TOWARDS CERTIFYING ℓ_∞ ROBUSTNESS USING NEURAL NETWORKS WITH ℓ_∞-DIST NEURONS

Anonymous authors
Paper under double-blind review

ABSTRACT

It is well-known that standard neural networks, even with a high classification accuracy, are vulnerable to small ℓ_∞ perturbations. Many attempts have been tried to learn a network that can resist such adversarial attacks. However, most previous works either can only provide empirical verification of the defense to a particular attack method, or can only develop a theoretical guarantee of the model robustness in limited scenarios. In this paper, we develop a theoretically principled neural network that inherently resists ℓ_∞ perturbations. In particular, we design a novel neuron that uses ℓ_∞ distance as its basic operation, which we call ℓ_∞-dist neuron. We show that the ℓ_∞-dist neuron is naturally a 1-Lipschitz function with respect to the ℓ_∞ norm, and the neural networks constructed with ℓ_∞-dist neuron (ℓ_∞-dist Nets) enjoy the same property. This directly provides a theoretical guarantee of the certified robustness based on the margin of the prediction outputs. We further prove that the ℓ_∞-dist Nets have enough expressiveness power to approximate any 1-Lipschitz function, and can generalize well as the robust test error can be upper-bounded by the performance of a large margin classifier on the training data. Preliminary experiments show that even without the help of adversarial training, the learned networks with high classification accuracy are already provably robust.

1 INTRODUCTION

Modern neural networks are usually sensitive to small, adversarially chosen perturbations to the inputs (Szegedy et al., 2013; Biggio et al., 2013). Given an image $\boldsymbol{x}$ that is correctly classified by a neural network, a malicious attacker may find a small adversarial perturbation $\boldsymbol{\delta}$ such that the perturbed image $\boldsymbol{x}+\boldsymbol{\delta}$, though visually indistinguishable from the original image, is assigned to a wrong class with high confidence by the network. Such vulnerability creates security concerns in many real-world applications.

Developing a model that can resist small ℓ_∞ perturbations has been extensively studied in the literature. Adversarial training methods (Szegedy et al., 2013; Madry et al., 2017; Goodfellow et al., 2015; Huang et al., 2015; Athalye et al., 2018; Ding et al., 2020) first learn on-the-fly adversarial examples of the inputs, and then update model parameters using these perturbed samples together with the original labels. Such approaches are restricted to a particular (class of) attack method and cannot be formally guaranteed whether the resulting model is robust against other attacks. Another line of algorithms trains robust models by maximizing the certified radius provided by robust certification methods. Weng et al. (2018); Wong & Kolter (2018a); Zhang et al. (2018); Mirman et al. (2018); Wang et al. (2018); Gowal et al. (2018); Zhang et al. (2019b) develop their methods based on linear or convex relaxations of fully connected ReLU networks. However, the certification methods are usually computationally expensive and can only handle ReLU activations. Cohen et al. (2019); Salman et al. (2019); Zhai et al. (2020) show that a certified guarantee on small ℓ_2 perturbations can be easily computed for general Gaussian smoothed classifiers. But recent works suggest that such methods are hard to extend to the ℓ_∞-perturbation scenario.

In this work, we overcome the challenge mentioned above by introducing a new type of neural network that naturally resists local adversarial attacks and can be easily certified under the ℓ_∞ perturbation. In particular, we propose a novel neuron called ℓ_∞-dist neuron. Unlike the standard neuron design that uses a non-linear activation after a linear transformation, the ℓ_∞-dist neuron is purely based on computing the ℓ_∞ distance between the inputs and the parameters. It is straightforward to see that such a neuron is 1-Lipschitz with respect to the ℓ_∞ norm and the neural networks constructed with ℓ_∞-dist neuron (ℓ_∞-dist Nets) enjoy the same property. Based on such a property, we can obtain the certified robustness for any ℓ_∞-dist Nets using the margin of the prediction outputs.

Theoretically, we investigate the expressive power of ℓ_∞-dist Nets and its adversarially robust generalization ability. We first prove a Lipschitz-universal approximation theorem for ℓ_∞-dist Net using a structured approach, which shows that ℓ_∞-dist Nets can approximate any 1-Lipschitz function with respect to ℓ_∞ norm. We then give upper bounds of robust test error based on the Rademacher complexity, which shows that the robust test error would be small if the ℓ_∞-dist Net learns a large margin classifier on the training data. Both results demonstrate the excellent capability and generalization ability of the ℓ_∞-dist Net function class.

The ℓ_∞-dist Nets have nice theoretical guarantees, but empirically, training an ℓ_∞-dist Net is not easy. For example, the gradient of the parameters in the ℓ_∞ norm is sparse, which makes the optimization inefficient. We show how to initialize the model parameters, apply proper normalization, and overcome the sparse gradient problem via smoothed approximated gradients. Preliminary experiments on MNIST and Fashion-MNIST show that even without using adversarial training, the learned networks are already provably robust.

Our contributions are summarized as follows:

- We propose a novel neural network using ℓ_∞-dist neurons, called ℓ_∞-dist Nets. Theoretically,
  - In Section 3, we show that ℓ_∞-dist Nets are 1-Lipschitz with respect to the ℓ_∞ norm in nature, which directly guarantees the certified robustness of any ℓ_∞-dist Net (with respect to the ℓ_∞ norm).
  - In Section 4.1, we prove that ℓ_∞-dist Nets have good expressive power as it can approximate any 1-Lipschitz function with respect to the ℓ_∞ norm.
  - In Section 4.2, we prove that ℓ_∞-dist Nets have good generalization ability as the robust test error can be upper-bounded by the performance of a large margin classifier on the training data.
- We provide several implementation strategies which are shown to be practically helpful for model training (in Section 5).

2 Related Works

There are two major lines of works seeking to get robust neural networks:

Robust Training Approaches. Previous works showed that the conventional neural networks learned using standard training approaches (e.g., maximum likelihood method) are sensitive to small adversarial perturbations, and significant efforts have been put on developing training approaches for learning robust models. Adversarial training is the most successful method against adversarial attacks. By adding adversarial examples to the training set on the fly, adversarial training methods (Szegedy et al., 2013; Goodfellow et al., 2015; Huang et al., 2015; Zhang et al., 2019a; Wong et al., 2020) can significantly improve the robustness of the conventional neural networks. However, all the methods above are evaluated according to the empirical robust accuracy against pre-defined adversarial attack algorithms, such as projected gradient decent. It cannot be formally guaranteed whether the resulting model is also robust against other attacks.

Certified Robustness. Many recent works focus on certifying the robustness of learned neural networks under *any* attack. Approaches based on bounding the certified radius layer by layer using some convex relaxation methods have been proposed for certifying the robustness of neural networks (Wong & Kolter, 2018b; Gowal et al., 2018; Mirman et al., 2018; Dvijotham et al., 2018; Raghunathan et al., 2018; Zhang et al., 2020). However, such approaches are usually computationally expensive and have difficulties in scaling to deep models.

More recently, researchers found a new approach called randomized smoothing that can provide a certified robustness guarantee for general models. Lecuyer et al. (2018); Li et al. (2018); Cohen et al. (2019); Salman et al. (2019); Zhai et al. (2020) showed that if a Gaussian random noise is added to the input layer, a certified guarantee on small ℓ_2 perturbations can be computed for Gaussian smoothed classifiers. However, Yang et al. (2020); Blum et al. (2020); Kumar et al. (2020) showed that randomized smoothing cannot achieve nontrivial certified accuracy against more than $\Omega\left(\min\left(1, d^{1/p-1/2}\right)\right)$ radius for ℓ_p perturbations, where d is the input dimension, therefore it cannot provide meaningful results for ℓ_∞ perturbations due to the curse of dimensionality.

Another line of more conservative certification approaches sought to bound the global Lipschitz constant of the neural network (Gouk et al., 2018; Tsuzuku et al., 2018; Anil et al., 2019; Cisse et al., 2017). However, the bounds are not tight (Cohen et al., 2019), and the final robustness performances are not as good as other certification methods.

3 ℓ_∞-DIST NETWORK AND ITS ROBUSTNESS GUARANTEE

3.1 PRELIMINARIES

Throughout this paper, we will use bold letters to denote vectors and otherwise scalars. Consider a standard classification task with an underlying data distribution $\mathcal{D}$ over pairs of examples $\boldsymbol{x} \in \mathcal{X}$ and corresponding labels $y \in \mathcal{Y} = \{1, 2, \cdots, M\}$. Usually $\mathcal{D}$ is unknown and we can only access a training set $\mathcal{T} = \{(\boldsymbol{x}_1, y_1), \cdots, (\boldsymbol{x}_n, y_n)\}$ in which $(\boldsymbol{x}_i, y_i)$ is *i.i.d.* drawn from D, $i = 1, 2, \cdots, n$. Let $f \in \mathcal{F}$ be the classifier of interest that maps any $\boldsymbol{x} \in \mathcal{X}$ to $\mathcal{Y}$. We call $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\delta}$ an *adversarial example* of $\boldsymbol{x}$ to classifier f if f can correctly classify $\boldsymbol{x}$ but assigns a different label to $\boldsymbol{x}'$. In real practice, the most commonly used setting is to consider the attack under ϵ-bounded ℓ_∞ norm constraint, i.e., $\boldsymbol{\delta}$ satisfies $\|\boldsymbol{\delta}\|_\infty \le \epsilon$, which is also called ℓ_∞ perturbations.

Our goal is to learn a model from $\mathcal{T}$ that can resist attacks at $(\boldsymbol{x}, y)$ with high probability over $(\boldsymbol{x}, y) \sim \mathcal{D}$ for any small ℓ_∞ perturbations. It relates to compute the radius of the largest ℓ_∞ ball centered at $\boldsymbol{x}$ in which f does not change its prediction. This radius is called the *robust radius*, which is defined as (Zhai et al., 2020; Zhang et al., 2019a):

$$R(f; \boldsymbol{x}, y) = \begin{cases} \inf\limits_{f(\boldsymbol{x}') \neq f(\boldsymbol{x})} \|\boldsymbol{x}' - \boldsymbol{x}\|_\infty, & \text{when } f(\boldsymbol{x}) = y \\ 0 & \text{, when } f(\boldsymbol{x}) \neq y \end{cases} \tag{1}$$

Unfortunately, computing the robust radius (1) of a classifier induced by a standard deep neural network is very difficult. For example, Weng et al. (2018) showed that computing the ℓ_p robust radius of a deep neural network is NP-hard for some specific p. Researchers then seek to derive a tight lower bound of $R(f; \boldsymbol{x}, y)$ for general f. This lower bound is called *certified radius* and we denote it as $CR(f; \boldsymbol{x}, y)$. The certified radius satisfies $0 \le CR(f; \boldsymbol{x}, y) \le R(f; \boldsymbol{x}, y)$ for any $f, \boldsymbol{x}, y$.

3.2 NETWORKS WITH ℓ_∞-DIST NEURONS

In this subsection, we propose the ℓ_∞-dist Neuron that is inherently Lipschitz with respect to ℓ_∞ norm. Using these neurons as building blocks, we then show how to obtain an ℓ_∞-Lipschitz neural networks dubbed ℓ_∞-dist Net.

Denote $\boldsymbol{z}$ as the input vector to a neuron. A standard neuron processes the input by first projecting $\boldsymbol{z}$ to a scalar value using a linear transformation, and then applying an non-linear activation function σ on it, i.e., $\sigma(\boldsymbol{w}^\top \boldsymbol{z} + b)$. $\boldsymbol{w}$ and b are parameters, and function σ can be the sigmoid or ReLU activation. Unlike the previous design paradigm, we introduce a new type of neuron using ℓ_∞ distance as the basic operation, which we call ℓ_∞-dist neuron:

$$u(\boldsymbol{z}, \theta) = \|\boldsymbol{z} - \boldsymbol{w}\|_\infty + b, \tag{2}$$

where $\theta = \{\boldsymbol{w}, b\}$ is the parameter set. From the above equation, we can see that the ℓ_∞-dist neuron is non-linear as it calculates the ℓ_∞ distance between input $\boldsymbol{z}$ and parameter $\boldsymbol{w}$ with a bias term b. As a result, there is no need to further apply a non-linear activation function.

Without loss of generality, we study the properties of multi-layer perceptron (MLP) networks that are constructed using ℓ_∞-dist neurons. All theoretical results can be easily extended to other neural network architectures, such as convolutional neural nets. We use $\boldsymbol{x} \in \mathbb{R}^{d_{\text{input}}}$ to denote the input vector of an MLP network. A (MLP) network using ℓ_∞-dist neurons can be formally defined as follows.

Definition 3.1. *(ℓ_∞-dist Net) Denote $\boldsymbol{g}$ as an MLP network which takes $\boldsymbol{x}^{(1)} \triangleq \boldsymbol{x} \in \mathbb{R}^{d_{\text{input}}}$ as input. Assume there are L hidden layers and the l-th hidden layer contains d_l hidden units. We call $\boldsymbol{g}$ is an MLP network constructed by ℓ_∞-dist neurons, if the k-th unit in the l-th hidden layer $x_k^{(l+1)}$ is computed by*

$$x_k^{(l+1)} = u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = \|\boldsymbol{x}^{(l)} - \boldsymbol{w}^{(l,k)}\|_\infty + b^{(l,k)}, 0 < l \leq L, 0 < k \leq d_l,$$

where $\boldsymbol{x}^{(l)} = (x_1^{(l)}, x_2^{(l)}, \cdots, x_{d_{l-1}}^{(l)})$ is the input vector to the l-th hidden layer.

For simplicity, we will call an MLP network constructed by ℓ_∞-dist neurons as *ℓ_∞-dist Net*. For classification tasks, the dimensionality of the output of $\boldsymbol{g}$ matches the number of categories, i.e., M. We write $\boldsymbol{g}(\boldsymbol{x}) = (g_1(\boldsymbol{x}), g_2(\boldsymbol{x}), \cdots, g_M(\boldsymbol{x}))$ and define the predictor $f(\boldsymbol{x}) = \arg\max_{i\in[M]} g_i(\boldsymbol{x})$. Note that $\boldsymbol{g}(\boldsymbol{x})$ can be used as the output logits the same as in any other networks, so we can apply any standard loss function on the ℓ_∞-dist Net, such as the cross-entropy loss or hinge loss.

We will further show that the ℓ_∞-dist neuron and the neural networks constructed using it have nice theoretical properties to control the robustness of the model. For completeness, we first introduce the definition of Lipschitz functions as below.

Definition 3.2. *(Lipschitz Function) A function $\boldsymbol{g}(\boldsymbol{z}) : \mathbb{R}^m \to \mathbb{R}^n$ is called λ-Lipschitz with respect to ℓ_p norm $\|\cdot\|_p$, if for any $\boldsymbol{z}_1, \boldsymbol{z}_2$, the following holds:*

$$\|\boldsymbol{g}(\boldsymbol{z}_1) - \boldsymbol{g}(\boldsymbol{z}_2)\|_p \leq \lambda \|\boldsymbol{z}_1 - \boldsymbol{z}_2\|_p$$

It is straightforward to see that the ℓ_∞-dist neuron is 1-Lipschitz with respect to ℓ_∞ norm.

3.3 Lipschitz and Robustness Facts about ℓ_∞-dist Nets

In this subsection, we introduce two straightforward but important facts about ℓ_∞-dist Nets. We first show that ℓ_∞-dist Nets are still 1-Lipschitz with respect to ℓ_∞ norm, and then derive the certified robustness of the model based on this property.

Fact 3.1. *Any ℓ_∞-dist Net $\boldsymbol{g}(\cdot)$ is 1-Lipschitz with respect to ℓ_∞ norm, i.e., for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^{d_{\text{input}}}$, we have $\|\boldsymbol{g}(\boldsymbol{x}_1) - \boldsymbol{g}(\boldsymbol{x}_2)\|_\infty \leq \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_\infty$.*

Proof. It's easy to check that every operation $u(\boldsymbol{x}^{(l)}, \theta^{(l,k)})$ is 1-Lipschitz, and therefore the mapping from one layer to the next $\boldsymbol{x}^{(l)} \to \boldsymbol{x}^{(l+1)}$ is 1-Lipschitz. Then we have for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^{d_{input}}$, $\|\boldsymbol{g}(\boldsymbol{x}_1) - \boldsymbol{g}(\boldsymbol{x}_2)\|_\infty \leq \|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_\infty$. □

Since $\boldsymbol{g}$ is 1-Lipschitz with respect to ℓ_∞ norm, if the perturbation over $\boldsymbol{x}$ is rather small, the change of the output can be bounded and the label of the perturbed data $\boldsymbol{x}'$ will not change as long as $\arg\max_{i\in[M]} g_i(\boldsymbol{x}) = \arg\max_{i\in[M]} g_i(\boldsymbol{x}')$, which directly bounds the certified radius.

Fact 3.2. *Given model $f(\boldsymbol{x}) = \arg\max_{i\in[M]} g_i(\boldsymbol{x})$ defined as above, and $\boldsymbol{x}$ is correctly classified as y. We define $\text{margin}(\boldsymbol{x}; \boldsymbol{g})$ as the difference between the largest and second-largest elements of $\boldsymbol{g}(\boldsymbol{x})$, then for any $\boldsymbol{x}'$ satisfying $\|\boldsymbol{x} - \boldsymbol{x}'\|_\infty \leq \text{margin}(\boldsymbol{x}; \boldsymbol{g})/2$, we have that $f(\boldsymbol{x}) = f(\boldsymbol{x}')$ and:*

$$CR(f, \boldsymbol{x}, y) \geq \frac{\text{margin}(\boldsymbol{x}; \boldsymbol{g})}{2} \tag{3}$$

Proof. Since $\boldsymbol{g}(\boldsymbol{x})$ is 1-Lipschitz, each element of $\boldsymbol{g}(\boldsymbol{x})$ can move at most $\frac{\text{margin}(\boldsymbol{x};\boldsymbol{g})}{2}$ when x changes to x', therefore the largest element will remain the same. □

Using this bound, we can certify the robustness of an ℓ_∞-dist Net of *any size* under ℓ_∞ norm with little computational cost (only a forward pass). In contrast, existing certified methods suffer from either poor scalability (methods based on convex relaxation) or curse of dimensionality (randomized smoothing), and lack efficiency as well.

4 Expressive power and Robust Generalization of ℓ_∞-dist Net

The expressive power of a model family and its generalization are two central topics in machine learning. In the previous section, we show that an ℓ_∞-dist Net is 1-Lipschitz with respect to ℓ_∞ norm. Then it's natural to ask whether the designed neural network can approximate *any* 1-Lipschitz function (with respect to ℓ_∞ norm) and whether we can give theoretical guarantees on the robust test error of a learned model.

In this section, we give affirmative answers to both questions. We will first prove a Lipschitz-universal approximation theorem for ℓ_∞-dist Net using a structured approach, then give upper bounds of robust test error based on the Rademacher complexity of the model family. Without loss of any generality, we consider *binary classification* problems and assume the output dimension is 1. All the omitted proofs in this section can be found in the appendix.

4.1 Lipschitz-Universal Approximation of ℓ_∞-dist Net

In this subsection, we show that ℓ_∞-dist Nets with bounded width can approximate *any* 1-Lipschitz function (with respect to ℓ_∞ norm), formalized in the following theorem:

Theorem 1. *For any 1-Lipschitz function $\tilde{g}(\boldsymbol{x})$ (with respect to ℓ_∞ norm) on a bounded domain $\mathbb{K} \in R^{d_{\text{input}}}$ and any $\epsilon > 0$, there exists an ℓ_∞-dist Net $g(\boldsymbol{x})$ with width no more than $d_{input} + 2$, such that for all $\boldsymbol{x} \in \mathbb{K}$, we have*

$$\|g(\boldsymbol{x}) - \tilde{g}(\boldsymbol{x})\|_\infty \leq \epsilon. \tag{4}$$

To prove Theorem 1, we need the following key lemma:

Lemma 4.1. *For any 1-Lipschitz function $f(\boldsymbol{x})$ (with respect to ℓ_∞ norm) on a bounded domain $\mathbb{K} \in R^n$, and any $\epsilon > 0$, there exists a finite number of functions $f_i(\boldsymbol{x})$ such that for all $\boldsymbol{x} \in \mathbb{K}$*

$$\max_i f_i(\boldsymbol{x}) \leq f(\boldsymbol{x}) \leq \max_i f_i(\boldsymbol{x}) + \epsilon,$$

where each $f_i(\boldsymbol{x})$ has the following form

$$f_i(\boldsymbol{x}) = \min\{x_1 - w_1^{(i)}, w_1^{(i)} - x_1, x_2 - w_2^{(i)}, ..., w_n^{(i)} - x_n\} + b_i.$$

Lemma 4.1 "decomposes" any target 1-Lipschitz function into simple "base functions", which will serve as building blocks in proving the main theorem. Lu et al. (2017) first proved such universal approximation theorem for width-bounded ReLU networks by constructing a special network that approximates the target function by "sum of cubes". For ℓ_∞-dist Net, such an approach cannot be directly applied as the summation will break the Lipschitz property. We employ a novel "max of pyramids" construction to overcome the issue. The key idea is to approximate the target function from below using the maximum of many "pyramid-like" basic 1-Lipschitz functions.

Theorem 1 implies that an ℓ_∞-dist Net can approximate any 1-Lipschitz function with respect to ℓ_∞ norm on a compact set, using only finite width which is barely larger than the input dimension. Combining with the fact that an ℓ_∞-dist Net is 1-Lipschitz, we conclude that ℓ_∞-dist Nets is a good class of model to approximate 1-Lipschitz functions.

4.2 Bounding robust test error of ℓ_∞-dist Net

Good robust generalization ability, i.e. small empirical (margin) error implies small robust test error, is as important as strong expressive power. In this subsection, we give upper bound for the *robust test error* of ℓ_∞-dist Net.

Consider the following classification problem: let $(\boldsymbol{x}, y)$ be an instance-label couple where $\boldsymbol{x} \in \mathbb{K}$ and $y \in \{1, -1\}$ and denote $\mathcal{D}$ as the distribution of $(\boldsymbol{x}, y)$. For a function $g(\boldsymbol{x}) : \mathbb{R}^{d_{\text{input}}} \to \mathbb{R}$, we use $\text{sign}(g(\boldsymbol{x}))$ as the classifier. The r-robust test error γ_r of a classifier g is defined as

$$\gamma_r = \mathbb{E}_{\mathbb{D}} \left[\sup_{\|\boldsymbol{x}' - \boldsymbol{x}\|_\infty \leq r} \mathbb{I}_{y g(\boldsymbol{x}') \leq 0} \right]$$

We can upper bound γ_r by the margin error on training data and the size of network:

Theorem 2. *Let $\mathbb{F}$ denote the set of all g represented by an ℓ_∞-dist Net with width w and depth L, for every $t > 0$, with probability at least $1 - 2e^{-2t^2}$ over the random drawing of n samples, for all $r > 0$ and $g \in \mathbb{F}$ we have that*

$$\gamma_r \le \inf_{\delta\in(0,1]} \left[\underbrace{\frac{1}{n}\sum_{i=1}^{n} \mathbb{I}_{y_i g(\boldsymbol{x}_i)\le\delta+r}}_{\text{large training margin}} + \underbrace{\tilde{O}\left(\frac{Lw^2}{\delta\sqrt{n}}\right)}_{\text{network size}} + \left(\frac{\log\log_2(\frac{2}{\delta})}{n}\right)^{\frac{1}{2}} \right] + \frac{t}{\sqrt{n}}. \tag{5}$$

Theorem 2 provides a theoretical guarantee on the adversarial robust test error: when a large margin classifier is found on training data, and the size of the ℓ_∞-dist Net is not too large, then with high probability the model can generalize well in terms of adversarial robustness.

We prove the theorem in two steps. One step is to provide a margin bound to control the gap between standard training error and standard test error. In this step, we use Rademacher complexity $R(\mathbb{F})$ of a hypothesis set $\mathbb{F}$, where the test error β_r with r as margin is defined as: $\beta_r = \mathbb{E}_{\mathcal{D}}\left[\mathbb{I}_{yg(\boldsymbol{x})\le r}\right]$. The other step is to bound the gap between test error and robust test error. The two steps are provided in the following two lemmas.

Lemma 4.2. *Let $\mathbb{F}$ be a hypothesis class, then for any $t > 0$,*

$$\mathbb{P}\left(\exists g \in \mathbb{F} : \beta_r > \inf_{\delta\in(0,1]} \left[\frac{1}{n}\sum_{i=1}^{n} \mathbb{I}_{y_i g(\boldsymbol{x}_i)\le\delta+r} + \frac{48}{\delta}R_n(\mathbb{F}) + \left(\frac{\log\log_2(\frac{2}{\delta})}{n}\right)^{\frac{1}{2}} \right] + \frac{t}{\sqrt{n}}\right) \le 2e^{-2t^2}.$$

Lemma 4.3. *The r-robust test error γ_r is no larger than the test error β_r, i.e., $\gamma_r \le \beta_r$.*

5 Training ℓ_∞-dist Net

Motivated by the theoretical analysis in the previous sections, we would like to find a large margin solution on the training data. Therefore we use the standard hinge loss as the objective function:

$$L(g, \mathcal{T}) = \frac{1}{n}\sum_{i=1}^{n} \max\left\{0, t + \max_{y_i' \ne y_i} \boldsymbol{g}(\boldsymbol{x}_i)_{y_i'} - \boldsymbol{g}(\boldsymbol{x}_i)_{y_i}\right\} \tag{6}$$

where t controls the desired margin and should be slightly larger than twice of the desired robustness level according to Eqn. 3. It can be easily seen that $L(g, \mathcal{T})$ is a continuous function of the network parameters and is differentiable almost everywhere. Therefore any gradient based optimization method can be used to train an ℓ_∞-dist Net.

However, the optimization is not easy. We find that directly using SGD or Adam optimizer *fails* to train the network well. In fact, the gradient of the ℓ_∞ norm (i.e., $\|\boldsymbol{z} - \boldsymbol{w}\|_\infty$) with respect to $\boldsymbol{w}$ is very sparse, which makes the optimization inefficient. In this section, we list a few important optimization tricks which we find is effective in practice.

Normalization and parameter initialization Batch Normalization (Ioffe & Szegedy, 2015), which shifts and scales feature values in each layer, is shown to be one of the most important components in training deep neural networks. However, in ℓ_∞-dist Net, if we directly apply batch normalization, the Lipschitz constant of the network will change due to the scaling operation, and the robustness of the trained model cannot be guaranteed. To leverage the advantages of BatchNorm, we only apply the shift operation that shifts the output of each layer using the mean over the current batch during training. Similar to BatchNorm, we use the running mean during inference. Note that in inference, the running mean serves as an additional bias term in ℓ_∞-dist neuron, which does not affect the Lipschitz constant. Finally, we do not use affine transformation which is typically used in BatchNorm.

We apply the shift operation in all intermediate layers after calculating the ℓ_∞-dist norm, but not the final layer. As a result, we remove the bias term in the ℓ_∞-dist neuron in all intermediate layers. The bias in the last layer is initialized to zero. As for the weight initialization, weights and inputs should have the same scale due to the ℓ_∞ distance operation. Therefore we initialize all weights from standard Gaussian distribution since the inputs are normalized to have unit variance.

Smoothed approximated gradients. We find that training an ℓ_∞-dist from scratch is usually inefficient, and the model can easily be stuck in some bad local minima. To obtain a well-optimized model, we relax the ℓ_∞ distance by ℓ_p distance for each neuron, to get an approximate and non-sparse gradient of the parameters (Chen et al., 2020). During training, we set p to be a small value in the beginning, and linearly increase it in each iteration to reach a large value. When the training finishes, we freeze the model parameter and set p to ∞ as our final model.

6 Experiment

6.1 Experimental setting

Model details and training configurations. We train the ℓ_∞-dist Net on two benchmark datasets: MNIST and Fashion-MNIST. For both tasks, we use a 4-layer ℓ_∞-dist Net. Each hidden layer has 4096 neurons. Normalization is applied between each intermediate layer. The batch size is set to 512. Random-crop data augmentation and image standardization are used during training. We train the network using Adam optimizer with hyper-parameters $\beta_1 = 0.9$ and $\beta_2 = 0.99$. The training procedure is as follows: First, we relax the ℓ_∞-dist Net to ℓ_p-dist Net with $p = 6$ and train the model parameters for 30 epochs. Then we gradually increase p from 6 to 100 in the next 170 epochs. Finally, we fix $p = 100$ and train the last 50 epochs. The learning rate in the first and second parts is set to be 0.015 and is divided by 5.0 in the final 50 epochs. We use hinge loss with parameter $t = 0.9$. For the Fashion-MNIST dataset, we use exactly *the same model and the same hyper-parameters* as MNIST, except for the hinge loss parameter $t = 0.4$. Adjusting models and hyper-parameters may further improve the results.

Evaluation. Following Wong & Kolter (2018b); Gowal et al. (2018); Madry et al. (2017); Zhang et al. (2019b), we test the robustness of ℓ_∞-dist Net under ℓ_∞ radius 0.3 on MNIST and 0.1 on Fashion-MNIST. We use two evaluation metrics to measure the robustness of the model. We first evaluate the robust test accuracy under the Projected Gradient Descent (PGD) attack (Madry et al., 2017). Following standard practice, we set the number of steps of the PGD attack to be 20 and set the step size to be 0.01. We also calculate the certified radius for each sample using Eq. 3, and check the percentage of test samples that can be certified to be robust within the chosen radius.

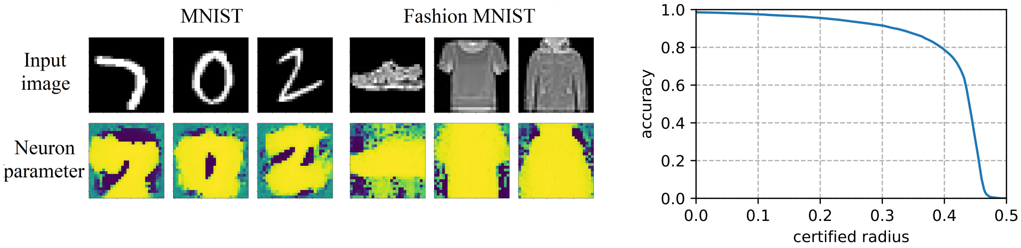
Table 1: Comparison of different methods on MNIST under ℓ_∞ radius 0.3.

Method	w/o Adv Train	Scalable	Standard Acc	Robust Acc	Certified Acc
Natural	✓	✓	99.3	≈ 0	≈ 0
IBP	✗	✗	98.34	93.88	91.95
CROWN-IBP	✗	✗	98.18	93.95	92.98
GroupSort	✓	✗	≈ 97.0	≈ 32.0	≈ 2.0
ℓ_∞-dist Net	✓	✓	98.61	93.78	91.59

Table 2: Comparison of different methods on Fashion-MNIST under ℓ_∞ radius 0.1.

Method	w/o Adv Train	Scalable	Standard Acc	Robust Acc	Certified Acc
Natural	✓	✓	94.0	≈ 0	≈ 0
CAP	✗	✗	78.27	68.37	65.47
IBP	✗	✗	-	-	76.51
ℓ_∞-dist Net	✓	✓	87.46	75.44	73.23

Baselines. In Table 1 and 2, we compare ℓ_∞-dist Net with several baselines, i.e., standard training on standard neural network (abbreviated as Natural in the table), CAP (Wong & Kolter, 2018b), IBP (Gowal et al., 2018), CROWN-IBP (Zhang et al., 2019b), GroupSort Network (Anil et al., 2019). Randomized Smoothing (Cohen et al., 2019; Salman et al., 2019; Zhai et al., 2020) is another strong baseline when considering the robustness with respect to ℓ_2 perturbation. However, as discussed in the related work section, it cannot achieve nontrivial certified accuracy against more than $\Omega\left(\min\left(1, d^{-1/2}\right)\right)$ radius for ℓ_∞ perturbation where d is the input dimension, which implies randomized smoothing cannot obtain good certification under ℓ_∞ norm. We report the performances



(a)

(b)

Figure 1: (a) We sample some neurons from the first layer and list the parameters w together with some input data on MNIST and Fashion-MNIST. It can be seen that on both tasks, the parameters (first row) look very similar to the data (second row) in some categories. This shows that the neuron captures the shape information, which is a meaningful feature and useful for classification. (b) The radius-(certified accuracy) curve of our trained model on MNIST dataset.

(picked from the original papers) together with other properties of these methods, e.g., whether these methods use adversarial training (abbreviated as Adv Train in the table); are these methods scalable to large neural networks.

6.2 Experimental results

We list the performances of ℓ_∞-dist Net in Table 1 and 2. In these tables, we use “Standard Acc”, “Robust Acc” and “Certified Acc” as abbreviations of standard (clean) test accuracy, robust test accuracy under PGD attack and certified robust test accuracy. All the numbers are reported in percentage.

From Table 1 and 2, we find that ℓ_∞-dist Net can achieve comparative or better robustness than other baselines while maintaining high standard accuracy. Specifically, on MNIST, among the methods that do not need adversarial training or can be scaled to large networks, ℓ_∞-dist Net has the best robustness and achieves more than 93% robust test accuracy and more than 91% certified test accuracy. These performances are also comparative with those methods that train the model using adversarial training and calculate the certified radius with huge computational cost (Wong & Kolter, 2018b; Gowal et al., 2018; Zhang et al., 2019b). We also plot the radius-(certified accuracy) curve on MNIST in Figure 1(b). Results on Fashion-MNIST are similar.

To understand how the ℓ_∞-dist Net extracts information from the data, we sample some neurons in the first layer and record its parameter w. We then visualize w together with some input data on both tasks. Interestingly, in Figure 1(a), we can see that the parameters w (first row) “look similar” to the input data (second row). This phenomenon is because we use ℓ_∞ distance in the network to obtain effective signals for classification. Then the neurons seek to find typical patterns that can differentiate one category to others in terms of ℓ_∞ distance, e.g., the shape of the objects, which is quite different from the feature extractor in standard neural networks.

7 Conclusion

In this paper, we design a novel neuron that uses ℓ_∞ distance as its basic operation. We show that the ℓ_∞-dist neuron is naturally a 1-Lipschitz function with respect to the ℓ_∞ norm and the neural networks constructed with ℓ_∞-dist neuron (ℓ_∞-dist Nets) enjoy the same property. This directly provides a theoretical guarantee of the certified robustness based on the margin of the prediction outputs. We further formally analyze the expressiveness power and the robust generalization ability of the network. Preliminary experiments show promising results on MNIST and Fashion-MNIST datasets. As future work, we will conduct experiments of the ℓ_∞-dist Net on more datasets (CIFAR10 and ImageNet) and extend it to modern networks, such as deep convolutional networks with residual connections.

REFERENCES

Cem Anil, James Lucas, and Roger Grosse. Sorting out lipschitz function approximation. In *International Conference on Machine Learning*, pp. 291–301, 2019.

Anish Athalye, Nicholas Carlini, and David A. Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *CoRR*, abs/1802.00420, 2018.

Peter L. Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63):1–17, 2019.

Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pp. 387–402. Springer, 2013.

Avrim Blum, Travis Dick, Naren Manoj, and Hongyang Zhang. Random smoothing might be unable to certify ℓ_∞ robustness for high-dimensional images. *arXiv preprint arXiv:2002.03517*, 2020.

Hanting Chen, Yunhe Wang, Chunjing Xu, Boxin Shi, Chao Xu, Qi Tian, and Chang Xu. Addernet: Do we really need multiplications in deep learning? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1468–1477, 2020.

Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. Parseval networks: Improving robustness to adversarial examples. In *International Conference on Machine Learning*, pp. 854–863, 2017.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 1310–1320, Long Beach, California, USA, 09–15 Jun 2019. PMLR.

Gavin Weiguang Ding, Yash Sharma, Kry Yik Chau Lui, and Ruitong Huang. Mma training: Direct input space margin maximization through adversarial training. In *International Conference on Learning Representations*, 2020.

Krishnamurthy Dvijotham, Sven Gowal, Robert Stanforth, Relja Arandjelovic, Brendan O'Donoghue, Jonathan Uesato, and Pushmeet Kohli. Training verified learners with learned verifiers. *arXiv preprint arXiv:1805.10265*, 2018.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael Cree. Regularisation of neural networks by enforcing lipschitz continuity. *arXiv preprint arXiv:1804.04368*, 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.

Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. Learning with a strong adversary. *arXiv preprint arXiv:1511.03034*, 2015.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pp. 448–456, 2015.

Vladimir Koltchinskii, Dmitry Panchenko, et al. Empirical margin distributions and bounding the generalization error of combined classifiers. *The Annals of Statistics*, 30(1):1–50, 2002.

Aounon Kumar, Alexander Levine, Tom Goldstein, and Soheil Feizi. Curse of dimensionality on randomized smoothing for certifiable robustness. *arXiv preprint arXiv:2002.03239*, 2020.

Mathias Lecuyer, Vaggelis Atlidakis, Roxana Geambasu, Daniel Hsu, and Suman Jana. Certified robustness to adversarial examples with differential privacy. *arXiv preprint arXiv:1802.03471*, 2018.

Bai Li, Changyou Chen, Wenlin Wang, and Lawrence Carin. Second-order adversarial attack and certifiable robustness. *CoRR*, abs/1809.03113, 2018.

Zhou Lu, Hongming Pu, Feicheng Wang, Zhiqiang Hu, and Liwei Wang. The expressive power of neural networks: A view from the width. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 6231–6239. Curran Associates, Inc., 2017.

Ulrike von Luxburg and Olivier Bousquet. Distance-based classification with lipschitz functions. *Journal of Machine Learning Research*, 5(Jun):669–695, 2004.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Matthew Mirman, Timon Gehr, and Martin Vechev. Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pp. 3578–3586, 2018.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. Certified defenses against adversarial examples. In *International Conference on Learning Representations*, 2018.

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*, pp. 11292–11303, 2019.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013.

Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In *Advances in neural information processing systems*, pp. 6541–6550, 2018.

Shiqi Wang, Yizheng Chen, Ahmed Abdou, and Suman Jana. Mixtrain: Scalable training of formally robust neural networks. *arXiv preprint arXiv:1811.02625*, 2018.

Lily Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. Towards fast computation of certified robustness for ReLU networks. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5276–5285, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 5286–5295, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018a. PMLR.

Eric Wong and Zico Kolter. Provable defenses against adversarial examples via the convex outer adversarial polytope. In *International Conference on Machine Learning*, pp. 5286–5295. PMLR, 2018b.

Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*, 2020.

Greg Yang, Tony Duan, Edward Hu, Hadi Salman, Ilya Razenshteyn, and Jerry Li. Randomized smoothing of all shapes and sizes. *arXiv preprint arXiv:2002.08118*, 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. *arXiv preprint arXiv:2001.02378*, 2020.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael Jordan. Theoretically principled trade-off between robustness and accuracy. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 7472–7482, Long Beach, California, USA, 09–15 Jun 2019a. PMLR.

Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *Advances in neural information processing systems*, pp. 4939–4948, 2018.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. *arXiv preprint arXiv:1906.06316*, 2019b.

Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. In *International Conference on Learning Representations*, 2020.

A Appendix

A.1 Proof of Lemma 4.1

Proof. Without loss of generality we may assume $\mathbb{K} \in [0,1]^n$. Consider the set $\mathbb{S}$ consisting of all points $\frac{\epsilon}{2}(N_1, ..., N_n)$ where N_j are integers, we can write $\mathbb{S} \cap \mathbb{K} = \{\boldsymbol{w}^{(1)}, ..., \boldsymbol{w}^{(m)}\}$ since it's a finite set. $\forall \boldsymbol{w}^{(i)} \in \mathbb{S} \cap \mathbb{K}$, we define the corresponding $f_i(\boldsymbol{x})$ as follows

$$f_i(\boldsymbol{x}) = \min\{x_1 - w_1^{(i)}, w_1^{(i)} - x_1, x_2 - w_2^{(i)}, ..., w_n^{(i)} - x_n\} + f(\boldsymbol{w}^{(i)}) \tag{7}$$

Obviously $\forall \boldsymbol{x} \in \mathbb{K}$ we have that $f(\boldsymbol{x}) \geq f_i(\boldsymbol{x})$, thus $\max_i f_i(\boldsymbol{x}) \leq f(\boldsymbol{x})$ holds as a direct consequence.

On the other hand, $\forall \boldsymbol{x} \in \mathbb{K}$ there exists its 'neighbour' $\boldsymbol{w}^{(j)} \in \mathbb{S} \cap \mathbb{K}$ such that $\|\boldsymbol{x} - \boldsymbol{w}^{(j)}\|_\infty \leq \frac{\epsilon}{2}$, therefore by using the Lipschitz properties of both $f(\boldsymbol{x})$ and $f_j(\boldsymbol{x})$, we have that

$$f(\boldsymbol{x}) \leq f(\boldsymbol{w}^{(j)}) + \frac{\epsilon}{2} = f_j(\boldsymbol{w}^{(j)}) + \frac{\epsilon}{2} \leq f_j(\boldsymbol{x}) + \epsilon \tag{8}$$

Since $\max_i f_i(\boldsymbol{x}) \geq f_j(\boldsymbol{x})$, we conclude our proof. □

A.2 Proof of Theorem 1

Proof. By lemma 4.1, there exists a finite number of functions $f_i(\boldsymbol{x})$ $(i = 1, ..., m)$ such that $\forall \boldsymbol{x} \in \mathbb{K}$

$$\max_i f_i(\boldsymbol{x}) \leq \tilde{g}(\boldsymbol{x}) \leq \max_i f_i(\boldsymbol{x}) + \epsilon \tag{9}$$

where each $f_i(\boldsymbol{x})$ has the form

$$f_i(\boldsymbol{x}) = \min\{x_1 - w_1^{(i)}, w_1^{(i)} - x_1, x_2 - w_2^{(i)}, ..., w_n^{(i)} - x_n\} + \tilde{g}(\boldsymbol{w}^{(i)}) \tag{10}$$

The high-level idea of the proof is very simple: among width $d_{\text{input}} + 2$, we allocate d_{input} neurons each layer to keep the information of $\boldsymbol{x}$, one to calculate each $f_i(\boldsymbol{x})$ one after another and the last neuron calculating the maximum of $f_i(\boldsymbol{x})$ accumulated.

To simplify the proof, we would first introduce three general basic maps which can be realized at a single unit, then illustrate how to represent $\max_i f_i(\boldsymbol{x})$ by combing these basic maps.

Let's assume for now that any input to any unit in the whole network has its ℓ_∞ norm upper bounded by a large constant C, we will come back later to determine this value and prove its validity.

Proposition A.1. *$\forall j, k, p$ and constant w, c, the following base functions are realizable at the kth unit in the lth hidden layer:*

1, the projection map:

$$u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = x_j^{(l)} + c \tag{11}$$

2, the negation map:

$$u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = -x_j^{(l)} + c \tag{12}$$

3, the maximum map:

$$u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = \max\{x_j^{(l)} + w, x_p^{(l)}\} + c, u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = \max\{-x_j^{(l)} + w, x_p^{(l)}\} + c \tag{13}$$

Proof. 1, the projection map: Setting $u(\boldsymbol{x}^{(l)}, \theta^{(l,k)})$ as follows

$$u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = \|(x_1^{(l)}, ..., x_j^{(l)} + 2C, ..., x_n^{(l)})\|_\infty - 2C + c \tag{14}$$

2, the negation map: Setting $u(\boldsymbol{x}^{(l)}, \theta^{(l,k)})$ as follows

$$u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = \|(x_1^{(l)}, ..., x_j^{(l)} - 2C, ..., x_n^{(l)})\|_\infty - 2C + c \tag{15}$$

3, the maximum map: Setting $u(\boldsymbol{x}^{(l)}, \theta^{(l,k)})$ as follows

$$u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = \|(x_1^{(l)}, ..., x_j^{(l)} + w + 2C, ..., x_p^{(l)} + 2C, ..., x_n^{(l)})\|_\infty - 2C + c \tag{16}$$

$$u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = \|(x_1^{(l)}, ..., -x_j^{(l)} + w + 2C, ..., x_p^{(l)} + 2C, ..., x_n^{(l)})\|_\infty - 2C + c \tag{17}$$

□

With three basic maps in hand, we are prepared to construct our network. Using proposition A.1, the first hidden layer realizes $u(\boldsymbol{x}, \theta^{(1,k)}) = x_k$ for $k = 1, ..., d_{\text{input}}$. The rest two units can be arbitrary, we set both to be x_1.

By proposition A.1, throughout the whole network, we can set $u(\boldsymbol{x}^{(l)}, \theta^{(l,k)}) = x_k$ for all l and $k = 1, ..., n$. Notice that $f_i(\boldsymbol{x})$ can be rewritten as

$$f_i(\boldsymbol{x}) = -\max\{x_1 - w_1^{(i)}, \max\{w_1^{(i)} - x_1, \max\{..., w_n^{(i)} - x_n\}...\}\} + \tilde{g}(\boldsymbol{w}^{(i)}) \tag{18}$$

Using the maximum map recurrently while keeping other units unchanged with the projection map, we can utilize the unit $u(\boldsymbol{x}^{(l)}, \theta^{(l,d_{\text{input}}+1)})$ to realize one $f_i(\boldsymbol{x})$ at a time. Again by the use of maximum map, the last unit $u(\boldsymbol{x}^{(l)}, \theta^{(l,d_{\text{input}}+2)})$ will recurrently calculate (initializing with $\max\{f_1(\boldsymbol{x})\} = f_1(\boldsymbol{x})$)

$$\max_i f_i(\boldsymbol{x}) = \max\{f_m(\boldsymbol{x}), \max\{..., \max\{f_1(\boldsymbol{x})\}...\}\} \tag{19}$$

using only finite depth, say L, then the network outputs $g(\boldsymbol{x}) = u(\boldsymbol{x}^{(L)}, \theta^{(L,1)}) = u(\boldsymbol{x}^{(L-1)}, \theta^{(L-1,d_{\text{input}}+2)}) = \max_i f_i(\boldsymbol{x})$ as desired. We are only left with deciding a valid value for C. Because $\mathbb{K}$ is bounded and $\tilde{g}(\boldsymbol{x})$ is continuous, there exists constants C_1, C_2 such that $\forall \boldsymbol{x} \in \mathbb{K}$, $\|\boldsymbol{x}\|_\infty \leq C_1$ and $|\tilde{g}(\boldsymbol{x})| \leq C_2$, it's easy to verify that $C = 2C_1 + C_2$ is valid.

□

A.3 PROOF OF LEMMA 4.2

First we give a quick revisit on Rademacher complexity and its properties.

Rademacher Complexity Given a sample $X_n = \{\boldsymbol{x}_1, ..., \boldsymbol{x}_n\} \in \mathbb{K}^n$, and a real-valued function class $\mathbb{F}$ on $\mathbb{K}$, the Rademacher complexity of $\mathbb{F}$ is defined as

$$R_n(\mathbb{F}) = \mathbb{E}_{X_n}\left(\frac{1}{n}\mathbb{E}_\sigma\left[\sup_{f\in\mathbb{F}}\sum_{i=1}^{n}\sigma_i f(\boldsymbol{x}_i)\right]\right)$$

where σ_i are drawn from the Rademacher distribution independently, i.e. $P(\sigma_i = 1) = P(\sigma_i = -1) = \frac{1}{2}$. It's worth noting that $\forall r, R_n(\mathbb{F}) = R_n(\mathbb{F} \oplus r)$ where $\mathbb{F} \oplus r = \{f + r | f \in \mathbb{F}\}$.

It's well known (using Massart's Lemma) that Rademacher complexity can be bounded by VC dimension:

$$R_n(\mathbb{F}) \leq \sqrt{\frac{2VCdim(\mathbb{F})\log\frac{en}{VCdim(\mathbb{F})}}{n}} \tag{20}$$

Proof. Denote the hypothesis set with $\mathbb{F}$ and its Rademacher complexity on training data with $R_n(\mathbb{F})$, we can upper bound the generalization error when a large margin solution is found on training data:

Lemma A.1. *(Theorem 11 in Koltchinskii et al. (2002)) For all $t > 0$,*

$$P\left(\exists g \in \mathbb{F} : \beta_0 > \inf_{\delta\in(0,1]}\left[\frac{1}{n}\sum_{i=1}^{n}\mathbb{I}_{y_i g(\boldsymbol{x}_i)\leq\delta} + \frac{48}{\delta}R_n(\mathbb{F}) + \left(\frac{\log\log_2(\frac{2}{\delta})}{n}\right)^{\frac{1}{2}}\right] + \frac{t}{\sqrt{n}}\right) \leq 2e^{-2t^2}$$

To further generalize lemma A.1 to β_r with $r > 0$, we use the fact that Rademacher complexity remain unchanged if the same constant r is added to all functions in $\mathbb{F}$. Lemma 4.2 is a direct consequence by replacing m_f by $m_f - r$ at the end of the proof of theorem 11 in Koltchinskii et al. (2002), where it plugs m_f into theorem 2 in Koltchinskii et al. (2002):

□

A.4 Proof of Lemma 4.3

Proof. $yg(\boldsymbol{x}) > r$ implies $\inf_{\|x'-x\|_\infty\leq r} yg(\boldsymbol{x}') > 0$ because $g(\boldsymbol{x})$ is 1-Lipschitz, therefore

$$\mathbb{E}_\mathcal{D}\left[\mathbb{I}_{yg(\boldsymbol{x})>r}\right] \leq \mathbb{E}_\mathcal{D}\left[\inf_{\|x'-x\|_\infty\leq r}\mathbb{I}_{yg(\boldsymbol{x}')>0}\right] \leq \mathbb{E}_\mathcal{D}\left[\sup_{\|x'-x\|_\infty\leq r}\mathbb{I}_{yg(\boldsymbol{x}')>0}\right]$$

and so $1 - \beta_r \leq 1 - \gamma_r$ which concludes the proof. □

A.5 Proof of Theorem 2

Proof. There are generalization bounds like (Luxburg & Bousquet, 2004), which involves only the Lipschitz property. Unfortunately, in those bounds, the dependence on sample size is of order $n^{\frac{-1}{d_{\text{input}}}}$, which suffers from the curse of dimensionality. To derive a more meaningful bound, we take the network size into consideration as well.

We can bound the VC dimension of ℓ_∞-dist Net by reducing a given ℓ_∞-dist Net network to a fully-connected ReLU network. We first introduce the VC bound for fully-connected neural networks with ReLU activation borrowed from Bartlett et al. (2019):

Lemma A.2. *(Theorem 6 in Bartlett et al. (2019)) Consider a fully-connected ReLU network architecture F with input dimension d, width $w \geq d$ and depth (number of hidden layers) L, then its VC dimension satisfies:*

$$VCdim(F) = \tilde{O}(L^2w^2) \tag{21}$$

The following lemma shows how to calculate ℓ_∞ distance using a fully-connected ReLU network architecture.

Lemma A.3. $\forall \boldsymbol{w} \in R^d$, *there exists a fully-connected ReLU network h with width $O(d)$ and depth $O(\log d)$ such that $h(\boldsymbol{x}) = \|\boldsymbol{x} - \boldsymbol{w}\|_\infty$*

Proof. The proof is by construction. Rewrite $\|\boldsymbol{x} - \boldsymbol{w}\|_\infty$ as

$$\max\{x_1 - w_1, w_1 - x_1, ..., x_d - w_d, w_d - x_d\}$$

which is a maximum of $2d$ items. Notice that $\max\{x, y\} = ReLU(x-y)+ReLU(y)-ReLU(-y)$, we can use $3d$ neurons in the first hidden layer so that the input to the second hidden layer are $\max\{x_i - w_i, w_i - x_i\}$, in all d items. Repeating this procedure which cuts the number of items within maximum by half, within $O(\log d)$ hidden layers this network finally outputs $\|\boldsymbol{x} - \boldsymbol{w}\|_\infty$ as desired. □

The VC bound of ℓ_∞-dist Net is formalized by the following lemma:

Lemma A.4. *Consider an ℓ_∞-dist Net architecture F with input dimension d, width $w \geq d$ and depth (number of hidden layers) L, then its VC dimension satisfies:*

$$VCdim(F) = \tilde{O}(L^2 w^4) \tag{22}$$

Proof. By lemma A.3, each neuron in the ℓ_∞-dist Net can be replaced by a fully-connected ReLU subnetwork with width $O(w)$ and depth $O(\log w)$. Therefore a fully-connected ReLU network architecture G with width $O(w^2)$ and depth $O(L \log w)$ can realize any function represented by the ℓ_∞-dist Net when parameters vary. Remember that VC dimension is monotone under the ordering of set inclusion, we conclude that such ℓ_∞-dist Net architecture F has VC dimension no more than that of G which equals $\tilde{O}(L^2 w^4)$ by lemma A.2. □

Finally, Theorem 2 is a direct consequence by combing Lemmas 4.2, 4.3, A.3, A.2 and A.4. □