAR-10. We train using standard smoothed training (Cohen et al., 2019) as well as with stability training (Li et al., 2019b). As a baseline, we compare to certificates computed from the ℓ_1 certificates given by Levine and Feizi (2021). We test with $\alpha = \{1, 3, 6, 9, 12, 15, 18\}$ where $1/\alpha$ is the Lipschitz constant of the model (as mentioned in Section 4, for Levine and Feizi (2021), $\Lambda = \alpha$), and report the highest certificate for each technique over all of the models. In parentheses, we report the the clean accuracy and the α parameter for the associated model. Complete results for all models are reported in Appendix I, as are base classifier accuracies for each model. For $p = 1/2$, we also provide results for larger values of α (up to $\alpha = 30$) in Appendix J.

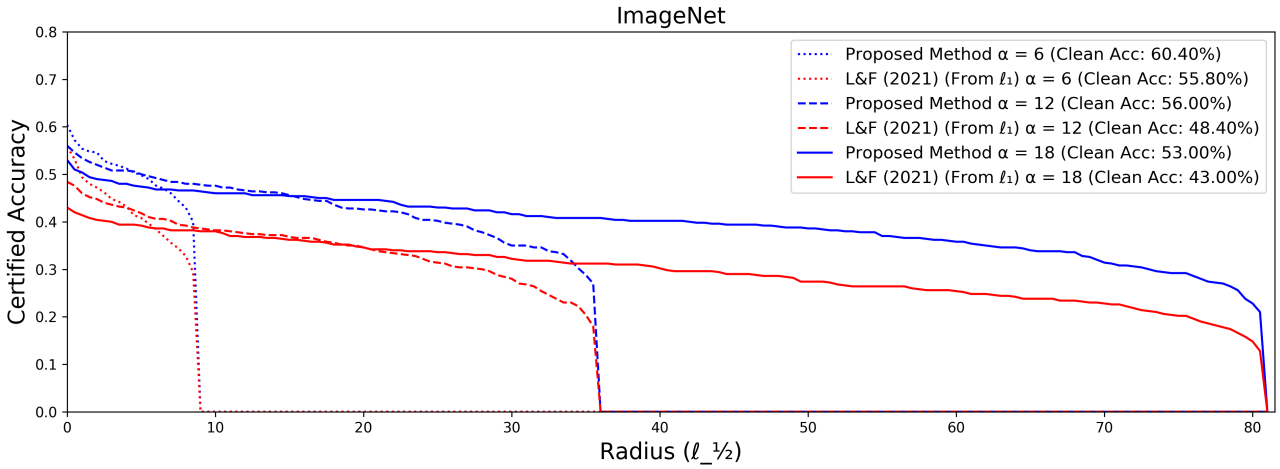

Figure 6: Certified Accuracy for variable-Λ smoothing as a function of $\ell_{1/2}$ norm on ImageNet-1000. We use a subset consisting of 500 samples from the validation set for evaluation. $1/\alpha$ is the $\ell_{1/2}^{1/2}$ Lipschitz constant of the classifier logits. As a baseline, we compare to certificates computed from the ℓ_1 certificates given by Levine and Feizi (2021). Base classifier accuracies are reported in Appendix K.

8 Conclusion

In this work, we presented a novel technique for implementing trainable models that have Lipschitz continuity under a wide family of elementwise-concave metrics, including fractional ℓ_p^p metrics. This allows us to develop certifiably robust classifiers with robustness guarantees under ℓ_p, $p < 1$ attacks, a domain that has not been thoroughly studied in the adversarial robustness literature. Our method is fully deterministic and is demonstrated on both CIFAR-10 and ImageNet, showing that it efficiently scales. This leaves the open problem of deterministic certification at ImageNet scale for ℓ_p, $p > 1$ attacks, and in particular the widely studied ℓ_2 norm. While our technique cannot be directly applied in this domain (because of the concavity requirement of our results), we hope that it can provide a starting point for future exploration into this problem.

9 Acknowledgements

This project was supported in part by NSF CAREER AWARD 1942230, a grant from NIST 60NANB20D134, HR001119S0026 (GARD) and ONR grant 13370299.

References

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

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

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1310–1320, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL `http://proceedings.mlr.press/v97/cohen19c.html`.

Alexander Levine and Soheil Feizi. Improved, deterministic smoothing for l1 certified robustness. In *ICML*, 2021.

Hao Wang, Xiangyu Yang, and Xin Deng. A hybrid first-order method for nonconvex ℓ_p-ball constrained optimization, 2021.

Alexander Levine and Soheil Feizi. Robustness certifi-

cates for sparse adversarial attacks by randomized ablation. In *AAAI*, 2020a.

Alexander Levine and Soheil Feizi. Deep partition aggregation: Provable defenses against general poisoning attacks. In *International Conference on Learning Representations*, 2020b.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pages 5283–5292, 2018.

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

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Semidefinite relaxations for certifying robustness to adversarial examples. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 10900–10910, Red Hook, NY, USA, 2018. Curran Associates Inc.

Vincent Tjeng, Kai Y. Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyGIdiRqtm.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, pages 4939–4948. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/d04863f100d59b3eb688a11f95b0ae60-Paper.pdf.

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

Cem Anil, James Lucas, and Roger Grosse. Sorting out Lipschitz function approximation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 291–301. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/anil19a.html.

Matt Jordan, Justin Lewis, and Alexandros G. Dimakis. Provable certificates for adversarial examples: Fitting a ball in the union of polytopes. In *NeurIPS*, 2019.

Sahil Singla and Soheil Feizi. Skew orthogonal convolutions. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9756–9766. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/singla21a.html.

Sahil Singla and Soheil Feizi. Second-order provable defenses against adversarial attacks. In *International Conference on Machine Learning*, pages 8981–8991. PMLR, 2020.

Asher Trockman and J Zico Kolter. Orthogonalizing convolutional layers with the cayley transform. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Pbj8H_jEHYv.

Hadi Salman, Jerry Li, Ilya Razenshteyn, Pengchuan Zhang, Huan Zhang, Sebastien Bubeck, and Greg Yang. Provably robust deep learning via adversarially trained smoothed classifiers. In *Advances in Neural Information Processing Systems*, pages 11292–11303, 2019.

Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 656–672. IEEE, 2019.

Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Certified adversarial robustness with additive noise. In *Advances in Neural Information Processing Systems*, pages 9464–9474, 2019b.

Guang-He Lee, Yang Yuan, Shiyu Chang, and Tommi Jaakkola. Tight certificates of adversarial robustness for randomly smoothed classifiers. In *Advances in Neural Information Processing Systems*, pages 4910–4921, 2019.

Greg Yang, Tony Duan, J. Edward Hu, Hadi Salman, Ilya Razenshteyn, and Jerry Li. Randomized smoothing of all shapes and sizes. In *Proceedings of the 37th International Conference on Machine Learning*, pages 10693–10705, 2020.

Runtian Zhai, Chen Dan, Di He, Huan Zhang, Boqing Gong, Pradeep Ravikumar, Cho-Jui Hsieh, and Liwei Wang. Macer: Attack-free and scalable robust training via maximizing certified radius. In *International Conference on Learning Representations*, 2019.

Jongheon Jeong and Jinwoo Shin. Consistency regularization for certified robustness of smoothed classifiers. *Advances in Neural Information Processing Systems*, 33, 2020.

Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. In Jan Van den Bussche and Victor Vianu, editors, *Database Theory — ICDT 2001*, pages 420–434, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. ISBN 978-3-540-44503-6.

Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, SCG '04, page 253–262, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138857. doi: 10.1145/997817.997857. URL `https://doi.org/10.1145/997817.997857`.

Dimitris G. Chachlakis and Panos P. Markopoulos. Novel algorithms for lp-quasi-norm principal-component analysis. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 1045–1049, 2021. doi: 10.23919/Eusipco47968.2020.9287335.

Peter Howarth and Stefan Rüger. Fractional distance measures for content-based image retrieval. In David E. Losada and Juan M. Fernández-Luna, editors, *Advances in Information Retrieval*, pages 447–456, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-31865-1.

Alexander Levine and Soheil Feizi. (de) randomized smoothing for certifiable defense against patch attacks. *Advances in Neural Information Processing Systems*, 33:6465–6475, 2020c.

Supplementary Material: Provable Adversarial Robustness for Fractional ℓ_p Threat Models

A Proofs

A.1 Proof Sketch of Theorem 1 (from Levine and Feizi (2021)) using modified notation

Suppose the $[0,1]$ domain of dimension i is divided into "bins", with dividers at each value $s_i + n\Lambda, \forall n \in \mathbb{N}$ (See Figure 1 in the main text) Then x_i^{lower} and x_i^{upper} are the lower- and upper-limits of the bin which x_i is assigned to. Note that the bins are of size Λ, and that the offset s_i of the dividers is uniformly random. Consider two points $\boldsymbol{x}$ and $\boldsymbol{y}$, and let $\delta_i := |x_i - y_i|$. Then the probability of a divider separating x_i and y_i is $\min(\delta_i/\Lambda, 1)$. By union bound, the probability that $(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}})$ and $(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}})$ differ at all is at most $\Sigma\delta_i/\Lambda = \|x - y\|_1/\Lambda$. If $\boldsymbol{x}$ and $\boldsymbol{y}$ are mapped to the same bin in every dimension, $f(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) = f(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}})$. Because the range of f is restricted to the interval $[0,1]$, this implies that $|p(\boldsymbol{x}) - p(\boldsymbol{y})| = |\,\mathbb{E}_{\boldsymbol{s}}\left[f(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}})\right] - \mathbb{E}_{\boldsymbol{s}}\left[f(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}})\right]\,| \leq \|x - y\|_1/\Lambda$.

A.2 Proof of Theorem 2

We first need the following lemma, which is implicit in the proofs in Levine and Feizi (2021), but which we prove explicitly here for completeness. Note that we closely follow the proof of Theorem 1 in Levine and Feizi (2021)

Lemma 1. *For any $\Lambda_i \in (0,1] \cup \{\infty\}$, let $s_i \sim \mathcal{U}(0, \Lambda_i)$. For any $x_i, y_i \in [0,1]$, let $\delta_i := |x_i - y_i|$ and define x_i^{upper}, x_i^{lower} as follows: If $\Lambda_i = \infty$, then $x_i^{upper} := 1$, $x_i^{lower} := 0$, otherwise:*

$$x_i^{upper} := \min(\Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i, 1) \tag{28}$$

$$x_i^{lower} := \max(\Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i - \Lambda_i, 0) \tag{29}$$

and define y_i^{upper}, y_i^{lower} similarly. Then:

$$\Pr_{s_i}((x_i^{lower}, x_i^{upper}) \neq (y_i^{lower}, y_i^{upper})) = \min\left(\frac{\delta_i}{\Lambda_i}, 1\right) \tag{30}$$

Proof. We first assume $\Lambda_i \in (0,1]$. Without loss of generality, assume $x_i \geq y_i$, so that $\delta_i = x_i - y_i$.

Note that, with probability 1, $(x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})$ iff $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{y_i - s_i}{\Lambda_i} \rceil$.

To see this, note that $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i - s_i}{\Lambda_i} \rceil \implies (x_i^{\text{lower}}, x_i^{\text{upper}}) = (y_i^{\text{lower}}, y_i^{\text{upper}})$ directly from the definitions.

For the converse, $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{y_i - s_i}{\Lambda_i} \rceil \implies (x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})$, first consider the case where $\Lambda_i < 1$. Because the first terms in the "min" or "max" of the definitions of x_i^{lower} and x_i^{upper} differ by a most $\Lambda_i < 1$, both of the $[0,1]$ box constraints cannot be active simultaneously: either $x_i^{\text{lower}} = \Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i - \Lambda_i$ (and not 0) and/or $x_i^{\text{upper}} = \Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i$ (and not 1). Therefore if $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{y_i - s_i}{\Lambda_i} \rceil$, whichever of x_i^{upper} or x_i^{lower} is not affected by the box constraint will necessarily differ from y_i^{upper} or y_i^{lower}, so $(x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})$. For the case where $\Lambda_i = 1$, both constraints can only be simultaneously active if $s_i = 0$ or $s_i = 1$, which both occur with probability zero, and otherwise the same argument from the $\Lambda_i < 1$ case applies.

Therefore, it is sufficient to show that

$$\Pr_{s_i}(\lceil \frac{x_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{y_i - s_i}{\Lambda_i} \rceil) = \min\left(\frac{\delta_i}{\Lambda_i}, 1\right) \tag{31}$$

First, if $\delta_i/\Lambda_i \geq 1$, then $\frac{x_i - s_i}{\Lambda_i}$ and $\frac{y_i - s_i}{\Lambda_i}$ differ by at least one, so their ceilings must differ. Then $\Pr_{s_i}(\lceil \frac{x_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{y_i - s_i}{\Lambda_i} \rceil) = 1 = \min\left(\frac{\delta_i}{\Lambda_i}, 1\right)$.

Otherwise, if $\delta_i/\Lambda_i < 1$, then $\frac{x_i}{\Lambda_i}$ and $\frac{y_i}{\Lambda_i}$ differ by less than one, so

$$\lceil \frac{x_i}{\Lambda_i} \rceil - \lceil \frac{y_i}{\Lambda_i} \rceil \in \{0, 1\} \tag{32}$$

And similarly:

$$\lceil \frac{x_i - s_i}{\Lambda_i} \rceil - \lceil \frac{y_i - s_i}{\Lambda_i} \rceil \in \{0, 1\} \tag{33}$$

Also, because s_i/Λ_i is at most one,

$$\begin{aligned} \lceil \frac{x_i}{\Lambda_i} \rceil - \lceil \frac{x_i - s_i}{\Lambda_i} \rceil \in \{0, 1\} \\ \lceil \frac{y_i}{\Lambda_i} \rceil - \lceil \frac{y_i - s_i}{\Lambda_i} \rceil \in \{0, 1\} \end{aligned} \tag{34}$$

We consider cases on $\lceil \frac{x_i}{\Lambda_i} \rceil - \lceil \frac{y_i}{\Lambda_i} \rceil$:

- Case $\lceil \frac{x_i}{\Lambda_i} \rceil - \lceil \frac{y_i}{\Lambda_i} \rceil = 0$. Then $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i - s_i}{\Lambda_i} \rceil$ only in two cases:
  - $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i}{\Lambda_i} \rceil$ iff $\frac{s_i}{\Lambda_i} < \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1)$ $(\leq \frac{x_i}{\Lambda_i} - (\lceil \frac{x_i}{\Lambda_i} \rceil - 1))$.
  - $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i}{\Lambda_i} \rceil - 1$ iff $\frac{s_i}{\Lambda_i} > \frac{x_i}{\Lambda_i} - (\lceil \frac{x_i}{\Lambda_i} \rceil - 1)$ $(\geq \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1))$.

  Then $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{x_i - s_i}{\Lambda_i} \rceil$ iff $\frac{y_i}{\Lambda_i} - (\lceil \frac{x_i}{\Lambda_i} \rceil - 1) < \frac{s_i}{\Lambda_i} < \frac{x_i}{\Lambda_i} - (\lceil \frac{x_i}{\Lambda_i} \rceil - 1)$. There are exactly $q(x_i - y_i) = q\delta_i$ values of s_i for which this occurs, out of a total $\Lambda_i q$ values of s_i, so this occurs with probability $\frac{\delta_i}{\Lambda_i}$.

- Case $\lceil \frac{x_i}{\Lambda_i} \rceil - \lceil \frac{y_i}{\Lambda_i} \rceil = 1$. Then $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{y_i - s_i}{\Lambda_i} \rceil$ only in two cases:
  - $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i}{\Lambda_i} \rceil$ and $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i}{\Lambda_i} \rceil = \lceil \frac{y_i}{\Lambda_i} \rceil + 1$. This happens iff $\frac{s_i}{\Lambda_i} < \frac{x_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil$ $(\leq \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1))$.
  - $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i}{\Lambda_i} \rceil - 1$ and $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i}{\Lambda_i} \rceil$. This happens iff $\frac{s_i}{\Lambda_i} > \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1)$ $(\geq \frac{x_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil)$.

  Therefore, $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i - s_i}{\Lambda_i} \rceil$ iff:

  $$\frac{x_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil < \frac{s_i}{\Lambda_i} < \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1) \tag{35}$$

  Which is:

  $$\frac{y_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil + \frac{\delta_i}{\Lambda_i} < \frac{s_i}{\Lambda_i} < \frac{y_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil + 1. \tag{36}$$

  There are exactly $q(1 - (x_i - y_i)) = q(1 - \delta_i)$ values of s_i for which this occurs, out of a total $\Lambda_i q$ values of s_i, so this occurs with probability $1 - \frac{\delta_i}{\Lambda_i}$. Then $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{x_i - s_i}{\Lambda_i} \rceil$ with probability $\frac{\delta_i}{\Lambda_i}$.

So in all cases, for $\delta_i/\Lambda_i < 1$, $\Pr_{s_i}(\lceil \frac{x_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{y_i - s_i}{\Lambda_i} \rceil) = \frac{\delta_i}{\Lambda_i} = \min\left(\frac{\delta_i}{\Lambda_i}, 1\right)$.

Finally, we consider $\Lambda_i = \infty$. In this case, $(x_i^{\text{lower}}, x_i^{\text{upper}}) = (y_i^{\text{lower}}, y_i^{\text{upper}})) = (0, 1)$ with probability 1, so

$$\Pr_{s_i}((x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})) = 0 = \frac{\delta_i}{\infty} = \min\left(\frac{\delta_i}{\Lambda_i}, 1\right) \tag{37}$$

□

We can now prove each part of the theorem:

Part 1. *Let $\mathcal{D}$ and $f(\cdot)$ be the Λ-distribution and base function used for Variable-Λ smoothing, respectively. Let $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^d$ be two points. For each dimension i, let $\delta_i := |x_i - y_i|$. The probability that $(x_i^{lower}, x_i^{upper}) \neq (y_i^{lower}, y_i^{upper})$ is given by $\Pr_i^{split}(\delta_i)$, where:*

$$\Pr_i^{split}(z) := \Pr_{\mathcal{D}_i}(\Lambda_i \leq z) + z \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (z,1])}}{\Lambda_i}\right] \tag{38}$$

Proof.

$$\begin{aligned}
&\Pr((x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})) = \\
&\mathbb{E}[\mathbf{1}_{(x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})}] = \\
&\mathop{\mathbb{E}}_{\Lambda_i \sim \mathcal{D}_i}[\mathbb{E}[\mathbf{1}_{(x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})} | \Lambda_i]] = \\
&\mathop{\mathbb{E}}_{\Lambda_i \sim \mathcal{D}_i}[\Pr[(x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}}) | \Lambda_i]] = \\
&\mathop{\mathbb{E}}_{\Lambda_i \sim \mathcal{D}_i}[\min\left(\frac{\delta_i}{\Lambda_i}, 1\right)] = \\
&\mathop{\mathbb{E}}_{\Lambda_i \sim \mathcal{D}_i}[\frac{\delta_i}{\Lambda_i} \cdot \mathbf{1}_{\Lambda_i > \delta_i}] + \mathop{\mathbb{E}}_{\Lambda_i \sim \mathcal{D}_i}[1 \cdot \mathbf{1}_{\Lambda_i \leq \delta_i}] = \\
&\mathop{\mathbb{E}}_{\Lambda_i \sim \mathcal{D}_i}[\frac{\delta_i}{\Lambda_i} \cdot \mathbf{1}_{\Lambda_i > \delta_i}] + \mathop{\mathbb{E}}_{\Lambda_i \sim \mathcal{D}_i}[1 \cdot \mathbf{1}_{\Lambda_i \leq \delta_i}] = \\
&\delta_i \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (\delta_i, 1])}}{\Lambda_i}\right] + \Pr_{\mathcal{D}_i}(\Lambda_i \leq \delta_i) = \Pr_i^{\text{split}}(\delta_i)
\end{aligned} \tag{39}$$

Where we use the law of total expectation in the third line, Lemma 1 in the fifth line, and in the last line, we use that δ_i is finite, so $\delta_i/\infty = 0$ □

Part 2. *Let $d(\cdot,\cdot)$ be an ECM defined by concave functions $g_1, ..., g_d$. Let $\mathcal{D}$ and $f(\cdot)$ be the Λ-distribution and base function used for Variable-Λ smoothing, respectively. If $\forall i \in [d]$ and $\forall z \in [0,1]$,*

$$\Pr_i^{split}(z) \leq g_i(z), \tag{40}$$

then, the smoothed function $p_{\mathcal{D},f}(\cdot)$ is 1-Lipschitz with respect to the metric $d(\cdot,\cdot)$.

Proof. Let $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^d$ be two points. For each dimension i, let $\delta_i := |x_i - y_i|$. By union bound:

$$\begin{aligned}
&\Pr_{\boldsymbol{s}}((\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) \neq (\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}})) = \\
&\Pr_{\boldsymbol{s}}\left[\bigcup_{i=1}^{d}(x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})\right] \leq \\
&\sum_{i=1}^{d} \Pr_i^{\text{split}}(\delta_i) \leq \\
&\sum_{i=1}^{d} g_i(\delta_i) = d(\boldsymbol{x}, \boldsymbol{y})
\end{aligned} \tag{41}$$

Then:

$$
\begin{aligned}
&|p_{\mathcal{D},f}(\boldsymbol{x}) - p_{\mathcal{D},f}(\boldsymbol{y})| \\
&= \left| \mathop{\mathbb{E}}_{\boldsymbol{s}} \left[f(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) \right] - \mathop{\mathbb{E}}_{\boldsymbol{s}} \left[f(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) \right] \right| \\
&= \left| \mathop{\mathbb{E}}_{\boldsymbol{s}} \left[f(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) - f(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) \right] \right| \\
&= \Bigg| \Pr_{\boldsymbol{s}}((\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) \neq (\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}})) \mathop{\mathbb{E}}_{\boldsymbol{s}} \left[f(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) - f(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) | (\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) \neq (\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) \right] \\
&+ \Pr_{\boldsymbol{s}}((\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) = (\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}})) \left[f(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) - f(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) | (\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) = (\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) \right] \Bigg|
\end{aligned} \tag{42}
$$

Because $\mathbb{E}_{\boldsymbol{s}} \left[f(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) - f(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) | (\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) = (\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) \right]$ is zero, we have:

$$
\begin{aligned}
&|p_{\mathcal{D},f}(\boldsymbol{x}) - p_{\mathcal{D},f}(\boldsymbol{y})| \\
&= \Pr_{\boldsymbol{s}}((\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) \neq (\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}})) \left| \mathop{\mathbb{E}}_{\boldsymbol{s}} \left[f(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) - f(\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) | (\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}}) \neq (\boldsymbol{y}^{\text{lower}}, \boldsymbol{y}^{\text{upper}}) \right] \right| \\
&\leq d(\boldsymbol{x}, \boldsymbol{y}) \cdot 1
\end{aligned} \tag{43}
$$

In the last step, we use Equation 41 and the assumption that $f(\cdot, \cdot) \in [0, 1]$. Therefore, by the definition of Lipschitz-continuity, $p_{\mathcal{D},f}$ is 1-Lipschitz with respect to $d(\cdot, \cdot)$.

□

Part 3. *Suppose g_i is continuous and twice-differentiable on the interval $(0, 1]$. Let $\mathcal{D}_i$ be constructed as follows:*

- *On the interval $(0, 1)$, Λ_i is distributed continuously, with pdf function:*

$$pdf_{\Lambda_i}(z) = -z g_i''(z) \tag{44}$$

- $\Pr(\Lambda_i = 1) = g_i'(1)$
- $\Pr(\Lambda_i = \infty) = 1 - g_i(1)$

then,

$$\Pr_i^{split}(z) = g_i(z) \quad \forall z \in [0, 1]. \tag{45}$$

If all $\mathcal{D}_i$ are constructed this way, then the conclusion of part (b) above applies.

Proof. We first show that this is in fact a normalized probability distribution:

$$
\begin{aligned}
\int_0^1 \text{pdf}_{\Lambda_i}(z) dz + \Pr(\Lambda_i = 1) + \Pr(\Lambda_i = \infty) = \\
\int_0^1 -z g_i''(z) dz + g_i'(1) + 1 - g_i(1) = \\
-\left(1 \cdot g_i'(1) - 0 \cdot g_i'(0) - \int_0^1 1 \cdot g_i'(z) dz \right) + g_i'(1) + 1 - g_i(1) = \\
-g_i'(1) + \int_0^1 g_i'(z) dz + g_i'(1) + 1 - g_i(1) = \\
g_i(1) - g_i(0) + 1 - g_i(1) = 1
\end{aligned} \tag{46}
$$

Where we use integration by parts in the third line, and the fact that $g_i(0) = 0$ in the last line.

We now show that $\Pr_i^{\text{split}}(z) = g_i(z)$ in the special case of $z = 1$:

$$\begin{aligned}\Pr_i^{\text{split}}(1) &= \Pr_{\mathcal{D}_i}(\Lambda_i \leq 1) + 1 \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (1,1])}}{\Lambda_i}\right] \\ &= \Pr_{\mathcal{D}_i}(\Lambda_i \leq 1) \\ &= 1 - \Pr_{\mathcal{D}_i}(\Lambda_i = \infty) \\ &= 1 - (1 - g_i(1)) = g_i(1)\end{aligned} \tag{47}$$

Where in the second line, we use that $(1, 1]$ represents the empty set, so the term in the expectation is always zero.

Now, we handle the remaining case of $z \in [0, 1)$:

$$\begin{aligned}\Pr_i^{\text{split}}(z) &= \Pr_{\mathcal{D}_i}(\Lambda_i \leq z) + z \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (z,1])}}{\Lambda_i}\right] \\ &= \int_0^z \text{pdf}_{\Lambda_i}(w)dw + z\left[\int_z^1 \text{pdf}_{\Lambda_i}(w) \cdot \frac{1}{w}dw + \Pr(\Lambda = 1)\frac{1}{1}\right] \\ &= \int_0^z -wg_i''(w)dw + z\left[\int_z^1 -wg_i''(w) \cdot \frac{1}{w}dw + g_i'(1)\right] \\ &= -\left[zg_i'(z) - 0 \cdot g_i'(0) - \int_0^z 1 \cdot g_i'(w)dw\right] + z\left[-\int_z^1 g_i''(w)dw + g_i'(1)\right] \\ &= -\left[zg_i'(z) - (g_i(z) - g_i(0))\right] + z\left[-[g_i'(1) - g_i'(z)] + g_i'(1)\right] \\ &= -zg_i'(z) + g_i(z) - zg_i'(1) + zg_i'(z) + zg_i'(1) = g_i(z)\end{aligned} \tag{48}$$

Where we use integration by parts in the fourth line, and the fact that $g_i(0) = 0$ in the last line.

Now we have that $\Pr_i^{\text{split}}(z) = g_i(z)\ \forall z \in [0, 1]$, as desired. The final statement follows directly from Part b. □

A.3 Proof of Corollary 1

Corollary 1. *For all $p \in (0, 1]$, $\alpha \in [1, \infty)$, if we perform Variable-Λ smoothing with all Λ_i's distributed identically (but not necessarily independently) as follows:*

$$\begin{aligned}\Lambda_i &\sim \textit{Beta}(p, 1), \ \textit{with prob.}\ \frac{1-p}{\alpha} \\ \Lambda_i &= 1, \ \textit{with prob.}\ \frac{p}{\alpha} \\ \Lambda_i &= \infty, \ \textit{with prob.}\ 1 - \frac{1}{\alpha}\end{aligned} \tag{49}$$

then, the resulting smoothed function will be $1/\alpha$-Lipschitz with respect to the ℓ_p^p metric

Proof. We consider the ECM defined as $\forall i,\ g_i(z) = \frac{z^p}{\alpha}$. One can easily verify that this is a valid ECM, and that it is twice-differentiable on $(0, 1]$.

We then apply Theorem 2-c:

- On the interval $(0, 1)$, we distribute Λ_i continuously, with pdf function:

$$\text{pdf}_{\Lambda_i}(z) = -zg_i''(z) = \frac{-p(p-1)z^{p-1}}{\alpha} = \frac{1-p}{\alpha} \cdot pz^{p-1} = \frac{1-p}{\alpha} \cdot \text{pdf}_{\text{Beta}(p,1)}(z) \tag{50}$$

- $\Pr(\Lambda_i = 1) = g_i'(1) = \frac{p \cdot 1^{p-1}}{\alpha} = \frac{p}{\alpha}$
- $\Pr(\Lambda_i = \infty) = 1 - g_i(1) = 1 - \frac{1}{\alpha}$

So distributing Λ as stated in the Corollary will result in $\Pr_i^{\text{split}}(z) = g_i(z)\ \forall z \in [0,1]$, and therefore the resulting smoothed function will be 1-Lipschitz w.r.t. the ECM. Then, from the definition of Lipschitzness and of the ECM, we have, for all $\boldsymbol{x}$, $\boldsymbol{y}$:

$$|p_{\mathcal{D},f}(\boldsymbol{x}) - p_{\mathcal{D},f}(\boldsymbol{y})| \leq \sum_{i=1}^{d} \frac{|x_i - y_i|^p}{\alpha} = \frac{1}{\alpha}\ell_p^p(\boldsymbol{x}, \boldsymbol{y}) \tag{51}$$

So $p_{\mathcal{D},f}$ is also $1/\alpha$-Lipschitz w.r.t. the ℓ_p^p metric. □

A.3.1 $\alpha < 1$ Case for Corollary 1

In a footnote in the main text, we mentioned that this technique cannot be applied directly to the $\alpha < 1$ case. To explain, note that taking

$$g_i(z) := \frac{z^p}{\alpha},\ \ \forall i \tag{52}$$

with $\alpha < 1$ is not a properly-defined ECM, because $g_i \notin [0,1] \to [0,1]$: for example, $g_i(1) = 1/\alpha > 1$. However, for the purpose of building a Lipschitz classifier with range $[0,1]$, we can instead define:

$$g_i(z) := \min(\frac{z^p}{\alpha}, 1),\ \ \forall i \tag{53}$$

This is a proper ECM. Furthermore, for functions $p(\boldsymbol{x}) \in [0,1]^d \to [0,1]$, it is equivalent to be 1-Lipschitz with respect to the ECM defined above in Equation 53 and to be 1-Lipschitz with respect to the "improper" ECM defined in Equation 52. To show that 1-Lipschitzness with respect to Equation 53 implies 1-Lipschitzness with respect to Equation 52, simply note that, $\forall \boldsymbol{x}, \boldsymbol{y}$:

$$|p(\boldsymbol{x}) - p(\boldsymbol{y})| \leq \sum_{i=1}^{d} \min(\frac{|x_i - y_i|^p}{\alpha}, 1) \leq \sum_{i=1}^{d} \frac{|x_i - y_i|^p}{\alpha} \tag{54}$$

To show the opposite direction, consider a function p which is 1-Lipschitz w.r.t. Equation 52, and note that $\forall \boldsymbol{x}, \boldsymbol{y}$, either:

- $\exists i : \frac{|x_i - y_i|^p}{\alpha} > 1$. Then $d(\boldsymbol{x}, \boldsymbol{y}) \geq 1$ for both metrics, so the 1-Lipschitz constraint is vacuously true regardless of the values of $p(\boldsymbol{x}), p(\boldsymbol{y})$.
- $\nexists i : \frac{|x_i - y_i|^p}{\alpha} > 1$. Then

$$|p(\boldsymbol{x}) - p(\boldsymbol{y})| \leq \sum_{i=1}^{d} \frac{|x_i - y_i|^p}{\alpha} = \sum_{i=1}^{d} \min(\frac{|x_i - y_i|^p}{\alpha}, 1) \tag{55}$$

  Therefore, we can consider the ECM in Equation 53 to derive an appropriate Lipschitz constraint for the ℓ_p^p metric. However, note that this is not twice-differentiable, so Theorem 2-c does not directly apply. We can however derive an ad-hoc distribution $\mathcal{D}_i$ such that, according to Theorem 2-a, $\Pr_i^{\text{split}}(z) = g_i(z),\ \ \forall z, i$.

  In particular, we use:

  - On the interval $(0, \alpha^{1/p})$, we distribute Λ_i continuously, with pdf function:

$$\text{pdf}_{\Lambda_i}(z) = \frac{1-p}{\alpha} \cdot p z^{p-1} \tag{56}$$

  - $\Pr(\Lambda_i = \alpha^{1/p}) = p$

We first show that $\Pr_i^{\text{split}}(z) = g_i(z)$ in the case of $z \geq \alpha^{1/p}$:

$$\begin{aligned} \Pr_i^{\text{split}}(z) &= \Pr_{\mathcal{D}_i}(\Lambda_i \leq z) + 1 \ \mathbb{E}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (z,1])}}{\Lambda_i}\right] \\ &= \Pr_{\mathcal{D}_i}(\Lambda_i \leq 1) + 0 \\ &= 1 = \min(\frac{z^p}{\alpha}, 1) = g_i(z) \end{aligned} \tag{57}$$

Now, we handle the remaining case of $z \in [0, \alpha^{1/p})$:

$$\begin{aligned}
\Pr_i^{\text{split}}(z) &= \Pr_{\mathcal{D}_i}(\Lambda_i \leq z) + z \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (z,1])}}{\Lambda_i}\right] \\
&= \int_0^z \text{pdf}_{\Lambda_i}(w)dw + z\left[\int_z^{\alpha^{1/p}} \text{pdf}_{\Lambda_i}(w) \cdot \frac{1}{w}dw + \Pr(\Lambda = \alpha^{1/p})\frac{1}{\alpha^{1/p}}\right] \\
&= \int_0^z \frac{1-p}{\alpha} \cdot pw^{p-1}dw + z\left[\int_z^{\alpha^{1/p}} \frac{1-p}{\alpha} \cdot pw^{p-1}\frac{1}{w}dw + \frac{p}{\alpha^{1/p}}\right] \\
&= \frac{1-p}{\alpha}z^p + z\left[\frac{1}{\alpha}(pz^{p-1} - p\alpha^{(p-1)/p}) + \frac{p}{\alpha^{1/p}}\right] \\
&= \frac{1-p}{\alpha}z^p + \frac{z}{\alpha}(pz^{p-1}) \\
&= \frac{z^p}{\alpha} = g_i(z)
\end{aligned} \tag{58}$$

So we have that $\Pr_i^{\text{split}}(z) = g_i(z)\ \forall z \in [0,1]$, as desired.

A.4 Theorem 3

This is the "quantized" form of Theorem 2. In order to introduce it, we need to define a quantized from of ECMs, as well as a quantized form of our smoothing method:

Definition 5 (Quantized Elementwise-concave metric (QECM))**.** *For any $\boldsymbol{x}, \boldsymbol{y}$, let $\delta_i := |x_i - y_i|$. A quantized elementwise-concave metric (QECM) is a metric on $[0,1]^d_{(q)}$ in the form:*

$$d(\boldsymbol{x}, \boldsymbol{y}) := \sum_{i=1}^d g_i(\delta_i), \tag{59}$$

where $g_1, ..., g_d \subset [0,1]_{(q)} \to [0,1]$ are increasing, concave functions with $g_i(0) = 0$.

Definition 6 (Quantized Variable-Λ smoothing)**.** *For any $f : [0,1]^d \times [0,1]^d \to [0,1]$, and distribution $\mathcal{D} = \{\mathcal{D}_1, ...\mathcal{D}_d\}$, such that each $\mathcal{D}_i$ has support $[1/q, 1]_{(q)} \cup \{\infty\}$, let:*

$$\Lambda_i \sim \mathcal{D}_i \tag{60}$$

If $\Lambda_i = \infty$, then $x_i^{upper} := 1$, $x_i^{lower} := 0$, otherwise:

$$s_i \sim \mathcal{U}(0, \Lambda_i)_{(q)} \tag{61}$$

$$x_i^{upper} := \min(\Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i, 1) \tag{62}$$

$$x_i^{lower} := \max(\Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i - \Lambda_i, 0) \tag{63}$$

$$\tag{64}$$

The quantized smoothed function $p_{D,f} \in [0,1]^d_{(q)} \to [0,1]$ is defined as:

$$p_{\mathcal{D},f}(\boldsymbol{x}) := \mathop{\mathbb{E}}_{\boldsymbol{s}}\left[f(\boldsymbol{x}^{lower}, \boldsymbol{x}^{upper})\right]. \tag{65}$$

Note that we make no assumptions about the joint distributions of Λ or of $\boldsymbol{s}$.

Before we state and prove each part of the theorem, we will need a "quantized" form of Lemma 1: Note again that we closely follow the proof of Corollary 1 in Levine and Feizi (2021), which implicitly contains the same result.

Lemma 2. *For any $\Lambda_i \in [1/q, 1]_q \cup \{\infty\}$, let $s_i \sim \mathcal{U}(0, \Lambda_i)_{(q)}$. For any $x_i, y_i \in [0,1]_{(q)}$, let $\delta_i := |x_i - y_i|$ and define x_i^{upper}, x_i^{lower} as follows: If $\Lambda_i = \infty$, then $x_i^{upper} := 1$, $x_i^{lower} := 0$, otherwise:*

$$x_i^{upper} := \min(\Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i, 1) \tag{66}$$

$$x_i^{lower} := \max(\Lambda_i \lceil \frac{x_i - s_i}{\Lambda_i} \rceil + s_i - \Lambda_i, 0) \tag{67}$$

and define y_i^{upper}, y_i^{lower} similarly. Then:

$$\Pr_{s_i}((x_i^{lower}, x_i^{upper}) \neq (y_i^{lower}, y_i^{upper})) = \min\left(\frac{\delta_i}{\Lambda_i}, 1\right) \tag{68}$$

Proof. The proof is mostly identical to the proof of Lemma 1, with minor differences occurring in the cases on $\lceil \frac{x_i}{\Lambda_i} \rceil - \lceil \frac{y_i}{\Lambda_i} \rceil$, which we show here for completeness:

- Case $\lceil \frac{x_i}{\Lambda_i} \rceil - \lceil \frac{y_i}{\Lambda_i} \rceil = 0$. Then $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i - s_i}{\Lambda_i} \rceil$ only in two cases:
  - $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i}{\Lambda_i} \rceil$ iff $\frac{s_i}{\Lambda_i} < \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1)$ $(\leq \frac{x_i}{\Lambda_i} - (\lceil \frac{x_i}{\Lambda_i} \rceil - 1))$.
  - $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i}{\Lambda_i} \rceil - 1$ iff $\frac{s_i}{\Lambda_i} \geq \frac{x_i}{\Lambda_i} - (\lceil \frac{x_i}{\Lambda_i} \rceil - 1)$ $(\geq \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1))$.

  Then $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{x_i - s_i}{\Lambda_i} \rceil$ iff $\frac{y_i}{\Lambda_i} - (\lceil \frac{x_i}{\Lambda_i} \rceil - 1) \leq \frac{s_i}{\Lambda_i} < \frac{x_i}{\Lambda_i} - (\lceil \frac{x_i}{\Lambda_i} \rceil - 1)$, which occurs with probability $\frac{x_i - y_i}{\Lambda_i} = \frac{\delta_i}{\Lambda_i}$.
- Case $\lceil \frac{x_i}{\Lambda_i} \rceil - \lceil \frac{y_i}{\Lambda_i} \rceil = 1$. Then $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{y_i - s_i}{\Lambda_i} \rceil$ only in two cases:
  - $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i}{\Lambda_i} \rceil$ and $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i}{\Lambda_i} \rceil = \lceil \frac{y_i}{\Lambda_i} \rceil + 1$. This happens iff $\frac{s_i}{\Lambda_i} < \frac{x_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil$ $(\leq \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1))$.
  - $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i}{\Lambda_i} \rceil - 1$ and $\lceil \frac{x_i - s_i}{\Lambda_i} \rceil = \lceil \frac{y_i}{\Lambda_i} \rceil$. This happens iff $\frac{s_i}{\Lambda_i} \geq \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1)$ $(\geq \frac{x_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil)$.

  Therefore, $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil = \lceil \frac{x_i - s_i}{\Lambda_i} \rceil$ iff:

$$\frac{x_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil \leq \frac{s_i}{\Lambda_i} < \frac{y_i}{\Lambda_i} - (\lceil \frac{y_i}{\Lambda_i} \rceil - 1) \tag{69}$$

  Which is:

$$\frac{y_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil + \frac{\delta_i}{\Lambda_i} \leq \frac{s_i}{\Lambda_i} < \frac{y_i}{\Lambda_i} - \lceil \frac{y_i}{\Lambda_i} \rceil + 1 \tag{70}$$

  which occurs with probability $1 - \frac{\delta_i}{\Lambda_i}$. Then $\lceil \frac{y_i - s_i}{\Lambda_i} \rceil \neq \lceil \frac{x_i - s_i}{\Lambda_i} \rceil$ with probability $\frac{\delta_i}{\Lambda_i}$.

□

We now state and prove Theorem 3:

Part 1. *Let $\mathcal{D}$ and $f(\cdot)$ be the Λ-distribution and base function used for Quantized Variable-Λ smoothing, respectively. Let $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^d_{(q)}$ be two points. For each dimension i, let $\delta_i := |x_i - y_i|$. The probability that $(x_i^{lower}, x_i^{upper}) \neq (y_i^{lower}, y_i^{upper})$ is given by $\Pr_i^{split}(\delta_i)$, where:*

$$\Pr_i^{split}(z) := \Pr_{\mathcal{D}_i}(\Lambda_i \leq z) + z \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (z,1])}}{\Lambda_i}\right] \tag{71}$$

Proof. Identical to Theorem 2-a, except using Lemma 2 in place of Lemma 1.

□

Part 2. *Let $d(\cdot,\cdot)$ be a QECM defined by concave functions $g_1, ..., g_d$. Let $\mathcal{D}$ and $f(\cdot)$ be the Λ-distribution and base function used for Quantized Variable-Λ smoothing, respectively. If $\forall i \in [d]$ and $\forall z \in [0,1]_{(q)}$,*

$$\Pr_i^{split}(z) \leq g_i(z), \tag{72}$$

then, the smoothed function $p_{\mathcal{D},f}(\cdot)$ is 1-Lipschitz with respect to the metric $d(\cdot,\cdot)$.

Proof. Identical to Theorem 2-b, except assuming $\boldsymbol{x}, \boldsymbol{y} \in [0,1]^d_{(q)}$ □

Part 3. *If $\mathcal{D}_i$ is constructed as follows:*

- *On the interval $[\frac{1}{q}, \frac{q-1}{q}]_{(q)}$, Λ_i is distributed as:*

$$\Pr(\Lambda_i = z) = -qz\left[g_i\left(z - \frac{1}{q}\right) + g_i\left(z + \frac{1}{q}\right) - 2g_i(z)\right] \quad \forall z \in \left[\frac{1}{q}, \frac{q-1}{q}\right]_{(q)} \tag{73}$$

- $\Pr(\Lambda_i = 1) = q\left[g_i(1) - g_i(\frac{q-1}{q})\right]$
- $\Pr(\Lambda_i = \infty) = 1 - g_i(1)$

then

$$\Pr_i^{split}(z) = g_i(z), \quad \forall z \in [0,1]_{(q)}. \tag{74}$$

Proof. We first show that this is in fact a normalized probability distribution:

$$\begin{aligned}
&\sum_{j=1}^{q-1} \Pr\left(\Lambda_i = \frac{j}{q}\right) + \Pr(\Lambda_i = 1) + \Pr(\Lambda_i = \infty) = \\
&\sum_{j=1}^{q-1} -j\left[g_i\left(\frac{j-1}{q}\right) + g_i\left(\frac{j+1}{q}\right) - 2g_i\left(\frac{j}{q}\right)\right] + q\left[g_i(1) - g_i\left(\frac{q-1}{q}\right)\right] + 1 - g_i(1) = \\
&2\sum_{j=1}^{q-1} j g_i\left(\frac{j}{q}\right) - \sum_{j=0}^{q-2}(j+1)g_i\left(\frac{j}{q}\right) - \sum_{j=2}^{q}(j-1)g_i\left(\frac{j}{q}\right) + q\left[g_i(1) - g_i\left(\frac{q-1}{q}\right)\right] + 1 - g_i(1) = \\
&\sum_{j=2}^{q-2}(2j - (j+1) - (j-1))g_i\left(\frac{j}{q}\right) - g_i(0) + (2-2)g_i\left(\frac{1}{q}\right) + (2(q-1) \\
&-(q-2))g_i\left(\frac{q-1}{q}\right) - (q-1)g_i(1) + q\left[g_i(1) - g_i\left(\frac{q-1}{q}\right)\right] + 1 - g_i(1) = 1
\end{aligned} \tag{75}$$

Where we use the fact that $g_i(0) = 0$ in the last line.

We now show that $\Pr_i^{\text{split}}(z) = g_i(z)$ in the special case of $z = 1$:

$$\begin{aligned}
\Pr_i^{\text{split}}(1) &= \Pr_{\mathcal{D}_i}(\Lambda_i \leq 1) + 1 \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (1,1])}}{\Lambda_i}\right] \\
&= \Pr_{\mathcal{D}_i}(\Lambda_i \leq 1) \\
&= 1 - \Pr_{\mathcal{D}_i}(\Lambda_i = \infty) \\
&= 1 - (1 - g_i(1)) = g_i(1)
\end{aligned} \tag{76}$$

Where in the second line, we use that $(1,1]$ represents the empty set, so the term in the expectation is always zero.

Now, we handle the remaining case of $z \in [0, (q-1)/q]_{(q)}$:

$$
\begin{aligned}
&\Pr_i^{\text{split}}(z) \\
&= \Pr_{\mathcal{D}_i}(\Lambda_i \leq z) + z \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (z,1])}}{\Lambda_i}\right] \\
&= \sum_{j=1}^{qz} \Pr\left(\Lambda_i = \frac{j}{q}\right) + z\left[\sum_{j=qz+1}^{q-1} \Pr\left(\Lambda_i = \frac{j}{q}\right) \cdot \frac{q}{j} + \Pr(\Lambda = 1)\frac{1}{1}\right] \\
&= \sum_{j=1}^{qz} -j\left[g_i\left(\frac{j-1}{q}\right) + g_i\left(\frac{j+1}{q}\right) - 2g_i\left(\frac{j}{q}\right)\right] \\
&+ z\left[\sum_{j=qz+1}^{q-1} -j\left[g_i\left(\frac{j-1}{q}\right) + g_i\left(\frac{j+1}{q}\right) - 2g_i\left(\frac{j}{q}\right)\right] \cdot \frac{q}{j} + \ q\left[g_i(1) - g_i\left(\frac{q-1}{q}\right)\right]\right] \\
&= \sum_{j=1}^{qz} -j\left[g_i\left(\frac{j-1}{q}\right) + g_i\left(\frac{j+1}{q}\right) - 2g_i\left(\frac{j}{q}\right)\right] \\
&+ qz\left[\sum_{j=qz+1}^{q-1} -\left[g_i\left(\frac{j-1}{q}\right) + g_i\left(\frac{j+1}{q}\right) - 2g_i\left(\frac{j}{q}\right)\right] + \left[g_i(1) - g_i\left(\frac{q-1}{q}\right)\right]\right] \\
&= -\sum_{j=0}^{qz-1}(j+1)g_i\left(\frac{j}{q}\right) - \sum_{j=2}^{qz+1}(j-1)g_i\left(\frac{j}{q}\right) + 2\sum_{j=1}^{qz} j g_i\left(\frac{j}{q}\right) \\
&+ qz\left[-\sum_{j=qz}^{q-2} g_i\left(\frac{j}{q}\right) - \sum_{j=qz+2}^{q} g_i\left(\frac{j}{q}\right) + 2\sum_{j=qz+1}^{q-1} g_i\left(\frac{j}{q}\right) + g_i(1) - g_i\left(\frac{q-1}{q}\right)\right] \\
&= \sum_{j=2}^{qz-2}(2j - (j+1) - (j-1))g_i\left(\frac{j}{q}\right) - g_i(0) + (2-2)g_i\left(\frac{1}{q}\right) + (2qz - qz + 1)g_i(z) - qz g_i\left(\frac{qz+1}{q}\right) \\
&+ qz\left[(2-1-1)\sum_{j=qz+2}^{q-2} g_i\left(\frac{j}{q}\right) - g_i(z) + (2-1)g_i\left(\frac{qz+1}{q}\right) + (2-1)g_i\left(\frac{q-1}{q}\right) - g_i(1) + g_i(1) - g_i\left(\frac{q-1}{q}\right)\right] \\
&= (qz+1)g_i(z) - qz g_i\left(\frac{qz+1}{q}\right) \\
&+ qz\left[-g_i(z) + g_i\left(\frac{qz+1}{q}\right)\right] \\
&= g_i(z)
\end{aligned}
\tag{77}
$$

Where we use the fact that $g_i(0) = 0$ in the second to last line.

Now we have that $\Pr_i^{\text{split}}(z) = g_i(z)\ \forall z \in [0,1]_{(q)}$, as desired.

□

B Drawbacks of the "Global Λ" Method

In the main text, we briefly discuss using a global value for Λ in order to help with derandomization, as follows:

$$
\begin{aligned}
&\Lambda \sim \mathcal{D}_{\cdot} \\
&s_i \sim \mathcal{U}(0, \Lambda) \quad \forall i
\end{aligned}
\tag{78}
$$

There are several issues with this approach. We will focus our discussion on the ℓ_p^p metric, with $\mathcal{D}_{\cdot}$ given as in Corollary 1.

Firstly, notice that if $\alpha > 1$, we have that $\Lambda = \infty$ with a nonzero probability $1 - 1/\alpha$: when $\Lambda = \infty$, then the entire vector $\boldsymbol{x}^{\text{lower}}$ will be the zero vector, and the entire vector $\boldsymbol{x}^{\text{upper}}$ will consist of entirely ones. Then the particular value of $f([0, ..., 0]^T, [1, ..., 1]^T)$ will be weighted with weight $1 - 1/\alpha$, and all other, meaningful values in the ensemble with have a combined weight of $1/\alpha$: the final value of the smoothed function $p_{\mathcal{D},f}$ will the differ from the fixed $f([0, ..., 0]^T, [1, ..., 1]^T)$ only by at most $1/\alpha$ at any point. In other words, we essentially have a 1-Lipschitz function scaled by $1/\alpha$, rather than a $1/\alpha$- Lipschitz function.[2]

However, even in the $\alpha = 1$ case, the "global Λ" technique still underperforms the method we ultimately propose, as shown in Figure 4 in the main text. One way to understand this is to note that the guarantee provided by this method is unnecessarily tight. In particular, as mentioned in the main text, the global Λ method produces a smoothed function $p_{\mathcal{D},f}$ that is a weighed average of functions which are each $1/\Lambda$-Lipschitz with respect to the ℓ_1 norm, for various values of Λ, by Theorem 1. Let each of these functions be $p_{\Lambda,f}$, so that

$$p_{\mathcal{D},f} \ = \mathop{\mathbb{E}}_{\mathcal{D}}[p_{\Lambda,f}] \tag{79}$$

Note that for each Λ, by the Lipschitz guarantee and $[0, 1]$ bounds on the range:

$$p_{\Lambda,f}(\boldsymbol{x}) - p_{\Lambda,f}(\boldsymbol{y}) \leq \min\left(\frac{\|\boldsymbol{x} - \boldsymbol{y}\|_1}{\Lambda}, 1\right) \tag{80}$$

However, note that:

$$\begin{aligned}
p_{\mathcal{D},f}(\boldsymbol{x}) - p_{\mathcal{D},f}(\boldsymbol{y}) = \\
\mathop{\mathbb{E}}_{\Lambda\sim\mathcal{D}}[p_{\Lambda,f}(\boldsymbol{x}) - p_{\Lambda,f}(\boldsymbol{y})] \leq \\
\mathop{\mathbb{E}}_{\Lambda\sim\mathcal{D}}\left[\min\left(\frac{\|\boldsymbol{x} - \boldsymbol{y}\|_1}{\Lambda}, 1\right)\right] = \\
\mathop{\mathbb{E}}_{\Lambda\sim\mathcal{D}}\left[\frac{\|\boldsymbol{x} - \boldsymbol{y}\|_1}{\Lambda} \cdot \mathbf{1}_{\Lambda > \|\boldsymbol{x}-\boldsymbol{y}\|_1}\right] + \mathop{\mathbb{E}}_{\Lambda\sim\mathcal{D}}[1 \cdot \mathbf{1}_{\Lambda \leq \|\boldsymbol{x}-\boldsymbol{y}\|_1}] = \\
\mathop{\mathbb{E}}_{\Lambda\sim\mathcal{D}}\left[\frac{\|\boldsymbol{x} - \boldsymbol{y}\|_1}{\Lambda} \cdot \mathbf{1}_{\Lambda > \|\boldsymbol{x}-\boldsymbol{y}\|_1}\right] + \mathop{\mathbb{E}}_{\Lambda\sim\mathcal{D}}[1 \cdot \mathbf{1}_{\Lambda \leq \|\boldsymbol{x}-\boldsymbol{y}\|_1}] = \\
\|\boldsymbol{x} - \boldsymbol{y}\|_1 \mathop{\mathbb{E}}_{\mathcal{D}}\left[\frac{\mathbf{1}_{(\Lambda \in (\|\boldsymbol{x}-\boldsymbol{y}\|_1, 1])}}{\Lambda}\right] + \Pr_{\mathcal{D}}(\Lambda \leq \|\boldsymbol{x} - \boldsymbol{y}\|_1) = \Pr^{\text{split}}_{\mathcal{D}}(\|\boldsymbol{x} - \boldsymbol{y}\|_1)
\end{aligned} \tag{81}$$

Where $\Pr^{\text{split}}_{\mathcal{D}}$ is defined in terms of $\mathcal{D}_\cdot$ exactly as in Theorem 3-a. Then, by the mechanics of Theorem 3-c and from the construction of $\mathcal{D}_\cdot$, we have:

$$p_{\mathcal{D},f}(\boldsymbol{x}) - p_{\mathcal{D},f}(\boldsymbol{y}) \leq \Pr^{\text{split}}_{\mathcal{D}}(\|\boldsymbol{x} - \boldsymbol{y}\|_1) = g_\cdot(\|\boldsymbol{x} - \boldsymbol{y}\|_1) \tag{82}$$

In the case of ℓ_p^p metrics with $p < 1$, this means:

$$p_{\mathcal{D},f}(\boldsymbol{x}) - p_{\mathcal{D},f}(\boldsymbol{y}) \leq \frac{\|\boldsymbol{x} - \boldsymbol{y}\|_1^p}{\alpha} \tag{83}$$

But note that:

$$p_{\mathcal{D},f}(\boldsymbol{x}) - p_{\mathcal{D},f}(\boldsymbol{y}) \leq \frac{\|\boldsymbol{x} - \boldsymbol{y}\|_1^p}{\alpha} \leq \frac{\|\boldsymbol{x} - \boldsymbol{y}\|_p^p}{\alpha} \tag{84}$$

In other words, we are imposing a tighter guarantee than necessary, which depends only on the ℓ_1 distance between $\boldsymbol{x}$ and $\boldsymbol{y}$: the desired ℓ_p^p guarantee is everywhere at least as loose. So, while this technique technically works, it does not really respect the "spirit" of the fractional ℓ_p^p threat model.

C Designing $\mathcal{D}_i$ for Derandomization using Mixed-Integer Linear Programming

As mentioned in Section 5 in the main text, one challenge in the derandomization of our technique is to design a distribution $\mathcal{D}_i$ such that all outcomes (Λ_i, s_i) occur with a probability in the form n/B, where $n \in \mathbb{N}$ is an integer, B is a constant integer, and additionally where:

$$\Pr^{\text{split}}_i(z) \approx g_i(z), \quad \forall z \in [0, 1]_{(q)}. \tag{85}$$

[2]Note that a similar observation was made in Levine and Feizi (2021) about using a global value of s_i for $\Lambda > 1$

However, strictly:

$$\Pr_i^{\text{split}}(z) \leq g_i(z), \quad \forall z \in [0,1]_{(q)}. \tag{86}$$

We first show that we can formulate Equation 85 as a linear program in the case where we allow arbitrary probabilities for each value of Λ, and then show that we can convert it into a MILP to obtain probabilities in the desired form.

Note that we are working with the quantized form of Variable-Λ smoothing: for convenience, we will therefore introduce the variables:

$$g^j := g_i\left(\frac{j}{q}\right) \forall j \in [q] \tag{87}$$

$$v_j := \Pr\left(\Lambda_i = \left(\frac{j}{q}\right)\right) \forall j \in [q] \tag{88}$$

Our distribution $\mathcal{D}_i$ is then defined by the vector $\boldsymbol{v}$: the probability that $\Lambda_i = \infty$ is determined by normalization ($\Pr(\Lambda_i = \infty) = 1 - \Sigma_j v_j$) .

We make Equation 85 rigorous by using the following objective:

$$\begin{aligned} &\text{minimize } \epsilon \text{ such that} \\ &g_i(z) - \epsilon \leq \Pr_i^{\text{split}}(z) \leq g_i(z), \quad \forall z \in [0,1]_{(q)}. \end{aligned} \tag{89}$$

Note that ϵ is a single scalar: we are attempting to achieve uniform convergence. We can write $\Pr_i^{\text{split}}(z)$ in the following form:

$$\begin{aligned} \Pr_i^{\text{split}}(z) = \\ \Pr_{\mathcal{D}_i}(\Lambda_i \leq z) + z \mathop{\mathbb{E}}_{\mathcal{D}_i}\left[\frac{\mathbf{1}_{(\Lambda_i \in (z,1])}}{\Lambda_i}\right] = \\ \sum_{j=1}^{qz} v_j + z \sum_{j=qz+1}^{q} \frac{v_j}{\left(\frac{j}{q}\right)} = \\ \sum_{j=1}^{qz} v_j + qz \sum_{j=qz+1}^{q} \frac{v_j}{j} \end{aligned} \tag{90}$$

Then our optimization becomes (letting $k := qz$):

$$\begin{aligned} &\text{minimize } \epsilon \text{ such that} \\ &g^k - \epsilon \leq \sum_{j=1}^{k} v_j + k \sum_{j=k+1}^{q} \frac{v_j}{j} \leq g^k, \quad \forall k \in [q]. \end{aligned} \tag{91}$$

With additional constraints:

- $v_j \geq 0, \ \forall j \in [q]$ (Probabilities are non-negative)
- $\sum_{j=1}^{q} v_j \leq 1$ (Normalization: recall that additional probability is assigned to $\Lambda = \infty$)
- $\epsilon \geq 0$

This linear program, with variables $\epsilon, \boldsymbol{v}$, completely describes the problem of designing $\mathcal{D}_i$. If $g_i(z)$ is concave (as it should be, by assumption), then this LP always has an optimal $\epsilon = 0$ solution, given in Theorem 3-c. (See the proof of that theorem in Appendix A.4).

However, we now want all outcomes to have probabilities in the form n/B. Note that for $\Lambda_i = j/q$, there are j outcomes for s_i, each of which must have equal probabilities. We therefore need $\Lambda_i = j/q$ to occur with a probability in the form $\frac{nj}{B}$, for some integer n. We will then re-scale our parameters:

$$w_j := \frac{Bv_j}{j} \;\; \forall j \in [q] \tag{92}$$

Our optimization now becomes:

$$\begin{aligned}
&\text{minimize } \epsilon \text{ such that} \\
&g^k - \epsilon \leq \sum_{j=1}^{k} \frac{j \cdot w_j}{B} + k \sum_{j=k+1}^{q} \frac{w_j}{B} \leq g^k, \quad \forall k \in [q]. \\
&w_j \in \mathbb{N} \\
&\sum_{j=1}^{q} j \cdot w_j \leq B \\
&\epsilon \geq 0
\end{aligned} \tag{93}$$

This is a mixed-integer linear program, with variables $\epsilon, \boldsymbol{w}$. Once solved, the desired distribution over Λ can be read off from $\boldsymbol{w}$. In practice, when using this method with $g_i(z) = z^p/\alpha$, we only solved the MILP directly for $\alpha = 1$, using budget $B = 1000$: for larger α, we used the fact that Equation 90 is linear in $\boldsymbol{v}$ to simply scale down $\Pr_{\mathcal{D}}^{\text{split}}(z)$ by scaling up B as $B = 1000\alpha$, without changing the integer allocations of $\boldsymbol{w}$: in practice, this just means adding additional $\Lambda_i = \infty$ outcomes to the list of possible outcomes that are uniformly selected from. Also, rather than optimizing over ϵ, we held ϵ constant at 0.02, so that the problem became a feasibility problem, rather than an optimization problem. The results are shown in Figure 5 in the main text. Each of the two MILPs took ≈ 10 minutes or less to solve.

We can show that, with sufficiently large budget, arbitrarily close approximations can always be made. In particular, consider using the optimal real-valued solution from Theorem 3-c, and the simply rounding each w_j down to integers. Because the coefficients on w_j's in Equation 93 are all non-negative, the upper-bounds on these terms will all still be met. The only lower-bound, the $g^k - \epsilon$ term, will remain feasible because ϵ can be made arbitrarily large. Therefore, this rounding technique will not break feasibility. Now, let's look at optimality. Let $\tilde{w}_j$'s be the real-valued, optimal solutions, and w_j's be the rounded solution. Then we have:

$$g^k - \epsilon \leq \sum_{j=1}^{k} \frac{j \cdot w_j}{B} + k \sum_{j=k+1}^{q} \frac{w_j}{B} \leq \sum_{j=1}^{k} \frac{j \cdot \tilde{w}_j}{B} + k \sum_{j=k+1}^{q} \frac{\tilde{w}_j}{B} = g^k, \quad \forall k \in [q]. \tag{94}$$

The tightest lower-bound on epsilon will be the constraint where:

$$\epsilon = \left(\sum_{j=1}^{k} \frac{j \cdot \tilde{w}_j}{B} + k \sum_{j=k+1}^{q} \frac{\tilde{w}_j}{B} \right) - \left(\sum_{j=1}^{k} \frac{j \cdot w_j}{B} + k \sum_{j=k+1}^{q} \frac{w_j}{B} \right) \tag{95}$$

However, for each j, $\tilde{w}_j - w_j < 1$, so:

$$\epsilon < \left(\sum_{j=1}^{k} \frac{j}{B} + k \sum_{j=k+1}^{q} \frac{1}{B} \right) \leq \sum_{j=1}^{q} \frac{j}{B} = \frac{q^2 + q}{2B} \tag{96}$$

Therefore, with sufficiently large budget B, the error ϵ can be made arbitrarily small.

D Deterministic ℓ_0 Certificates

Consider the following ECM, parameterized by α:

$$g_i(z) := \begin{cases} 0 \text{ if } z = 0 \\ \frac{1}{\alpha} \text{ otherwise} \end{cases} \tag{97}$$

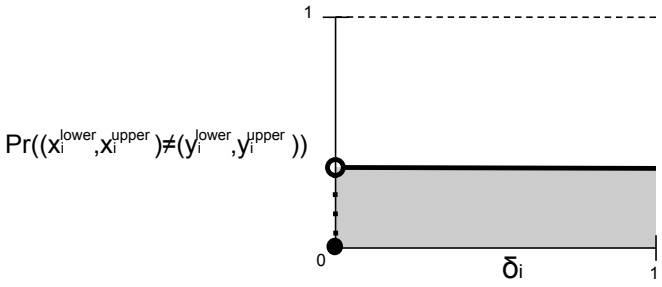

Figure 7: Diagram of the ℓ_0 $g_i(z)$ function.

Note that the resulting metric $d(\boldsymbol{x}, \boldsymbol{y})$ is in fact $\|\boldsymbol{x} - \boldsymbol{y}\|_0/\alpha$. However, because this is not a continuous function, we cannot apply Theorem 2-c directly. However, if we still want $\Pr_i^{\text{split}}(z) = g_i(z)$, we have two options:

- Option 1: expand the support of $\mathcal{D}_i$ to include $\Lambda_i = 0$, where, if $\Lambda_i = 0$, then $x_i^{\text{lower}} = x_i^{\text{upper}} = x_i$. We can then distribute Λ_i as:

$$\Lambda_i = \begin{cases} 0 \text{ with prob. } \frac{1}{\alpha} \\ \infty \text{ otherwise} \end{cases} \tag{98}$$

  It is easy to verify that in this case, $\Pr_i^{\text{split}}(z) = g_i(z)$. (In particular, if $x_i = y_i$, then $\Pr((x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})) = 0$; otherwise $\Pr((x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}})) = \Pr(\Lambda = 0) = 1/\alpha$. See Figure 7.)

- Option 2: consider the quantized case. Then we can just apply Theorem 3-c directly, yielding

$$\Lambda_i = \begin{cases} 1/q \text{ with prob. } \frac{1}{\alpha} \\ \infty \text{ otherwise} \end{cases} \tag{99}$$

  Note that with $\Lambda = 1/q$, the original value of the pixel is always preserved, with $x_i^{\text{upper}} = x_i + 0.5/q$, $x_i^{\text{lower}} = x_i - 0.5/q$.

In practice, we use Option 2 in our experiments, because we are using quantized image datasets (and for code consistency). However, either option will yield classifiers that $1/\alpha$-Lipschitz with respect to the ℓ_0 metric. If α is an integer (as in our experiments), then we only need $B = \alpha$ smoothing samples: each pixel is preserved ($\Lambda = 1/q$) in exactly one sample, and is ablated ($\Lambda = \infty$) in the other $\alpha - 1$ samples. The choice of which pixels to retain in which samples should be arbitrary, but should remain fixed throughout training and testing. (This is a direct application of the "fixed offset" method mentioned in the paper).

In practice, this produces an algorithm which is very similar to the "randomized ablation" randomized ℓ_0 certificate proposed in Levine and Feizi (2020a): in both techniques, we are retaining some pixels unchanged while completely removing information about other pixels. In fact, this deterministic variant of "randomized ablation" was already proposed to provide provable robustness against poisoning attacks in Levine and Feizi (2020b): in particular, the technique proposed for label-flipping poisoning attacks is basically identical, with the features being training-data labels rather than pixels: the idea is to train α separate models, each using a disjoint arbitrary subset of labels, and then take the consensus output at test time. Levine and Feizi (2020b) note that the certificate is looser than that of Levine and Feizi (2020a), due to the use of a union bound, however there are added benefits of determinism and using only a small number of smoothing samples (Levine and Feizi (2020a) uses 11,000 smoothing samples (1000 for prediction and 10,000 for bounding); in the case of Levine and Feizi (2020b), each "smoothing sample" requires training a classifier).

Note that, on image data, there are two somewhat different definitions of "ℓ_0 adversarial attack" which are often used: true ℓ_0 attacks in the space of features, where each feature is a single color channel of a pixel value, and "sparse" attacks, where the attack magnitude signifies the number of pixel positions modified, but potentially all channels may be affected. Our method can be applied in both situations: to certify for "sparse" attacks, simply

insure that $\Lambda_i = \Lambda_j$ if features i, j are channels of the same pixel: then $\Pr((x_i^{\text{lower}}, x_i^{\text{upper}}) \neq (y_i^{\text{lower}}, y_i^{\text{upper}}) \cup (x_j^{\text{lower}}, x_j^{\text{upper}}) \neq (y_j^{\text{lower}}, y_j^{\text{upper}})) \leq 1/\alpha$. In Figure 8, we compare the certificates generated by this deterministic "sparse" certificate to the results of Levine and Feizi (2020a). While the reported certificates are somewhat worse, particularly on ImageNet, note that these are exact, rather than probabilistic certificates, and furthermore that the number of forward-passes required to certify is significantly reduced, leading to reduced certification times. For example, on ImageNet, the most computationally-intensive certification for the deterministic method used 100 forward-passes, and averaged 0.13 seconds / image for certification using a single GPU. By contrast, each randomized certification from Levine and Feizi (2020a) averaged 16 seconds, using four GPUs (note that this is around four times less efficient than expected, compared to the proposed derandomized method, based on the number of smoothing samples alone: other implementation differences must also be at play). We also provide certificates for "ℓ_0" attacks, which Levine and Feizi (2020a) do not test.

E Explicit Certification Procedure

In order to use our ℓ_p^p Lipschitz guarantee to generate ℓ_p- norm certificates, we follow a procedure similar to the ℓ_1 certificate from Levine and Feizi (2021). Concretely, for each class c, let $p_c(\boldsymbol{x})$ be the smoothed, $1/\alpha$-ℓ_p^p-Lipschitz logit function that our algorithm produces. In our implementation, we have the *base* classifier f output "hard" classifications: $f_c(\boldsymbol{x}) = 1$ if the base classifier classifies $\boldsymbol{x}$ into class c, and zero otherwise. Therefore $p_c(\boldsymbol{x})$ can also be though of as the fraction of base classifier outputs with value c.

If two points $\boldsymbol{x}, \boldsymbol{y}$ differ by at most δ in the ℓ_p "norm", then they must differ by at most δ^p in the ℓ_p^p metric. Then by Lipschitz property, we have:

$$|p_c(\boldsymbol{x}) - p_c(\boldsymbol{y})| \leq \frac{\delta^p}{\alpha} \tag{100}$$

Now, assume that $\boldsymbol{x}$ is classified as class c by the smoothed classifier ($c = \arg\max_{c'} p_{c'}(\boldsymbol{x})$). Let c' be any other class. By algebra, we have:

$$p_c(\boldsymbol{x}) - p_{c'}(\boldsymbol{x}) - |p_c(\boldsymbol{x}) - p_c(\boldsymbol{y})| - |p_{c'}(\boldsymbol{x}) - p_{c'}(\boldsymbol{y})| \leq p_c(\boldsymbol{y}) - p_{c'}(\boldsymbol{y}) \tag{101}$$

Therefore, using Equation 100, we have:

$$p_c(\boldsymbol{x}) - p_{c'}(\boldsymbol{x}) - \frac{2\delta^p}{\alpha} \leq p_c(\boldsymbol{y}) - p_{c'}(\boldsymbol{y}) \tag{102}$$

Then:

$$\left(\frac{\alpha}{2}(p_c(\boldsymbol{x}) - p_{c'}(\boldsymbol{x}))\right)^{1/p} \geq \delta \implies p_c(\boldsymbol{y}) \geq p_{c'}(\boldsymbol{y}) \tag{103}$$

This means that the class is guaranteed not to change to c' within an ℓ_p radius of $\left(\frac{\alpha}{2}(p_c(\boldsymbol{x}) - p_{c'}(\boldsymbol{x}))\right)^{1/p}$ of $\boldsymbol{x}$. Computing the minimum of this quantity over all classes $c' \neq c$ gives the certified radius.

The above argument ignores the equality case: at radius δ, the two class probabilities may still be equal, leading to an unclear classification result. To deal with this, we borrow a trick originally from Levine and Feizi (2020c) (also used by Levine and Feizi (2021)). Specifically, at classification time, we break ties deterministically using the class index: if $p_c(\boldsymbol{x}) = p_{c'}(\boldsymbol{x})$ and $c < c'$ then the class c will be the final classification. In the case that $c < c'$, then $p_c(\boldsymbol{y}) \geq p_{c'}(\boldsymbol{y})$ is a sufficient condition to ensure that class c is chosen, so we can certify that the class c will be chosen at all points up to and including radius $\delta = \left(\frac{\alpha}{2}(p_c(\boldsymbol{x}) - p_{c'}(\boldsymbol{x}))\right)^{1/p}$.

To deal with the other case, $c' < c$, we subtract any positive ϵ from both sides of Equation 102:

$$p_c(\boldsymbol{x}) - p_{c'}(\boldsymbol{x}) - \epsilon - \frac{2\delta^p}{\alpha} \leq p_c(\boldsymbol{y}) - p_{c'}(\boldsymbol{y}) - \epsilon \tag{104}$$

$$\left(\frac{\alpha}{2}(p_c(\boldsymbol{x}) - p_{c'}(\boldsymbol{x}) - \epsilon)\right)^{1/p} \geq \delta \implies p_c(\boldsymbol{y}) \geq p_{c'}(\boldsymbol{y}) + \epsilon \implies p_c(\boldsymbol{y}) > p_{c'}(\boldsymbol{y}) \tag{105}$$

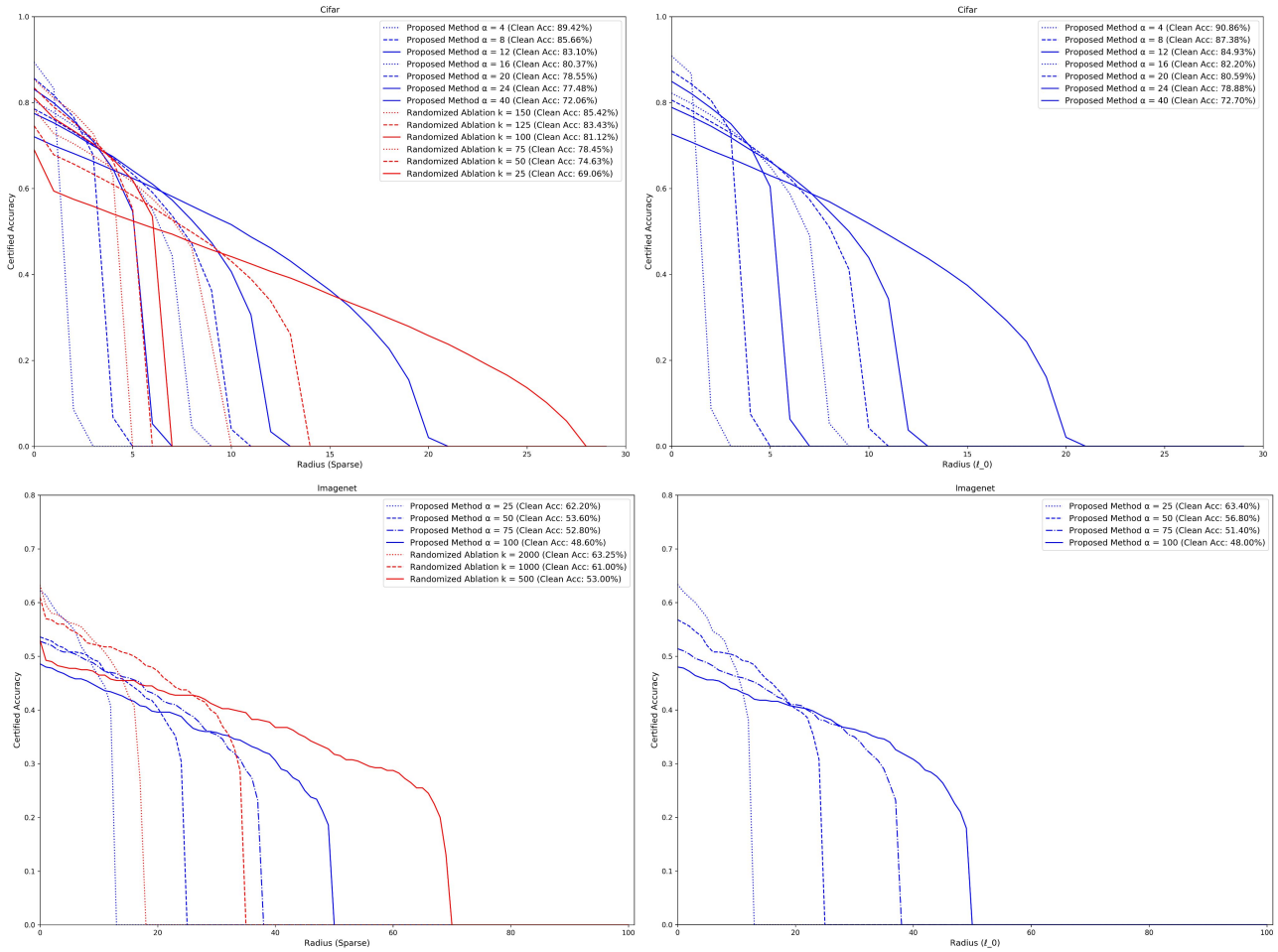

Figure 8: Sparse and ℓ_0 certification results, on CIFAR-10 (top) and ImageNet (bottom). In the left column, we compare to Levine and Feizi (2020a), a randomized method, where certificates were reported with 95% confidence. Results are directly from that work: note that training times, model architectures, and parameters may differ, in addition to the smoothing method. On ImageNet, we use a subset of 500 images from the validation set; the results from Levine and Feizi (2020a) are using a different random subset of 400 validation images, so this may cause some variance. The parameter k is the number of pixels retained in each image in Levine and Feizi (2020a): because ImageNet has 50176 pixels, the fraction of retained pixels is roughly 50000/k, which functionally corresponds to α in our model: it is the appropriate to compare $k = 2000$ with $\alpha = 25$, etc. For CIFAR-10, there are 1024 pixels, so a similar heuristic of $\alpha \approx 1000/k$ can be used. In the right column, we show certificates for ℓ_0 attacks, where the attack budget represents the number of individual pixel channels, rather than whole pixels, attacked. Levine and Feizi (2020a) did not test for this threat model.

In our deterministic certification implementation, we use $\epsilon := 1/B$, where B is the number of (nonrandom) smoothing samples: this is the smallest difference possible between two values of $p_{cdot}(\cdot)$. Combining the two cases, we get the final form of our certificate:

$$\min_{c':c'\neq c}\left[\left(\frac{\alpha}{2}\left(p_c(\boldsymbol{x})-p_{c'}(\boldsymbol{x})-\frac{\mathbf{1}_{c'<c}}{B}\right)\right)^{1/p}\right]\geq\delta \implies \boldsymbol{y} \text{ is assigned class } c \quad \forall \boldsymbol{x},\boldsymbol{y},\ \|\boldsymbol{x}-\boldsymbol{y}\|_p\leq\delta \tag{106}$$

F Representations of Inputs

As we stated in the main text, we modified the architectures used for f in order to accept both inputs $(\boldsymbol{x}^{\text{lower}}, \boldsymbol{x}^{\text{upper}})$, by doubling the number of input channels in the first layer. We tried a variety of alternative methods as well on CIFAR-10 for p=1/2:

- 'Center': using only a single input $\frac{\boldsymbol{x}^{\text{lower}}+\boldsymbol{x}^{\text{upper}}}{2}$, as in Levine and Feizi (2021), but with variable-Λ smoothing.
- 'Center Center': Same as 'Center', but with channels duplicated. This acted as an ablation study, to isolate the effect of the additional information of having both channels from the mere increase in network parameters from doubling the number of channels.
- 'Center Error': Channels are $\frac{\boldsymbol{x}^{\text{lower}}+\boldsymbol{x}^{\text{upper}}}{2}$ and $\frac{\boldsymbol{x}^{\text{upper}}-\boldsymbol{x}^{\text{lower}}}{2}$.
- 'Upper Lower': Channels are $\boldsymbol{x}^{\text{upper}}$ and $\boldsymbol{x}^{\text{lower}}$. This is the method presented the main text, and used in other experiments.

See Table 2 for results. As would be anticipated, the general trend was:

$$\text{L\&F (2021)} < \text{'Center'} \approx \text{'Center Center'} < \text{'Center Error'} \approx \text{'Upper Lower'} \tag{107}$$

This tells us that Variable-Λ smoothing has an advantage over Levine and Feizi (2021) for p=1/2 certification, even if only the center of the interval is given to the base classifier. However, having full knowledge of the range of the interval clearly provides an added benefit.

G Effect of Pseudorandom Seed Value

As mentioned in the main text, we use cyclic permutations with pseudorandom offsets to generate the coupled distribution of $\mathcal{D}$, using a seed value of 0. In Table 3, we compare alternate choices of seed values for CIFAR-10 with $p = 1/2$. Note that the seed value has very little effect on the certified accuracy: certified accuracies are within 1 percentage point of each other. Similar conclusions about the effect of the seed hyperparameter were found in Levine and Feizi (2021) for the ℓ_1 case.

H Effect of Cyclic Permutations vs. Arbitrary Permutations

In the main text, we mention that Theorem 3 allows for the use of arbitrary permutations in defining the coupling for the distribution $\mathcal{D}$. However, in practice, we choose to use only cyclic permutations of a single list of outcomes. This is because using arbitrary permutations involves storing in memory the complete permutation (each consisting of B outcomes, with up to $B = 18,000$ in our experiments) for *each* dimension. This does not scale efficiently to higher-dimensional problems. On CIFAR-10 with $p = 1/2$, we did attempt this arbitrary permutation method, using pseudo-randomly generated arbitrary permutations for each dimension. Results are found in Table 4: in general, we find no major benefit to using arbitrary permutations.

I Complete Certification Results on CIFAR-10

In Figures 9 and 10, we show the complete certification results for all models used in Table 1 in the main text. Note that our method dominates at every noise level, except when $\alpha = 1$: this is because when $\alpha = 1$, the

ρ	10	20	30	40	50	60	70	80
L&F (2021) (From ℓ_1)	42.69% (60.42% @ α=18)	35.04% (60.42% @ α=18)	28.89% (60.42% @ α=18)	23.46% (60.42% @ α=18)	18.81% (60.42% @ α=18)	13.76% (60.42% @ α=18)	8.38% (60.42% @ α=18)	1.27% (60.42% @ α=18)
L&F (2021) (From ℓ_1) (Stab. Training)	41.32% (55.38% @ α=12)	35.56% (50.11% @ α=18)	32.07% (50.11% @ α=18)	28.70% (50.11% @ α=18)	24.95% (50.11% @ α=18)	20.79% (50.11% @ α=18)	16.20% (50.11% @ α=18)	6.98% (50.11% @ α=18)
Center	49.83% (68.35% @ α=15)	42.26% (65.59% @ α=18)	36.54% (65.59% @ α=18)	31.10% (65.59% @ α=18)	25.65% (65.59% @ α=18)	19.93% (65.59% @ α=18)	13.53% (65.59% @ α=18)	2.68% (65.59% @ α=18)
Center (Stab. Training)	47.98% (64.31% @ α=9)	42.27% (56.96% @ α=15)	38.47% (54.79% @ α=18)	35.31% (54.79% @ α=18)	31.91% (54.79% @ α=18)	28.17% (54.79% @ α=18)	23.10% (54.79% @ α=18)	11.82% (54.79% @ α=18)
Center Center	49.78% (66.06% @ α=18)	42.15% (66.06% @ α=18)	36.15% (66.06% @ α=18)	31.17% (66.06% @ α=18)	25.49% (66.06% @ α=18)	19.87% (66.06% @ α=18)	13.21% (66.06% @ α=18)	2.55% (66.06% @ α=18)
Center Center (Stab. Training)	48.33% (60.17% @ α=12)	42.24% (54.83% @ α=18)	38.84% (54.83% @ α=18)	35.59% (54.83% @ α=18)	32.28% (54.83% @ α=18)	28.11% (54.83% @ α=18)	23.16% (54.83% @ α=18)	11.62% (54.83% @ α=18)
Center Error	56.66% (75.80% @ α=12)	49.61% (70.56% @ α=18)	43.50% (70.56% @ α=18)	37.76% (70.56% @ α=18)	32.26% (70.56% @ α=18)	25.80% (70.56% @ α=18)	18.51% (70.56% @ α=18)	4.99% (70.56% @ α=18)
Center Error (Stab. Training)	55.58% (69.99% @ α=9)	48.73% (63.02% @ α=15)	**45.08%** (60.49% @ α=18)	41.86% (60.49% @ α=18)	38.31% (60.49% @ α=18)	34.39% (60.49% @ α=18)	28.98% (60.49% @ α=18)	**16.45%** (60.49% @ α=18)
Upper Lower	**56.74%** (73.22% @ α=15)	**49.80%** (70.57% @ α=18)	43.60% (70.57% @ α=18)	37.97% (70.57% @ α=18)	32.37% (70.57% @ α=18)	25.83% (70.57% @ α=18)	18.19% (70.57% @ α=18)	5.02% (70.57% @ α=18)
Upper Lower (Stab. Training)	55.21% (69.87% @ α=9)	48.72% (62.74% @ α=15)	45.05% (60.44% @ α=18)	**42.26%** (60.44% @ α=18)	**38.62%** (60.44% @ α=18)	**34.42%** (60.44% @ α=18)	**29.01%** (60.44% @ α=18)	16.28% (60.44% @ α=18)

Table 2: Comparison of $\ell_{1/2}$ CIFAR-10 certificates for a variety of noise representations. See text of Appendix F.

	$\ell_{1/2}$							
ρ	10	20	30	40	50	60	70	80
Seed: 0	56.74% (73.22% @ α=15)	49.80% (70.57% @ α=18)	43.60% (70.57% @ α=18)	37.97% (70.57% @ α=18)	32.37% (70.57% @ α=18)	25.83% (70.57% @ α=18)	18.19% (70.57% @ α=18)	5.02% (70.57% @ α=18)
Seed: 1	56.64% (73.17% @ α=15)	49.28% (70.14% @ α=18)	43.94% (70.14% @ α=18)	38.53% (70.14% @ α=18)	32.61% (70.14% @ α=18)	26.12% (70.14% @ α=18)	18.43% (70.14% @ α=18)	5.25% (70.14% @ α=18)
Seed: 2	56.60% (73.10% @ α=15)	49.35% (70.44% @ α=18)	43.60% (70.44% @ α=18)	38.01% (70.44% @ α=18)	32.10% (70.44% @ α=18)	25.80% (70.44% @ α=18)	18.52% (70.44% @ α=18)	4.70% (70.44% @ α=18)
Seed: 3	56.80% (72.77% @ α=15)	49.71% (70.74% @ α=18)	43.77% (70.74% @ α=18)	38.30% (70.74% @ α=18)	32.18% (70.74% @ α=18)	25.95% (70.74% @ α=18)	18.04% (70.74% @ α=18)	5.01% (70.74% @ α=18)
Seed: 4	56.82% (73.09% @ α=15)	49.70% (70.56% @ α=18)	43.64% (70.56% @ α=18)	37.79% (70.56% @ α=18)	32.09% (70.56% @ α=18)	25.98% (70.56% @ α=18)	18.54% (70.56% @ α=18)	5.04% (70.56% @ α=18)
Seed: 0 (Stab Training)	55.21% (69.87% @ α=9)	48.72% (62.74% @ α=15)	45.05% (60.44% @ α=18)	42.26% (60.44% @ α=18)	38.62% (60.44% @ α=18)	34.42% (60.44% @ α=18)	29.01% (60.44% @ α=18)	16.28% (60.44% @ α=18)
Seed: 1 (Stab Training)	55.81% (69.84% @ α=9)	48.67% (62.51% @ α=15)	44.43% (60.07% @ α=18)	41.45% (60.07% @ α=18)	38.16% (60.07% @ α=18)	34.17% (60.07% @ α=18)	28.82% (60.07% @ α=18)	16.10% (60.07% @ α=18)
Seed: 2 (Stab Training)	55.18% (69.83% @ α=9)	48.52% (62.80% @ α=15)	44.77% (60.13% @ α=18)	41.38% (60.13% @ α=18)	38.03% (60.13% @ α=18)	34.02% (60.13% @ α=18)	28.81% (60.13% @ α=18)	16.08% (60.13% @ α=18)
Seed: 3 (Stab Training)	55.99% (70.27% @ α=9)	48.64% (62.77% @ α=15)	45.16% (60.02% @ α=18)	41.98% (60.02% @ α=18)	38.60% (60.02% @ α=18)	34.61% (60.02% @ α=18)	29.13% (60.02% @ α=18)	16.60% (60.02% @ α=18)
Seed: 4 (Stab Training)	56.10% (69.87% @ α=9)	48.59% (62.90% @ α=15)	44.76% (60.30% @ α=18)	41.53% (60.30% @ α=18)	38.19% (60.30% @ α=18)	34.17% (60.30% @ α=18)	28.81% (60.30% @ α=18)	16.00% (60.30% @ α=18)

Table 3: Certified accuracy as a function of fractional ℓ_p distance ρ, for $p = 1/2$ on CIFAR-10, using various values of the seed for pseudo-random generation of cyclic permutations for $\mathcal{D}$. We test with $\alpha = \{1, 3, 6, 9, 12, 15, 18\}$ where $1/\alpha$ is the Lipschitz constant of the model, and report the highest certificate for each technique over all of the models. In parentheses, we report the the clean accuracy and the α parameter for the associated model.

	$\ell_{1/2}$							
	10	20	30	40	50	60	70	80
Cyclic Perm.	56.74% (73.22% @ α=15)	49.80% (70.57% @ α=18)	43.60% (70.57% @ α=18)	37.97% (70.57% @ α=18)	32.37% (70.57% @ α=18)	25.83% (70.57% @ α=18)	18.19% (70.57% @ α=18)	5.02% (70.57% @ α=18)
Cyclic Perm. (Stab. Training)	55.21% (69.87% @ α=9)	48.72% (62.74% @ α=15)	45.05% (60.44% @ α=18)	42.26% (60.44% @ α=18)	38.62% (60.44% @ α=18)	34.42% (60.44% @ α=18)	29.01% (60.44% @ α=18)	16.28% (60.44% @ α=18)
Arbitrary Perm.	56.85% (72.90% @ α=15)	49.62% (70.56% @ α=18)	43.74% (70.56% @ α=18)	38.12% (70.56% @ α=18)	32.08% (70.56% @ α=18)	25.94% (70.56% @ α=18)	18.29% (70.56% @ α=18)	4.70% (70.56% @ α=18)
Arbitrary Perm. (Stab. Training)	55.56% (70.28% @ α=9)	48.56% (62.73% @ α=15)	44.60% (60.08% @ α=18)	41.46% (60.08% @ α=18)	38.13% (60.08% @ α=18)	34.39% (60.08% @ α=18)	28.93% (60.08% @ α=18)	16.26% (60.08% @ α=18)

Table 4: Certified accuracy as a function of fractional ℓ_p distance ρ, for $p = 1/2$ on CIFAR-10, using either pseudorandom cyclic permutations (as in the main text) or psuedorandom arbitrary permutations. We test with $\alpha = \{1, 3, 6, 9, 12, 15, 18\}$ where $1/\alpha$ is the Lipschitz constant of the model, and report the highest certificate for each technique over all of the models. In parentheses, we report the the clean accuracy and the α parameter for the associated model.

α	ℓ_1 (L&F 2021)	ℓ_1 (L&F 2021) (Stability)	$\ell_{1/2}$	$\ell_{1/2}$ (Stability)	$\ell_{1/3}$	$\ell_{1/3}$ (Stability)
1	83.98%	82.12%	90.16%	88.74%	92.14%	90.84%
3	74.38%	70.89%	83.51%	81.47%	86.35%	84.72%
6	65.68%	61.51%	76.38%	73.51%	79.91%	77.73%
9	59.82%	55.68%	70.97%	67.15%	75.23%	71.82%
12	55.48%	51.68%	66.70%	62.86%	71.42%	67.59%
15	52.21%	48.84%	63.66%	59.51%	67.86%	64.03%
18	49.54%	46.13%	60.75%	56.87%	65.39%	61.25%

Table 5: Base classifier accuracies on CIFAR-10. Note that as p decreases, the base classifier accuracy increases for a fixed value of α: this leads to larger certificates.

maximum possible certificate using our method is $(1/2)^{1/p}$, while it is $1/2$ using equivalence of norms from an ℓ_1 certificate. However, this is largely irrelevant, because we show that by selecting larger values of the hyperparameter α, we are able to achieve consistently larger certificates.

In Table 5, we provide the *base* classifier accuracies for the models. Note that at large α, the form of the certificates using our method, and using ℓ_1 certificates through norm conversion, are essentially the same: both are (roughly):

$$\min_{c':c'\neq c} \left[\left(\frac{\alpha}{2} \left(p_c(\boldsymbol{x}) - p_{c'}(\boldsymbol{x}) \right) \right)^{1/p} \right] \tag{108}$$

where $p_c(\boldsymbol{x})$ is the fraction of the smoothing samples on which the base classifier returns the class c (see Appendix E and Section 6 in the main text for details.) Therefore the success of our technique at producing larger certificates is entirely because the base classifier is more accurate under our fractional-ℓ_p noise than under splitting noise with a fixed $\Lambda = \alpha$.

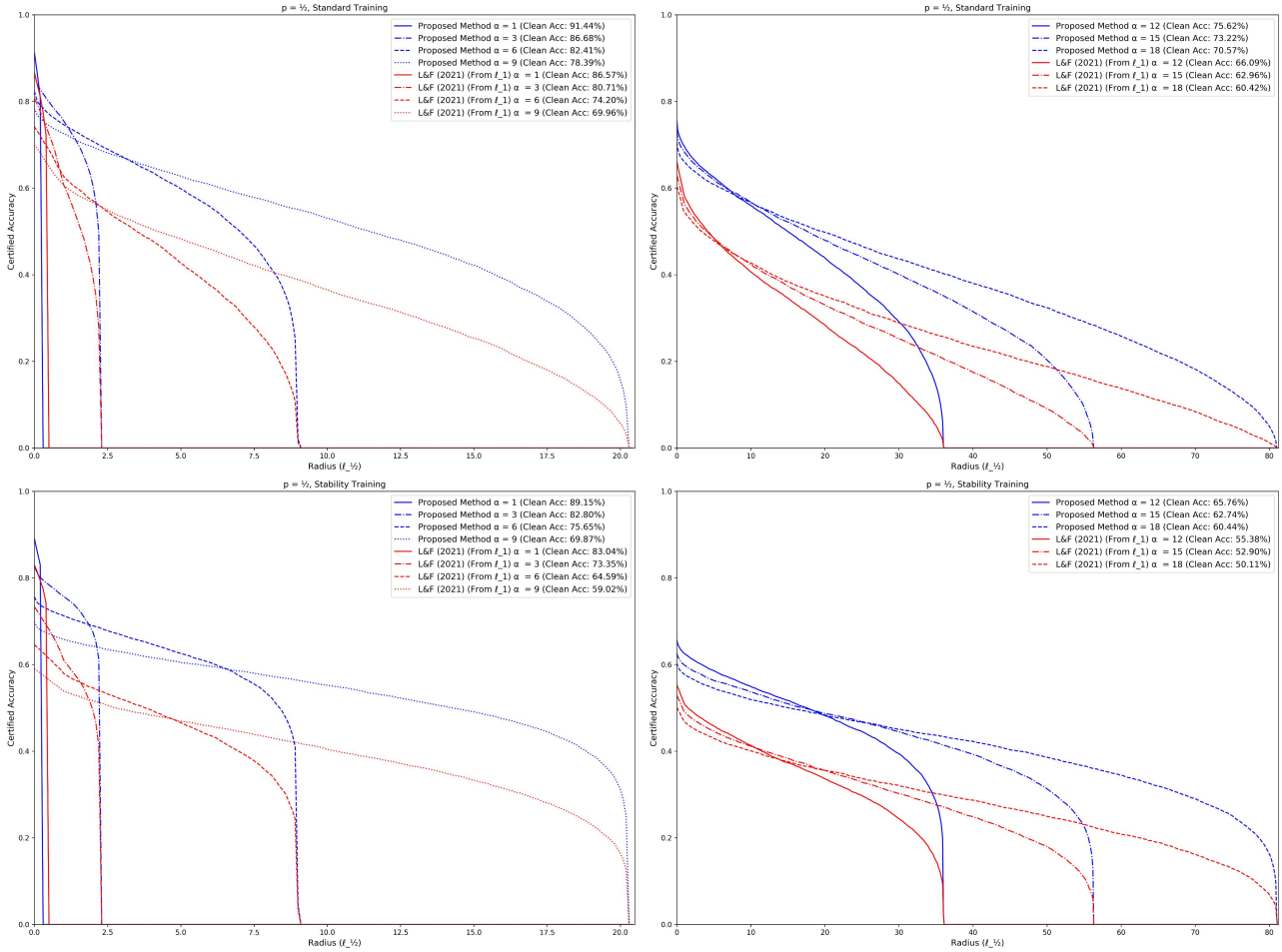

Figure 9: Full certification results for $p = 1/2$ on CIFAR-10. Left column shows $\alpha \in \{1, 3, 6, 9\}$, right column shows $\alpha \in \{12, 15, 18\}$, top row shows standard training, and bottom row shows stability training.

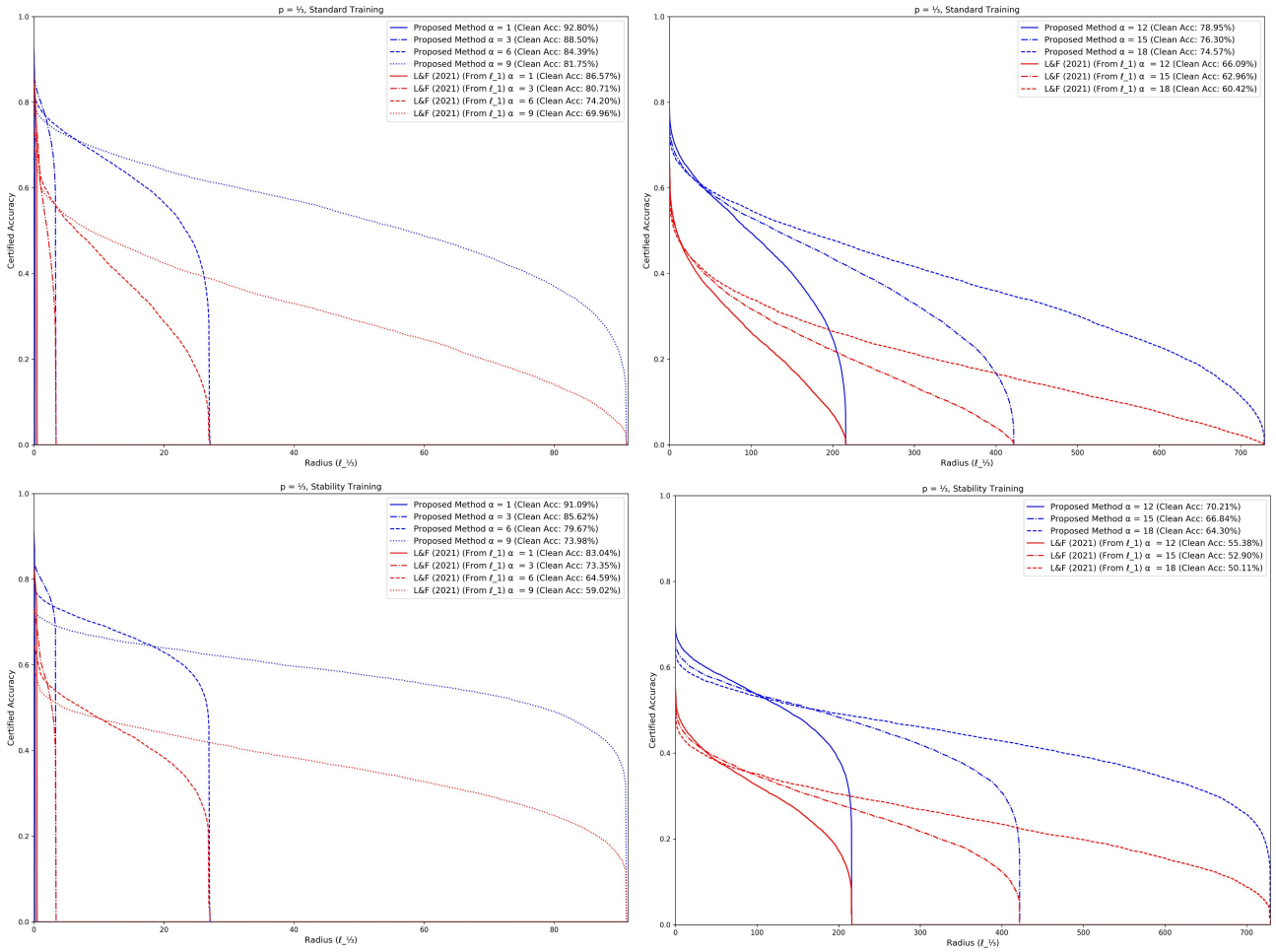

Figure 10: Full certification results for $p = 1/3$ on CIFAR-10. Left column shows $\alpha \in \{1, 3, 6, 9\}$, right column shows $\alpha \in \{12, 15, 18\}$, top row shows standard training, and bottom row shows stability training.

J CIFAR-10 $p = 1/2$ results with larger values of α

We repeated $p = 1/2$ CIFAR-10 experiments in Table 1 in the main text for the additional values of $\alpha \in \{21, 24, 27, 30\}$. Summary results are presented in Table 6. While this increases certified accuracy under large perturbations, it does so at the cost of decreased clean accuracy. The conclusion that our method significantly outperforms Levine and Feizi (2021) in the $p < 1$ case still holds. Full results for all classifiers are presented in Figure 11, and base classifier accuracies are in Table 7.

	$\ell_{1/2}$						
	30	60	90	120	150	180	210
L&F (2021) (From ℓ_1)	32.28% (53.35% @ α=30)	24.72% (53.35% @ α=30)	18.95% (53.35% @ α=30)	14.20% (53.35% @ α=30)	9.50% (53.35% @ α=30)	5.42% (53.35% @ α=30)	1.55% (53.35% @ α=30)
L&F (2021) (From ℓ_1) (Stab. Training)	32.39% (47.03% @ α=24)	26.41% (44.38% @ α=30)	22.34% (44.38% @ α=30)	18.68% (44.38% @ α=30)	15.07% (44.38% @ α=30)	11.21% (44.38% @ α=30)	6.21% (44.38% @ α=30)
Variable-Λ	**45.45%** (66.56% @ α=24)	37.45% (63.40% @ α=30)	30.71% (63.40% @ α=30)	24.90% (63.40% @ α=30)	19.40% (63.40% @ α=30)	13.11% (63.40% @ α=30)	5.67% (63.40% @ α=30)
Variable-Λ (Stab Training)	45.05% (60.44% @ α=18)	**38.16%** (56.23% @ α=24)	**33.74%** (52.58% @ α=30)	**29.79%** (52.58% @ α=30)	**25.83%** (52.58% @ α=30)	**20.84%** (52.58% @ α=30)	**14.19%** (52.58% @ α=30)

Table 6: Certified accuracy as a function of fractional ℓ_p distance ρ, for $p = 1/2$ on CIFAR-10 under large perturbations, with large values of α ($\alpha \in \{21, 24, 27, 30\}$) in addition to the α values used in the main text. As in Table 1, we report the highest certificate for each technique over all of the models.

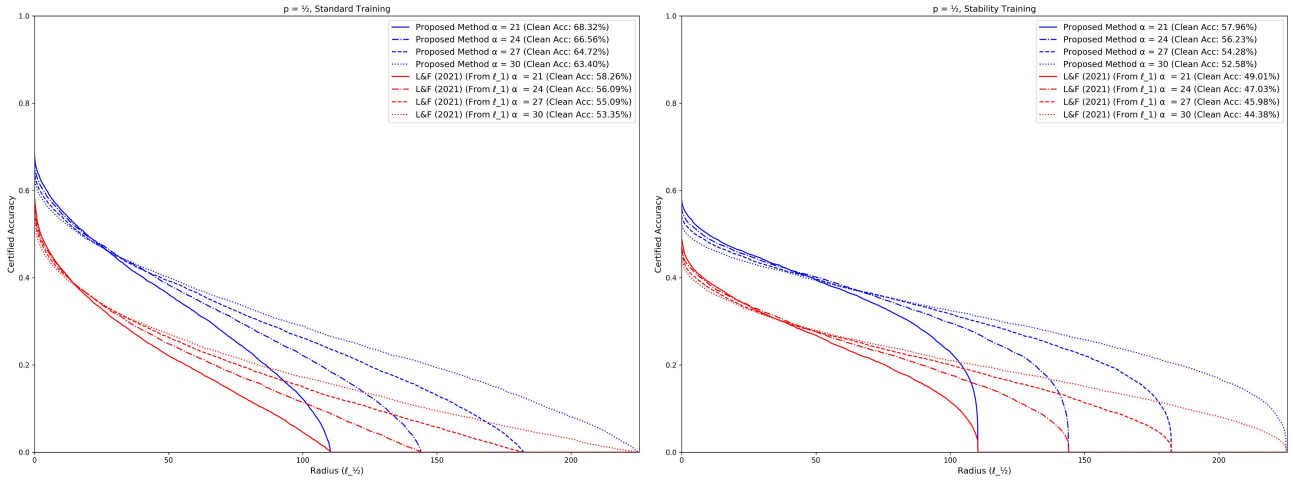

Figure 11: Full certification results for $p = 1/2$ on CIFAR-10, with $\alpha \in \{21, 24, 27, 30\}$. Left panel shows standard training, right panel shows stability training.

α	ℓ_1 (L&F 2021)	ℓ_1 (L&F 2021) (Stability)	$\ell_{1/2}$	$\ell_{1/2}$ (Stability)
21	47.04%	44.14%	58.17%	54.14%
24	45.13%	42.36%	55.91%	52.31%
27	43.49%	40.82%	53.97%	50.46%
30	41.99%	39.36%	52.34%	48.70%

Table 7: Base classifier accuracies for CIFAR-10, for large values of α.

K Base Classifier Accuracies for ImageNet

Base classifier accuracies for the ImageNet results in the main text are provided in Table 8.

α	ℓ_1 (L&F 2021)	$\ell_{1/2}$
6	52.50%	58.67%
12	45.49%	53.51%
18	40.39%	49.82%

Table 8: Base classifier accuracies on ImageNet. Note that for $p = 1/2$, the base classifier accuracy increases compared to $p = 1$ for each fixed value of α: this leads to larger certificates.