Convergence of Adversarial Training in Overparametrized Networks

Ruiqi Gao*[1], Tianle Cai*[1], Haochuan Li[1], Liwei Wang[1], Cho-Jui Hsieh[2], and Jason D. Lee[3]

[1]Peking University
[2]University of California, Los Angeles
[3]University of Southern California

June 20, 2019

Neural networks are vulnerable to adversarial examples, i.e. inputs that are imperceptibly perturbed from natural data and yet incorrectly classified by the network. Adversarial training [34], a heuristic form of robust optimization that alternates between minimization and maximization steps, has proven to be among the most successful methods to train networks that are robust against a pre-defined family of perturbations. This paper provides a partial answer to the success of adversarial training. When the inner maximization problem can be solved to optimality, we prove that adversarial training finds a network of small robust train loss. When the maximization problem is solved by a heuristic algorithm, we prove that adversarial training finds a network of small robust surrogate train loss. The analysis technique leverages recent work on the analysis of neural networks via Neural Tangent Kernel (NTK), combined with online-learning when the maximization is solved by a heuristic, and the expressiveness of the NTK kernel in the ℓ_∞-norm.

1 Introduction

Recent studies have demonstrated that neural network models, despite achieving human-level performance on many important tasks, are not robust to adversarial examples—a

*Joint first author.

small and human imperceptible input perturbation can easily change the prediction label [45, 24]. This phenomenon brings out security concerns when deploying neural network models to real world systems [22]. In the past few years, many defense algorithms have been developed [25, 44, 33, 30, 40] to improve the network's robustness, but most of them are still vulnerable under stronger attacks, as reported in [3]. Among current defense methods, adversarial training [34] has become one of the most successful methods to train robust neural networks.

To obtain a robust network, we need to consider the "robust loss" instead of a regular loss. The robust loss is defined as the maximal loss within an ϵ-ball around each sample, and minimizing the robust loss under empirical distribution leads to a min-max optimization problem. Adversarial training [34] is a way to minimize the robust loss. At each iteration, it (approximately) solves the inner maximization problem by an attack algorithm $\mathcal{A}$ to get an adversarial sample, and then runs a (stochastic) gradient-descent update to minimize the loss on the adversarial sample. Although adversarial training has been widely used in practice and hugely improves the robustness of neural networks in many applications, its convergence properties are still unknown. It is unclear whether a network with small robust error exists and whether adversarial training is able to converge to a solution with minimal train adversarial loss.

In this paper, we study the convergence of adversarial training algorithms and try to answer the above questions on over-parameterized neural networks. We consider the setting where the neural network has H layers with width m, smooth activation, and n training samples. This assumption holds for many activation functions including the soft-plus and sigmoid. Our contributions are summarized below.

- For a general attack/perturbation algorithm $\mathcal{A}$, we show that gradient descent converges to a network where the robust surrogate loss with respect to the attack $\mathcal{A}$ is within ϵ of the optimal robust loss, when the width $m \geq \frac{2^{O(H)}n}{\epsilon^2}$ (Theorem 4.1).

- We then consider the expressivity of neural networks w.r.t. robust loss (or robust interpolation). We show when the width m is sufficiently large, the neural network can achieve optimal robust loss ϵ; see Theorems 5.1 and 5.2 for precise statement. By combining these results, we show that adversarial training finds networks of small robust training loss (Corollary 5.1 and Corollary 5.2).

- Conversely, the complexity of robust learning is higher. We show that the VC-Dimension of the model class which can *robustly* interpolate any n samples is lower bounded by $\Omega(nd)$ where d is the dimension. In contrast, there are neural net architectures that can interpolate n samples with only $O(n)$ parameters. For this class of architectures the VC-Dimension is upper bounded by $O(n \log n)$. Thus robust learning provably requires larger complexity and capacity.

2 Related Work

Attack and Defense Adversarial examples are inputs that are slightly perturbed from a natural sample and yet incorrectly classified by the model. An adversarial example can be generated by maximizing the loss function within an ϵ-ball around a natural sample. Thus, generating adversarial examples can be viewed as solving a constrained optimization problem and can be (approximately) solved by a projected gradient descent (PGD) method [34]. Some other techniques have also been proposed in the literature including l-BFGS [45], FGSM [24], iterative FGSM [28] and C&W attack [14], where they differ from each other by the distance measurements, loss function or optimization algorithms. There are also studies on adversarial attacks with limited information about the target model. For instance, [15, 26, 10, 32] considered the black-box setting where the model is hidden but the attacker can make queries and get the corresponding outputs of the model.

Improving the robustness of neural networks against adversarial attacks, also known as defense, has been recognized as an important and unsolved problem in machine learning. Various kinds of defense methods have been proposed [25, 44, 33, 30, 40], but many of them are based on obfuscated gradients which does not really improve robustness under stronger attacks [3]. As an exception, [3] reported that the adversarial training method developed in [34] is the only defense that works even under carefully designed attacks.

Adversarial Training Adversarial training is one of the first defense ideas proposed in earlier papers [24]. The main idea is to add adversarial examples into the training set to improve the robustness. However, earlier work usually only adds adversarial example once or only few times during the training phase. Recently, [34] showed that adversarial training can be viewed as solving a min-max optimization problem where the training algorithm aims to minimize the robust loss, defined as the maximal loss within a certain ϵ-ball around each training sample. Based on this formulation, a clean adversarial training procedure based on PGD-attack has been developed and achieved state-of-the-art results even under strong attacks. This also motivates some recent research on gaining theoretical understanding of robust error [11, 41]. Also, adversarial training suffers from slow training time since it runs several steps of attacks within one update, and several recent works are trying to resolve this issue [42, 53]. From the theoretical perspective, a recent work [46] considers to quantitatively evaluate the convergence quality of adversarial examples found in the inner maximization and therefore ensure robustness. [51] consider generalization upper and lower bounds for robust generalization. [31] improves the robust generalization by data augmentation with GAN. [23] considers to reduce the optimization of min-max problem to online learning setting and use their results to analyze the convergence of GAN. In this paper, our analysis for adversarial is quite general and is not restricted to any specific kind of attack algorithm.

Certified Defense and Robustness Verification For each sample, the robust loss is defined as the max loss within an ϵ-ball. Due to the non-convexity, attack algorithms usually fail to find the exact max, so robust error computed by an attack algorithm cannot give us a formal guarantee of robustness. As a consequence, networks trained by standard adversarial training algorithms [34], although being robust under strong attacks, do not have a certified guarantee of robustness.

Neural network verification methods, in contrast to attack, are trying to find upper bounds of robust error and provide certified robustness measurements. Several algorithms have been proposed recently. [48] proposed to solve the dual of a linear relaxation problem to obtain a certified bound. [47, 54] provides a similar algorithm based on primal relaxation. [43] proposed another approach based on abstract interpretation. More recently, [39] provided a unified view, showing that most of the existing verification methods are based on a convex relaxation of ReLU network.

Equipped with these verification methods for computing upper bounds of robust error, one can then apply adversarial training to get a network with certified robustness. This is first proposed in [48]. At each iteration, instead of finding a lower bound of robust error by attack, we can find an upper bound of robust error by verification and and train the model to minimize this upper bound. Several certified adversarial training algorithms along this line have been proposed recently [49, 21]. Our analysis in Section 4 can incorporate certified adversarial training.

Global convergence of Gradient Descent Recent work on the over-parametrization of neural networks prove that when the width greatly exceeds the sample size, gradient descent converges to a global minimizer from random initialization [29, 19, 20, 1, 55]. The key idea in the earlier literature is to show that the Jacobian w.r.t. parameters has minimum singular value lower bounded, and thus there is a global minimum near every random initialization, with high probability. However for the robust loss and robust surrogate loss, the maximization cannot be evaluated and the Jacobian is not necessarily full rank. Similarly with the robust surrogate loss, the heuristic attack algorithm may not even be continuous and so the same arguments cannot be utilized.

3 Preliminaries

3.1 Notation

Let $[n] = \{1, 2, \ldots, n\}$. We use $\mathcal{N}(\mathbf{0}, \mathbf{I})$ to denote the standard Gaussian distribution. For a vector $\mathbf{v}$, we use $\|\mathbf{v}\|_2$ to denote the Euclidean norm. For a matrix $\mathbf{A}$ we use $\|\mathbf{A}\|_F$ to denote the Frobenius norm and $\|\mathbf{A}\|_2$ to denote the operator norm. We use $\langle \cdot, \cdot \rangle$ to denote the standard Euclidean inner product between two vectors or matrices. We let $O(\cdot)$ and $\Omega(\cdot)$ denote standard Big-O and Big-Omega notations that suppress multiplicative constants.

3.2 Neural Network

In this paper we focus on the training of multilayer fully-connected neural networks. Formally, we consider a neural network of the following form.

Let $\mathbf{x} \in \mathbb{R}^d$ be the input, the fully-connected neural network is defined as follows: $\mathbf{W}^{(1)} \in \mathbb{R}^{m \times d}$ is the first weight matrix, $\mathbf{W}^{(h)} \in \mathbb{R}^{m \times m}$ is the weight at the h-th layer for $2 \leq h \leq H$, $\mathbf{a} \in \mathbb{R}^{m \times 1}$ is the output layer and $\sigma(\cdot)$ is the activation function.[1] The parameters are $\mathbf{W} = (\mathrm{vec}\{\mathbf{W}^{(1)}\}^\top, \cdots, \mathrm{vec}\{\mathbf{W}^{(H)}\}^\top, \mathbf{a}^\top)^\top$. We define the prediction function recursively (for simplicity we let $\mathbf{x}^{(0)} = \mathbf{x}$):

$$\begin{aligned} \mathbf{x}^{(h)} &= \sqrt{\frac{c_\sigma}{m}} \sigma\left(\mathbf{W}^{(h)} \mathbf{x}^{(h-1)}\right), 1 \leq h \leq H, \\ f(\mathbf{W}, \mathbf{x}) &= \mathbf{a}^\top \mathbf{x}^{(H)}, \end{aligned} \tag{1}$$

where $c_\sigma = \left(\mathbb{E}_{x \sim N(0,1)}\left[\sigma(x)^2\right]\right)^{-1}$ is a scaling factor to normalize the input at initialization.

We make a technical assumption on the activation function $\sigma(\cdot)$ which holds for many activation functions, although not the ReLU.

Assumption 3.1 (Smoothness of activation function)**.** *The activation function is Lipschitz and smooth, that is, we can assume there exists a constant $C > 0$ such that for any $z \in \mathbb{R}$*

$$|\sigma'(z)| \leq C \text{ and } \sigma'(z) \text{ is } C\text{-Lipschitz.}$$

Assumption 3.2 (Smoothness of loss)**.** *The loss $\ell(f(x), y)$ is Lipschitz, smooth, convex in $f(x)$ and satisfies $\ell(y, y) = 0$.*

[1] We assume intermediate layers are square matrices of size m for simplicity. It is not difficult to generalize our analysis to rectangular weight matrices.

We use the following initialization scheme: each entry in all $\mathbf{W}^{(h)}$ for $h \in [H]$ follows from the i.i.d. standard Gaussian distribution $\mathcal{N}(0,1)$, and $\mathbf{a}$ follows the i.i.d. uniform distribution on $\{-1,1\}$. Similar to [20], we consider the case when we only train on $\mathbf{W}^{(h)}$ for $h \in [H]$ and fix $\mathbf{a}$. For training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$, the (non-robust) training loss is $L(\mathbf{W}) = \frac{1}{n}\sum_{i=1}^n \ell(f(\mathbf{W}, \mathbf{x}_i), y_i)$.

The key architectural parameter is the width m. As we shall see, the robust train loss we obtain scales inversely with the width m, and so for overparametrized networks we are able to minimize the robust train loss.

3.3 Perturbation and the Surrogate Loss Function

The goal of adversarial training is to make the model robust in a neighbor of each datum. We first introduce the definition of the perturbation set function to determine the perturbation set at each points.

Definition 3.1 (Perturbation Set)**.** *The perturbation set function is $\mathcal{B} : \mathbb{R}^d \to \mathcal{P}(\mathbb{R}^d)$, where, we use $\mathcal{P}(\mathbb{R}^d)$ to stand for the power set of $\mathbb{R}^d$. At each data point $\mathbf{x}$, $\mathcal{B}(\mathbf{x})$ gives the perturbation set that we would like to guarantee the robustness on. For example, commonly used perturbation sets are $\mathcal{B}_\infty(\mathbf{x}) = \{\mathbf{x}' : \|\mathbf{x}' - \mathbf{x}\|_\infty < \epsilon_0\}$ and $\mathcal{B}_2(\mathbf{x}) = \{\mathbf{x}' : \|\mathbf{x}' - \mathbf{x}\|_2 < \epsilon_0\}$. Given a dataset $\{\mathbf{x}_i, y_i\}_{i=1}^n$, we say that the perturbation set is compatible with the dataset if $\overline{\mathcal{B}(\mathbf{x}_i)} \cap \overline{\mathcal{B}(\mathbf{x}_j)} \neq \phi$ implies $y_i = y_j$. In the rest of the paper, we will always assume that $\mathcal{B}$ is compatible to the given data. Our framework allows for arbitrary perturbation sets compatible with the empirical dataset.*

Given a perturbation set, we are now ready to define the perturbation function that map a data point to another point inside its perturbation set. We note that the perturbation function can be quite general including the identity function, the adversarial attack mapping and some random sample mapping. Formally, we give the following definition.

Definition 3.2 (Perturbation Function)**.** *A perturbation function is defined as a function $\mathcal{A} : \mathcal{W} \times \mathbb{R}^d \to \mathbb{R}^d$, where $\mathcal{W}$ is the parameter space. Given the parameter $\mathbf{W}$ of the neural network* (1)*, $\mathcal{A}(\mathbf{W}, \mathbf{x})$ maps $\mathbf{x} \in \mathbb{R}^d$ to $\mathbf{x}' \in \mathcal{B}(\mathbf{x})$ where $\mathcal{B}(\mathbf{x})$ refers to the perturbation set defined in Definition 3.1.*

With the definition of perturbation function, we can now define a large family of loss functions on the training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$. We will show this definition covers the standard loss used in empirical risk minimization and the robust loss used in adversarial training.

Definition 3.3 (Surrogate Loss Function)**.** *Given a perturbation function $\mathcal{A}$ defined in Definition 3.2, the current parameter $\mathbf{W}$ of the neural network f defined in* (1)*, a training* *set $\{\mathbf{x}_i, y_i\}_{i=1}^n$, we define the surrogate loss function $L_{\mathcal{A}}(\mathbf{W})$ on the training set as*

$$L_{\mathcal{A}}(\mathbf{W}) = \frac{1}{n}\sum_{i=1}^{n} \ell(f(\mathbf{W}, \mathcal{A}(\mathbf{W}, \mathbf{x}_i)), y_i).$$

It can be easily observed that the standard training loss $L(\mathbf{W})$ is a special case of surrogate loss function with $\mathcal{A}$ as the identity. The goal of adversarial training is to minimize the robust loss, i.e. $L_{\mathcal{A}}(\mathbf{W})$ with $\mathcal{A}(\mathbf{W}, \mathbf{x}) = \mathrm{argmax}_{\mathbf{x}\in\mathcal{B}(\mathbf{x})}\, \ell(f(\mathbf{W}, \mathcal{A}(\mathbf{W}, \mathbf{x})), y)$. We denote the robust loss as $L_*(\mathbf{W})$.

4 Convergence Results of Adversarial Training

We consider optimizing the surrogate loss $L_{\mathcal{A}}$ with the perturbation function $\mathcal{A}(\mathbf{W}, \mathbf{x})$ defined in Definition 3.2. In this section, we will prove that after certain steps of projected gradient descent with a convex set $\mathcal{R}(\mathbf{W}_0, B)$, the loss $L_{\mathcal{A}}$ is provably upper-bounded with the best minimax loss in this set.

$$\min_{\mathbf{W}\in\mathcal{R}(\mathbf{W}_0,B)} L_*(\mathbf{W}),$$

where

$$\mathcal{R}(\mathbf{W}_0, B) = \left\{\mathbf{W} : \left\|w_r^{(h)} - w_r^{(h)}(0)\right\|_2 \le \frac{B}{\sqrt{m}}, h \in [H], r \in [m]\right\}, \tag{2}$$

where $w_r^{(h)}$ is the r-th row of $\mathbf{W}^{(h)}$ and $w_r^{(h)}(0)$ is the r-th row of $\mathbf{W}_0^{(h)}$, and B depends polynomially on the smoothness parameters of Assumptions 3.1 and 3.2.

Denote the parameter $\mathbf{W}$ at the k-th iteration as $\mathbf{W}^k$, and similarly $\mathbf{W}^{k,(h)}$ and $w_r^{k,(h)}$. For each step in adversarial training, projected gradient descent takes an update

$$\begin{aligned}\mathbf{V}^{k+1} &= \mathbf{W}^k - \alpha\nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k),\\ \mathbf{W}^{k+1} &= \mathcal{P}_{\mathcal{R}}(\mathbf{V}^{k+1}),\end{aligned}$$

where

$$\nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k) = \frac{1}{n}\sum_{i=1}^{n} l'\left(f(\mathbf{W}, \mathcal{A}(\mathbf{W}, \mathbf{x}_i)), y_i\right) \nabla_{\mathbf{W}} f(\mathbf{W}, \mathcal{A}(\mathbf{W}, \mathbf{x}_i)),$$

the gradient $\nabla_{\mathbf{W}} f$ is with respect to the first argument $\mathbf{W}$, and $\mathcal{P}_{\mathcal{R}}$ is the Euclidean projection to a convex set $\mathcal{R}$. We will take $\mathcal{R}$ as the convex set $\mathcal{R}(\mathbf{W}_0, B)$ defined in Equation (2).

We show that for sufficiently wide neural networks, within the set $\mathcal{R}(\mathbf{W}_0, B)$ in the parameter space, gradient descent can find a point with surrogate loss no more than the minimum robust loss in $\mathcal{R}(\mathbf{W}_0, B)$. In Section 5, we show that the set $\mathcal{R}(\mathbf{W}_0, B)$ is sufficiently large to find a classifier of low robust loss. We assume the perturbation set of the input $\mathcal{B}(\mathbf{x})$ is in a Euclidean ball with radius e and $e \leq 1$. Specifically, we have the following theorem.

Theorem 4.1 (Convergence of Projected Gradient Descent for Optimizing Surrogate Loss)**.** *Suppose the input $\mathbf{x}$ is bounded, the activation function $\sigma(\cdot)$ satisfies Assumption 3.1, and the loss function satisfies Assumption 3.2. If we run projected gradient descent based on the convex constraint set $\mathcal{R}(\mathbf{W}_0, B)$ with stepsize $\alpha = 2^{-\Omega(H)}$, then with probability 0.99, for any $\epsilon > 0$, if $m = \Omega\left(\frac{B^4 n 2^{O(H)}}{\epsilon^2}\right)$, we have*

$$\min_{k=1,\cdots,T} L_{\mathcal{A}}(\mathbf{W}^k) - L_*(\mathbf{W}^*) \leq \epsilon, \tag{3}$$

where $\mathbf{W}^* = \min_{\mathbf{W} \in \mathcal{R}(\mathbf{W}_0, B)} L_*(\mathbf{W})$ and $T = \Omega(\frac{B^2}{\epsilon\alpha})$.

Remark. *Recall that the surrogate loss $L_{\mathcal{A}}(\mathbf{W})$ is the loss suffered when with respect to the perturbation function $\mathcal{A}$. For example if the adversary uses the projected gradient ascent algorithm, then the theorem guarantees that projected gradient ascent cannot successfully attack the learned network.*

Remark. *For two-layer networks $H = 1$, the update on $\mathbf{W}$ does not require the projection step as it is implicitly enforced by gradient descent.*

4.1 Proof Sketch

Our proof idea utilizes the same high-level intuition as [29, 19, 55, 12, 13] that near the initialization the network is linear. However, unlike these earlier works, the surrogate loss neither smooth, nor semi-smooth so there is no Polyak gradient domination phenomenon to allow for the global geometric contraction of gradient descent. In fact due to the the generality of perturbation function $\mathcal{A}$ allowed, the robust surrogate loss is not differentiable nor even continuous in $\mathbf{W}$, and so the standard analysis cannot be applied. Our analysis utilizes two key observations. First the network $f(\mathbf{W}, \mathcal{A}(\mathbf{W}, \mathbf{x}))$ is still smooth w.r.t. the first argument[2], and is close to linear in the first argument near initialization, which is shown by directly bounding the Hessian w.r.t. $\mathbf{W}$. Second, the perturbation function $\mathcal{A}$ can be treated as an adversary providing a worst-case loss function $\ell_{\mathcal{A}}(f, y)$ as done in online

[2]It is not jointly smooth in W, which is part of the subletly of the analysis.

learning. However, online learning typically assumes the sequence of losses is convex, which is not the case here. We make a careful decoupling of the contribution to non-convexity from the first argument and the worst-case contribution from the perturbation function, we can prove gradient descent succeeds in minimizing the surrogate loss.

5 Adversarial Training Finds Robust Classifier

Motivated by the optimization result in Theorem 4.1, we hope to show that there is indeed a robust classifier in $\mathcal{R}(\mathbf{W}_0, B)$. To show this, we utilize the connection between neural networks and their induced Reproducing Kernel Hilbert Space (RKHS) via viewing neural networks trained near initialization as a random feature scheme [16, 17, 27, 2]. Since we only need to show the existence of a network architecture that robustly fits the training data in $\mathcal{R}(\mathbf{W}_0, B)$ and neural networks are at least as expressive as their induced kernels, we may prove this via the RKHS connection. The strategy is to first show the existence of a robust classifier in the RKHS, and then show that a sufficiently wide network can approximate the kernel via random feature analysis. The results of this section will have, in general, exponential in dimension dependence due to the known issue of d-dimensional functions having exponentially large RKHS norm [4], so only offer *qualitative guidance on existence of robust classifiers*.

Since deep networks contain two-layer networks as a sub-network, and this section is only concerned with expressivity, we focus on the local expressivity of two-layer networks. We write the standard two-layer network in the suggestive way

$$f(\mathbf{W}, \mathbf{x}) = \frac{1}{\sqrt{2m}} \left(\sum_{r=1}^{m} a_r \sigma(\mathbf{w}_r^\top \mathbf{x}) + \sum_{r=1}^{m} a_r' \sigma(\bar{\mathbf{w}}_r^\top \mathbf{x}) \right),$$

and initialize as $\mathbf{w}_r \sim \mathcal{N}(0, \frac{1}{d})$ and $\bar{\mathbf{w}}_r$ is set to be equal to $\mathbf{w}_r$, a_r is randomly drawn from $\{1, -1\}$ and $a_r' = -a_r$. We denote $\mathbf{w}_{r0}, a_{r0}, \bar{\mathbf{w}}_{r0}, a_{r0}'$ the initialization parameters respectively. In this section, we will consider the data and perturbation set defined on the surface $\mathcal{S}$ of the unit ball, i.e. we assume $\mathbf{x} \in \mathcal{S}$ and $\mathcal{B}(\mathbf{x}) \subset \mathcal{S}$.

For convenience, we firstly introduce the Neural Tangent Kernel (NTK) [27] w.r.t. our neural network formulation in Equation (1).

Definition 5.1 (NTK [27])**.** *The NTK with activation function $\sigma(\cdot)$ and initialization distribution $\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I}_d)$ is defined as $K_\sigma(\mathbf{x}, \mathbf{y}) = \mathbb{E}_{\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I}_d)} \langle \mathbf{x}\sigma'(\mathbf{w}^\top \mathbf{x}), \mathbf{y}\sigma'(\mathbf{w}^\top \mathbf{y}) \rangle$.*

For a given kernel K, there is a reproducing kernel Hilbert space (RKHS) introduced by K. We denote it as $\mathcal{H}(K)$. We refer the readers to [37] for an introduction of the theory of RKHS.

In Section 5.1, we will first give a general existence result of classifier with robust loss no more than ϵ for two-layer networks with activation functions that induce universal kernels. Secondly, specifically for a two-layer quadratic-ReLU activation neural network, we show that adversarial training can find a robust classifier, and provide the explicit dependence of the width m w.r.t. ϵ.

5.1 Existence of Robust Classifier near Initialization

We formally make the following assumption, which is later verified when the activation induces an universal kernel.

Assumption 5.1 (Existence of Robust Classifier in NTK)**.** *For any $\epsilon > 0$, there exists $f \in \mathcal{H}(K_\sigma)$, such that $|f(\mathbf{x}) - y_i| \leq \epsilon$, for every $\mathbf{x} \in \mathcal{B}(\mathbf{x}_i)$, where $\mathcal{B}(\mathbf{x}_i) \subset \mathcal{S}$ is the perturbation set defined in Definition 3.1.*

Assumption 5.1 can be verified for a large class of activation functions by showing their induced kernel is universal as done in [35]. In addition, we will show that this assumption is mild in our example of quadratic-ReLU network.

Under this assumption, by applying the strategy of approximating the infinite situation by finite sum of random features, we can get the following theorem:

Theorem 5.1 (Robust Classifier near Initialization)**.** *Given dataset $\mathcal{D}$ equipped with a compatible perturbation set function $\mathcal{B}$ (See Definition 3.1). Under Assumption 5.1, given $\epsilon > 0$, there exists $B_{\mathcal{D},\mathcal{B},\epsilon}$ such that when the width m satisfies $m > \frac{B^2_{\mathcal{D},\mathcal{B},\epsilon}}{\epsilon}$, with probability at least 0.99 there exists $\mathbf{W}$ such that*

$$L_*(\mathbf{W}) \leq \epsilon \textit{ and } \mathbf{W} \in \mathcal{R}(\mathbf{W}_0, B_{\mathcal{D},\mathcal{B},\epsilon}).$$

This theorem shows that we can indeed find a classifier of low robust loss within a neighborhood of the initialization. Combining Theorem 4.1 and 5.1 we know that

Corollary 5.1 (Adversarial Training Finds a Network of Small Robust Train Loss)**.** *Given data set on the unit sphere equipped with a compatible perturbation set function and an associated perturbation function $\mathcal{A}$, which also takes value on the unit sphere. Suppose Assumption 3.1, 3.2, 5.1 are satisfied. Then there exists a $B_{\mathcal{D},\mathcal{B},\epsilon}$ which only depends on dataset $\mathcal{D}$, perturbation $\mathcal{B}$ and ϵ, corresponding to the RKHS radius, such that for any 2-layer fully connected network with width $m = \Omega(\frac{B^4_{\mathcal{D},\mathcal{B},\epsilon}}{\epsilon^2})$, if we run projected gradient descent with stepsize α on $\mathcal{R}(\mathbf{W}_0, B_{\mathcal{D},\mathcal{B},\epsilon})$ for $T = \Omega(\frac{B^2_{\mathcal{D},\mathcal{B},\epsilon}}{\epsilon\alpha})$ steps, then with probability 0.99,*

$$\min_{k=1,\cdots,T} L_{\mathcal{A}}(\mathbf{W}^k) \leq \epsilon. \tag{4}$$

Therefore, adversarial training is guaranteed to find a robust classifier under a given attack algorithm when the network width is sufficiently large.

5.2 Example: Two-layer Quadratic-ReLU Network

We consider the arc-cosine neural tangent kernel (NTK) introduced by two-layer network with *quadratic ReLU* activation function as a guide example. In this section, we quantitatively derive the dependency of ϵ for $B_{\mathcal{D},\mathcal{B},\epsilon}$ and m in Theorem 5.1 for this two-layer network and verify that the induced kernel is universal. The network has the expression

$$f(\mathbf{W}, \mathbf{x}) = \frac{1}{\sqrt{M}} \sum_{r=1}^{M} a_r \sigma(\mathbf{w}_r^\top \mathbf{x}) \tag{5}$$

where the activation $\sigma(x) = \mathrm{ReLU}\,(x)^2$ ($\mathrm{ReLU}\,(x) = \max\{0, x\}$), and a is initialized uniformly from $\{\pm 1\}$, and $\mathbf{W}$ is initialized i.i.d. from $\mathcal{N}(0,1)$ and only $\mathbf{W}$ is trained. The NTK has the following explicit expression:

$$K_\sigma(\mathbf{x}, \mathbf{y}) = \mathbb{E}_{\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I}_d)} \langle \mathbf{x}\mathrm{ReLU}\left(\mathbf{w}^\top \mathbf{x}\right), \mathbf{y}\mathrm{ReLU}\left(\mathbf{w}^\top \mathbf{y}\right) \rangle, \tag{6}$$

We denote $\|\cdot\|_{\mathcal{H}}$ the RKHS norm of $\mathcal{H}(K_\sigma)$. The following lemma gives a sufficient condition for the function to be in $\mathcal{H}(K_\sigma)$.

Lemma 5.1 (RKHS contains smooth functions, Proposition 2 in [4], Corollary 6 in [8])**.** *Let $f : \mathcal{S} \to \mathbb{R}$ be an even function such that all i-th order derivatives exist and are bounded by η for $0 \le i \le s$, with $s \ge (d+3)/2$. Then $f \in \mathcal{H}(K_\sigma)$ with $\|f\|_{\mathcal{H}} \le C_d \eta$ where C_d is a constant that only depend on the dimension d.*

We then make a mild assumption of the dataset[3]

Assumption 5.2 (Non-overlapping)**.** *The dataset $\{\mathbf{x}_i, y_i\}_{i=1}^n \subset \mathcal{S}$ and the perturbation set function $\mathcal{B}$ satisfies:*

- *$\mathcal{B}(\mathbf{x}_i)$ is compact set on $\mathcal{S}$ for all i,*
- *There does not exist $\mathbf{x}, \bar{\mathbf{x}}$ and i, j such that $\mathbf{x} \in \mathcal{B}(\mathbf{x}_i) \cup (-\mathcal{B}(\mathbf{x}_i)), \bar{\mathbf{x}} \in \mathcal{B}(\mathbf{x}_j) \cup (-\mathcal{B}(\mathbf{x}_j))$ but $y_i \neq y_j$.*

[3] Our assumption on the dataset essentially requires $\mathbf{x}_i \neq \pm \mathbf{x}_j$ since the ReLU NTK kernel only contains even functions. However, this can be enforced via a lifting trick: let $\tilde{\mathbf{x}} = [\mathbf{x}, 1] \in \mathbb{R}^{d+1}$, then the data $\tilde{\mathbf{x}}$ lie on the positive hemisphere. On the lifted space, even functions can separate any datapoints.

Under this assumption, one can easily construct a smooth classifier g on $\mathcal{S}$ such that $g(\mathbf{x}) = y_i$ for all $\mathbf{x} \in \mathcal{B}(\mathbf{x}_i)$. By Lemma 5.1, we have $g \in \mathcal{H}(K_\sigma)$ with RKHS norm $\|g\|_{\mathcal{H}} \leq C_{\mathcal{D}}$ where $C_{\mathcal{D}}$ is a constant only depends on dataset $\mathcal{D}$ and perturbation function. We then approximate g using random feature techniques. The following theorem provides the desired result:

Theorem 5.2 (Approximation by finite sum)**.** *For a given Lipschitz function $h \in \mathcal{H}(K_\sigma)$. For $\epsilon > 0, \delta \in (0,1)$, let $\mathbf{w}_1, \cdots, \mathbf{w}_M$ be sampled i.i.d. from $\mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I}_d)$ where*

$$M = \Omega\left(C_{\mathcal{D},\mathcal{B}} \frac{1}{\epsilon^{d+1}} \log \frac{1}{\epsilon^{d+1}\delta}\right). \tag{7}$$

and $C_{\mathcal{D},\mathcal{B}}$ is a constant that only depends on the dataset $\mathcal{D}$ and the compatible perturbation $\mathcal{B}$. Then with probability at least $1-\delta$, there exists $c_1, \cdots, c_M$ where $c_i \in \mathbb{R}^d$ such that $\hat{h} = \sum_{r=1}^{M} c_r^\top \mathbf{x} \mathrm{ReLU}\left(\mathbf{w}_r^\top \mathbf{x}\right)$ satisfies

$$\sum_{r=1}^{M} \|c_r\|_2^2 = O\left(\frac{1}{M}\right), \tag{8}$$

$$\left\|h - \hat{h}\right\|_{\infty,\mathcal{S}} \leq \epsilon. \tag{9}$$

We then specializes Theorem 4.1 for our two-layer quadratic-ReLU network. We make a modification to the set $\mathcal{R}(\mathbf{W}_0, B)$ defined in Equation (2) in order to match the previous approximation results, which is

$$\hat{\mathcal{R}}(\mathbf{W}_0, B) = \left\{\mathbf{W} : \|\mathbf{W} - \mathbf{W}(0)\|_F \leq B\right\}. \tag{10}$$

Due to this modification and that for two-layer the projection step to the set $\hat{\mathcal{R}}$ is unnecessary, we provide a full proof in Appendix C.

Theorem 5.3 (Convergence of Gradient Descent for Optimizing Surrogate Loss For Two-layer Networks)**.** *Suppose the input $\mathbf{x}$ is bounded, and the loss function satisfies Assumption 3.2. For the two-layer network defined in Equation (5), if we run projected gradient descent based on the convex constraint set $\hat{\mathcal{R}}(\mathbf{W}_0, B)$ with a small stepsize α, then for any $\epsilon > 0$, if $m = \Omega\left(\frac{B^4 n}{\epsilon^2}\right)$, we have*

$$\min_{k=1,\cdots,T} L_{\mathcal{A}}(\mathbf{W}^k) - L_*(\mathbf{W}^*) \leq \epsilon, \tag{11}$$

where $\mathbf{W}^* = \min_{\mathbf{W} \in \hat{\mathcal{R}}(\mathbf{W}_0, B)} L_*(\mathbf{W})$ and $T = \Omega(\frac{B^2}{\epsilon\alpha})$.

Then, we can get an overall theorem for the quadratic-ReLU network which is similar to Corollary 5.1 but with explicit ϵ dependence:

Corollary 5.2 (Adversarial Training Finds a Network of Small Robust Train Loss for Quadratic-ReLU Network)**.** *Given data set on the unit sphere equipped with a compatible perturbation set function and an associated perturbation function $\mathcal{A}$, which also takes value on the unit sphere. Suppose Assumption 3.1, 3.2, 5.2 are satisfied. Then for any B and any 2-layer quadratic-ReLU network with width $m = \Omega(\frac{B^4 C'_{\mathcal{D},\mathcal{B}}}{\epsilon^{d+1}} \log \frac{1}{\epsilon})$ (where $C'_{\mathcal{D},\mathcal{B}}$ is a constant that only depends on the dataset $\mathcal{D}$ and perturbation $\mathcal{B}$), if we run projected gradient descent with stepsize α on $\mathcal{R}(\mathbf{W}_0, B)$ for $T = \Omega(\frac{B^2}{\epsilon\alpha})$ steps, then with probability 0.99,*

$$\min_{k=1,\cdots,T} L_{\mathcal{A}}(\mathbf{W}^k) \leq \epsilon. \tag{12}$$

6 Capacity Requirement of Robustness

In this section, we will show that in order to achieve adversarially robust interpolation (which is formally defined below), one needs more capacity than just normal interpolation. In fact, empirical evidence have already shown that to reliably withstand strong adversarial attacks, networks require a significantly larger capacity than for correctly classifying benign examples only [34]. This implies, in some sense, that using a neural network with larger width is necessary.

Let $\mathcal{S}_\delta = \{(\mathbf{x}_1, \cdots, \mathbf{x}_n) \in \mathcal{X}^n : \|\mathbf{x}_i - \mathbf{x}_j\|_2 > 2\delta\}$ and $\mathcal{B}_\delta(x) = \{\mathbf{x}' : \|\mathbf{x}' - \mathbf{x}\|_2 \leq \delta\}$, where δ is a constant, we will consider each n data in $\mathcal{S}_\delta$ and use $\mathcal{B}_\delta$ as the perturbation set function in this section.

We begin with the definition of the interpolation class and the robust interpolation class.

Definition 6.1 (Interpolation class)**.** *We say that a function class $\mathcal{F}$ of functions $f : \mathbb{R}^d \to \{1, -1\}$*[4]*is an n-interpolation class, if the following is satisfied:*

$$\begin{aligned} &\forall(\mathbf{x}_1, \cdots, \mathbf{x}_n) \in \mathcal{S}_\delta, \forall(y_1, \cdots, y_n) \in \{\pm 1\}^n, \\ &\exists f \in \mathcal{F}, \text{ s.t. } f(x_i) = y_i, \forall i \in [n]. \end{aligned}$$

[4]Here we let the classification output be ± 1, and a usual classifier f outputing a number in $\mathbb{R}$ is treated as $\mathrm{sign}(f)$ here.

Definition 6.2 (Robust interpolation class)**.** *We say that a function class $\mathcal{F}$ is an n-robust interpolation class, if the following is satisfied:*

$$\forall(\mathbf{x}_1, \cdots, \mathbf{x}_n) \in \mathcal{S}_\delta, \forall(y_1, \cdots, y_n) \in \{\pm 1\}^n,$$
$$\exists f \in \mathcal{F}, s.t. f(\mathbf{x}_i') = y_i, \forall \mathbf{x}_i' \in \mathcal{B}_\delta(\mathbf{x}_i), \forall i \in [n].$$

We will use the VC-Dimension of a function class $\mathcal{F}$ to measure its complexity. In fact, as shown in [6] (Equation(2)), for neural networks there is a tight connection between the number of parameters W, the number of layers H and their VC-Dimension $\Omega(HW \log(W/L)) \leq$ VC-Dimension $\leq O(HW \log W)$. In addition, combining with the results in [52] (Theorem 3) which shows the existence of a 4-layer neural network with $O(n)$ parameters that can interpolate any n data points, i.e. an interpolation class, we have that an n-interpolation class can be realized by a fixed depth neural network with VC-Dimension upper bound

$$\text{VC-Dimension} \leq O(n \log n). \tag{13}$$

For a general hypothesis class $\mathcal{F}$, we can evidently see that when $\mathcal{F}$ is an n-interpolation class, $\mathcal{F}$ has VC-Dimension at least n. For a neural network that is an n-interpolation class, without further architectural constraints, this lower bound of its VC-dimension is tight up to logarithmic factors as indicated in 13. However, we show that for a robust-interpolation class we will have a much larger VC-Dimension lower bound:

Theorem 6.1. *If $\mathcal{F}$ is an n-robust interpolation class. Then we have lower bound on the VC-Dimension of $\mathcal{F}$*

$$\textit{VC-Dimension} \geq \Omega(nd), \tag{14}$$

where d is the dimension of the input space.

For neural networks, Equation (14) shows that any architecture that is an n-robust interpolation class should have VC-Dimension at least $\Omega(nd)$. Comparing with Equation (13) which shows n-interpolation class can be realized by a network architecture with VC-Dimension $O(n \log n)$, we can conclude that robust interpolation by neural networks needs more capacity, so increasing the width of neural network is indeed necessary.

7 Discussion

This work provides a theoretical analysis of the empirically successful adversarial training algorithm in the training of robust neural networks. Our main results indicate that adversarial training will find a network of low robust surrogate loss, even when the maximization is computed via a heuristic algorithm such as projected gradient ascent. We feel these results lead to several thought-provoking future steps. Can we ensure the robust surrogate loss is low with respect to a larger family of perturbation functions than that used during training? It is natural to ask whether the depth dependence can be improved to $\mathrm{poly}(H)$ using the tools of [1], and whether the projection step can be removed as it is empirically unnecessary and also unnecessary for our analysis for $H = 1$. On the expressiveness side, the current argument utilizes that a neural net restricted to a local region can approximate its induced RKHS. Although the RKHS is universal, they do not avoid the curse of dimension, so it is natural to ask whether the robust expressivity of neural networks can adapt to structure such as low latent dimension of the data mechanism [18, 50]. Since this question is largely unanswered even for neural nets in the non-robust setting, we leave it to future work.

References

[1] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. *arXiv preprint arXiv:1811.03962*, 2018.

[2] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*, 2019.

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

[4] Francis Bach. Breaking the curse of dimensionality with convex neural networks. *The Journal of Machine Learning Research*, 18(1):629–681, 2017.

[5] Francis Bach. On the equivalence between kernel quadrature rules and random feature expansions. *The Journal of Machine Learning Research*, 18(1):714–751, 2017.

[6] 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.

[7] Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

[8] Alberto Bietti and Julien Mairal. On the inductive bias of neural tangent kernels. *arXiv preprint arXiv:1905.12173*, 2019.

[9] Stéphane Boucheron, Gábor Lugosi, and Olivier Bousquet. Concentration inequalities. *Advanced lectures on machine learning*, pages 208–240, 2004.

[10] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017.

[11] Sébastien Bubeck, Eric Price, and Ilya Razenshteyn. Adversarial examples from computational constraints. *arXiv preprint arXiv:1805.10204*, 2018.

[12] Tianle Cai, Ruiqi Gao, Jikai Hou, Siyu Chen, Dong Wang, Di He, Zhihua Zhang, and Liwei Wang. A gram-gauss-newton method learning overparameterized deep neural networks for regression problems. *arXiv preprint arXiv:1905.11675*, 2019.

[13] Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. *arXiv preprint arXiv:1905.13210*, 2019.

[14] 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.

[15] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26. ACM, 2017.

[16] Amit Daniely. SGD learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems*, pages 2422–2430, 2017.

[17] Amit Daniely, Roy Frostig, and Yoram Singer. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pages 2253–2261, 2016.

[18] Simon S Du and Jason D Lee. On the power of over-parametrization in neural networks with quadratic activation. *arXiv preprint arXiv:1803.01206*, 2018.

[19] Simon S Du, Jason D Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. *arXiv preprint arXiv:1811.03804*, 2018.

[20] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

[21] 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.

[22] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning models. *arXiv preprint arXiv:1707.08945*, 2017.

[23] Alon Gonen and Elad Hazan. Learning in non-convex games with an optimization oracle. *arXiv preprint arXiv:1810.07362*, 2018.

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

[25] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[26] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2142–2151, 2018.

[27] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572*, 2018.

[28] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016.

[29] Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. *arXiv preprint arXiv:1808.01204*, 2018.

[30] Xuanqing Liu, Minhao Cheng, Huan Zhang, and Cho-Jui Hsieh. Towards robust neural networks via random self-ensemble. In *European Conference on Computer Vision*, pages 381–397. Springer, 2018.

[31] Xuanqing Liu and Cho-Jui Hsieh. Rob-gan: Generator, discriminator, and adversarial attacker. In *CVPR*, 2019.

[32] Tiange Luo, Tianle Cai, Mengxiao Zhang, Siyu Chen, and Liwei Wang. RANDOM MASK: Towards robust convolutional neural networks, 2019.

[33] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Michael E Houle, Grant Schoenebeck, Dawn Song, and James Bailey. Characterizing adversarial subspaces using local intrinsic dimensionality. *arXiv preprint arXiv:1801.02613*, 2018.

[34] 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.

[35] Charles A Micchelli, Yuesheng Xu, and Haizhang Zhang. Universal kernels. *Journal of Machine Learning Research*, 7(Dec):2651–2667, 2006.

[36] Mehryar Mohri and Andres Munoz Medina. New analysis and algorithm for learning with drifting distributions. In *Algorithmic Learning Theory*, pages 124–138. Springer, 2012.

[37] Vern I Paulsen and Mrinal Raghupathi. *An introduction to the theory of reproducing kernel Hilbert spaces*, volume 152. Cambridge University Press, 2016.

[38] Ali Rahimi and Benjamin Recht. Uniform approximation of functions with random bases. In *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, pages 555–561. IEEE, 2008.

[39] Hadi Salman, Greg Yang, Huan Zhang, Cho-Jui Hsieh, and Pengchuan Zhang. A convex relaxation barrier to tight robust verification of neural networks. *arXiv preprint arXiv:1902.08722*, 2019.

[40] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-GAN: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

[41] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems*, pages 5014–5026, 2018.

[42] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *arXiv preprint arXiv:1904.12843*, 2019.

[43] Gagandeep Singh, Timon Gehr, Matthew Mirman, Markus Püschel, and Martin Vechev. Fast and effective robustness certification. In *Advances in Neural Information Processing Systems*, pages 10802–10813, 2018.

[44] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

[45] 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.

[46] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *International Conference on Machine Learning*, pages 6586–6595, 2019.

[47] Tsui-Wei Weng, Huan Zhang, Hongge Chen, Zhao Song, Cho-Jui Hsieh, Luca Daniel, Duane Boning, and Inderjit Dhillon. Towards fast computation of certified robustness for relu networks. In *International Conference on Machine Learning*, pages 5273–5282, 2018.

[48] 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.

[49] Eric Wong, Frank Schmidt, Jan Hendrik Metzen, and J Zico Kolter. Scaling provable adversarial defenses. In *Advances in Neural Information Processing Systems*, pages 8400–8409, 2018.

[50] Dmitry Yarotsky. Optimal approximation of continuous functions by very deep relu networks. *arXiv preprint arXiv:1802.03620*, 2018.

[51] Dong Yin, Kannan Ramchandran, and Peter Bartlett. Rademacher complexity for adversarially robust generalization. *arXiv preprint arXiv:1810.11914*, 2018.

[52] Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. Finite sample expressive power of small-width relu networks. *arXiv preprint arXiv:1810.07770*, 2018.

[53] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. You only propagate once: Painless adversarial training using maximal principle. *arXiv preprint arXiv:1905.00877*, 2019.

[54] 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*, pages 4939–4948, 2018.

[55] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep ReLU networks. *arXiv preprint arXiv:1811.08888*, 2018.

A Proof of Convergence Results for Deep Nets in Section 4

Proof of Theorem 4.1. Denote $d_k = \|\mathbf{W}^k - \mathbf{W}^*\|_F$. We will perform T steps of projected gradient descent with step size α and then stop.

For projected gradient descent, $\mathbf{W}^k \in \mathcal{R}(\mathbf{W}_0, B)$ holds for all $k = 1, \cdots, T$. Recall the update rule of projected gradient descent is $\mathbf{W}_{k+1} = \mathcal{P}_{\mathcal{R}(\mathbf{W}_0,B)}\left(\mathbf{W}_k - \alpha\nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}_k)\right)$. We have

$$\begin{aligned} d_{k+1}^2 &= \|\mathbf{W}^{k+1} - \mathbf{W}^*\|_F^2 \\ &\le \|\mathbf{V}^{k+1} - \mathbf{W}^*\|_F^2 \\ &= \|\mathbf{W}^k - \mathbf{W}^*\|_F^2 + 2(\mathbf{V}^{k+1} - \mathbf{W}^k)\cdot(\mathbf{W}^k - \mathbf{W}^*) + \|\mathbf{V}^{k+1} - \mathbf{W}^k\|_F^2 \\ &= d_k^2 + 2\alpha\nabla_{\mathbf{W}} L(\mathbf{W}^k, \mathcal{A}(\mathbf{W}^k, \mathbf{x}))\cdot(\mathbf{W}^* - \mathbf{W}^k) + \alpha^2\|\nabla_{\mathbf{W}} L(\mathbf{W}^k, \mathcal{A}(\mathbf{W}^k, \mathbf{x}))\|_F^2, \end{aligned} \tag{15}$$

where in the first inequality we use that fact that when we project a point onto $\mathcal{R}(\mathbf{W}_0, B)$, we move closer to every point in $\mathcal{R}(\mathbf{W}_0, B)$, and in particular, any optimal point. Now we need to analyze the gradient $\nabla_{\mathbf{W}} L(\mathbf{W}^k, \mathcal{A}(\mathbf{W}^k, \mathbf{x}))$. To simplify notations, we define

$$\begin{aligned} f'(\mathbf{W}, \mathbf{x}) = \frac{\partial f(\mathbf{W}, \mathbf{x})}{\partial \mathbf{W}}, &\quad f'^{(h)}(\mathbf{W}, \mathbf{x}) = \frac{\partial f(\mathbf{W}, \mathbf{x})}{\partial \mathbf{W}^{(h)}}, \\ L'_{\mathcal{A}}(\mathbf{W}) = \nabla_{\mathbf{W}} L(\mathbf{W}^k, \mathcal{A}(\mathbf{W}^k, \mathbf{x})), &\quad L'^{(h)}_{\mathcal{A}}(\mathbf{W}) = \nabla_{\mathbf{W}^{(h)}} L(\mathbf{W}^k, \mathcal{A}(\mathbf{W}^k, \mathbf{x})). \end{aligned}$$

where $\nabla_{\mathbf{W}^{(h)}} L$ is the derivative to $\mathbf{W}^{(h)}$ in the first argument of L.

Note that

$$L'^{(h)}_{\mathcal{A}}(\mathbf{W}^k) = \frac{1}{n}\sum_{i=1}^{n} l'(f(\mathbf{W}^k, \widehat{\mathbf{x}}_i^k), y_i) f'^{(h)}(\mathbf{W}^k, \widehat{\mathbf{x}}_i^k),$$

where $\widehat{\mathbf{x}}_i^k = \mathcal{A}(\mathbf{W}^k, \mathbf{x}_i)$. Since the loss function l is Lipschitz, we know $|l'| \le 1$, we have

$$\begin{aligned} \left\| L'^{(h)}_{\mathcal{A}}(\mathbf{W}^k) \right\|_F &\le \max_{i\in[n]} \left\| f'^{(h)}(\mathbf{W}^k, \widehat{\mathbf{x}}_i^k) \right\|_F \\ &\le \max_{i\in[n]} \left\| \left(\frac{c_\sigma}{m}\right)^{\frac{H-h+1}{2}} \widehat{\mathbf{x}}_i^{k,(h-1)} \left(\mathbf{a}^\top \left(\prod_{l=h+1}^{H} \mathbf{J}_i^{k,(l)} \mathbf{W}^{k,(l)} \right) \mathbf{J}_i^{k,(h)} \right) \right\|_F, \end{aligned}$$

where $\mathbf{J}_i^{(l)}$ is a diagonal matrix whose i-th diagonal entry is $\sigma'(w_r^{(h)} \cdot \widehat{\mathbf{x}}_i^{(h-1)})$.

To bound the RHS, note that he definition of $\mathcal{R}(\mathbf{W}_0, B)$ implies that $\left\|\mathbf{W}^{k,(h)} - \mathbf{W}_0^{(h)}\right\|_F \leq B$. According to Lemma B.1, B.3 and G.2 in [19], with probability 0.99, we have for all $i \in [n], h \in [H]$, $\|\mathbf{W}^{0,(h)}\|_2 = O(1)$, $\left\|\mathbf{x}_i^{0,(h)}\right\|_2 = O(1)$ and $\left\|\mathbf{x}_i^{k,(h)} - \mathbf{x}_i^{0,(h)}\right\|_2 \leq \frac{2^{O(H)}B}{\sqrt{m}}$. Therefore, under our choice of m, we have

$$\begin{aligned}\left\|\mathbf{x}_i^{k,(h)}\right\|_2 &= O(1), \\ \frac{\left\|\mathbf{W}^{k,(h)}\right\|_2}{\sqrt{m}} &= O(1).\end{aligned}$$

Also note that by the Lipschitz-ness of our neural network, it is easy to show $\left\|\widehat{\mathbf{x}}_i^{k,(h)} - \mathbf{x}_i^{k,(h)}\right\|_2 \leq e \cdot 2^{O(H)}$, which implies

$$\left\|\widehat{\mathbf{x}}_i^{k,(h)}\right\|_2 = O(1) + e \cdot 2^{O(H)} = 2^{O(H)}.$$

Recall $\left\|\mathbf{J}_i^{k,(h)}\right\|_2 = O(1)$ due to the Lipschitzness of our activation function, we have

$$\left\|L_{\mathcal{A}}^{'(h)}(\mathbf{W}^k)\right\|_F = 2^{O(H)}.$$

Thus

$$\|\nabla_{\mathbf{W}} L_{\mathcal{A}}\|_F^2 \leq \sum_{h=1}^{H} \left\|L_{\mathcal{A}}^{'(h)}(\mathbf{W}^k)\right\|_F^2 = 2^{O(H)}.$$

which gives the bound of the third term of Equation 15. Now we are going to bound the second term of Equation 15. Note that letting $\Delta\mathbf{W} = \mathbf{W}^* - \mathbf{W}^k$, we have

$$\begin{aligned}&L_{\mathcal{A}}(\mathbf{W}^*) - L_{\mathcal{A}}(\mathbf{W}^k) = \int_{t=0}^{1} \left\langle \Delta\mathbf{W}, \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k + t\Delta\mathbf{W}) \right\rangle dt \\ &= \left\langle \Delta\mathbf{W}, \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k) \right\rangle + \\ &\int_{t=0}^{1} \left\langle \Delta\mathbf{W}, \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k + t\Delta\mathbf{W}) - \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k) \right\rangle dt \\ &\geq \left\langle \Delta\mathbf{W}, \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k) \right\rangle \\ &- \sum_{h=1}^{H} \left\|\Delta\mathbf{W}^{(h)}\right\|_F \max_{0\leq t\leq 1} \left\|\nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^{k,(h)} + t\Delta\mathbf{W}^{(h)}) - \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^{k,(h)})\right\|_F,\end{aligned}$$

We use $\mathbf{W}^s$ to denote $\mathbf{W}^k + s\Delta\mathbf{W}$, then

$$\begin{aligned}
&\left\| f'^{(h)}(\mathbf{W}^k, \widehat{\mathbf{x}}_i^k) - f'^{(h)}(\mathbf{W}^s, \widehat{\mathbf{x}}_i^k) \right\|_F \\
=& \left(\frac{c_\sigma}{m}\right)^{\frac{H-h+1}{2}} \left\| \widehat{\mathbf{x}}_i^{k,(h-1)} \left(\mathbf{a}^\top \left(\prod_{l=h+1}^{H} \mathbf{J}_i^{k,(l)} \mathbf{W}^{k,(l)} \right) \mathbf{J}_i^{k,(h)} \right) \right. \\
& \left. - \widehat{\mathbf{x}}_i^{s,(h-1)} \left(\mathbf{a}^\top \left(\prod_{l=h+1}^{H} \mathbf{J}_i^{s,(l)} \mathbf{W}^{s,(l)} \right) \mathbf{J}_i^{s,(h)} \right) \right\|_F .
\end{aligned}$$

Note that both $\mathbf{W}^k, \mathbf{W}^s \in \mathcal{R}(\mathbf{W}_0, B)$, again using Lemma B.3 in [19], we have

$$\begin{aligned}
\left\| \mathbf{W}^{k,(l)} - \mathbf{W}^{s,(l)} \right\|_F \leq & 2B, \\
\left\| \widehat{\mathbf{x}}_i^{k,(h-1)} - \widehat{\mathbf{x}}_i^{s,(h-1)} \right\|_2 \leq & O(\frac{2^{O(H)}B}{\sqrt{m}}).
\end{aligned}$$

Recall $\left\| w_r^{k,(h)} - w_r^{s,(h)} \right\|_2 \leq \frac{2B}{\sqrt{m}}$, it is easy to show

$$|\sigma'(w_r^{k,(h)} \cdot \mathbf{x}_i^{k,(h-1)}) - \sigma'(w_r^{s,(h)} \cdot \mathbf{x}_i^{s,(h-1)})| = O(\frac{2^{O(H)}B}{\sqrt{m}}),$$

by the definition of $\mathbf{J}_i^{(l)}$, we know,

$$\left\| \mathbf{J}_i^{k,(l)} - \mathbf{J}_i^{s,(l)} \right\|_2 \leq O(\frac{2^{O(H)}B}{\sqrt{m}}).$$

Thus, according to Lemma G.1 in [19], we have

$$\left\| f'^{(h)}(\mathbf{W}^k, \widehat{\mathbf{x}}_i^k) - f'^{(h)}(\mathbf{W}^s, \widehat{\mathbf{x}}_i^k) \right\|_F \leq O(\frac{2^{O(H)}B}{\sqrt{m}}),$$

which implies

$$\left\langle \Delta\mathbf{W}, \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k) \right\rangle \leq L_{\mathcal{A}}(\mathbf{W}^*) - L_{\mathcal{A}}(\mathbf{W}^k) + O(\frac{2^{O(H)}B^2}{\sqrt{m}}).$$

Thus, let $D_k = \min_{i\in[k]}(L_{\mathcal{A}}(\mathbf{W}^i) - L_*(\mathbf{W}^*))$, we have

$$d_{k+1}^2 = d_k^2 - 2\alpha D_k + \alpha \cdot O(\frac{2^{O(H)}B^2}{\sqrt{m}}) + \alpha^2 O(2^{O(H)}).$$

Recall that $d_{k+1} < 2B$, we have

$$D_T = O(\frac{B^2}{\alpha T} + \frac{2^{O(H)}B^2}{\sqrt{m}} + \frac{\alpha 2^{O(H)}}{T} + \alpha 2^{O(H)}).$$

Choosing $T = \Omega(\frac{B^2}{\epsilon\alpha})$ and $\alpha = 2^{-\Omega(H)}$, under the choice of m, we complete the proof.

B Proof of Gradient Descent Finding Robust Classifier in Section 5

B.1 Proof of Theorem 5.1

As discussed in Section 5.1, we will use the idea of random feature [38] to approximate $g \in \mathcal{H}(K_\sigma)$ on the unit sphere. We consider functions of the form

$$h(\mathbf{x}) = \int_{\mathbb{R}^d} c(\mathbf{w})^\top \mathbf{x}\sigma'(\mathbf{w}^\top \mathbf{x})d\mathbf{w},$$

where $c(\mathbf{w}) : \mathbb{R}^d \to \mathbb{R}^d$ is any function from $\mathbb{R}^d$ to $\mathbb{R}^d$. We define the RF-norm of h as $\|h\|_{\text{RF}} = \sup_{\mathbf{w}} \frac{\|c(\mathbf{w})\|_2}{p_0(\mathbf{w})}$ where $p_0(\mathbf{w})$ is the probability density function of $\mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I}_d)$, which is the distribution of initialization. Define the function class with finite $\mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I}_d)$-norm as $\mathcal{F}_{\text{RF}} = \left\{h(\mathbf{x}) = \int_{\mathbb{R}^d} c(\mathbf{w})^\top \mathbf{x}\sigma'(\mathbf{w}^\top \mathbf{x})d\mathbf{w} : \|h\|_{\text{RF}} < \infty\right\}$. We firstly show that $\mathcal{F}_{\text{RF}}$ is dense in $\mathcal{H}(K_\sigma)$.

Lemma B.1 (Universality of $\mathcal{F}_{\text{RF}}$)**.** *Let $\mathcal{F}_{RF}$ and $\mathcal{H}(K_\sigma)$ be defined as above. Then $\mathcal{F}_{RF}$ is dense in $\mathcal{H}(K_\sigma)$ w.r.t. $\|\cdot\|_{\infty,\mathcal{S}}$, where $\|f\|_{\infty,\mathcal{S}} = \sup_{\mathbf{x}\in\mathcal{S}} |f(\mathbf{x})|$.*

Proof. Observe that by the definition of the RKHS introduced by K_σ, functions with form $h(\mathbf{x}) = \sum_t a_t K(\mathbf{x}, \mathbf{x}_t), \quad \mathbf{x}_t \in \mathcal{S}$ are dense in $\mathcal{H}(K_\sigma)$. But these functions can also be written in the form $h(\mathbf{x}) = \int_{\mathbb{R}^d} c(\mathbf{w})^\top \mathbf{x}\sigma'(\mathbf{w}^\top \mathbf{x})d\mathbf{w}$ where $c(\mathbf{w}) = p_0(\mathbf{w}) \sum_t a_t \mathbf{x}_t \sigma'(\mathbf{w}^\top \mathbf{x}_t)$. Notice that $\|c(\mathbf{w})\|_2 \le p(\mathbf{w}) \sum_t \left\|a_t \mathbf{x}_t \sigma'(\mathbf{w}^\top \mathbf{x}_t)\right\|_2 < \infty$ since $\mathcal{S}$ is a compact set and σ' is continuous, this verifies that h is an element in $\mathcal{F}_{\text{RF}}$. So $\mathcal{F}_{\text{RF}}$ contains a dense set of $\mathcal{H}(K_\sigma)$ and therefore dense in $\mathcal{H}(K_\sigma)$. Then notice that the evaluation operator $K_{\sigma,\mathbf{x}}$ is uniformly bounded for $\mathbf{x} \in \mathcal{S}$, so $\|\cdot\|_{\infty,\mathcal{S}}$ can be controlled by the RKHS norm and therefore complete the proof. ☐

We then show that we can approximate elements of $\mathcal{F}_{\text{RF}}$ by finite random features. Our results are inspired by [38]. For the next theorem, recall Assumption 3.1, the constant C satisfies σ' is C-Lipschitz, $|\sigma'(\cdot)| \le C$.

Proposition B.1 (Approximation by finite sum)**.** *Let* $h(\mathbf{x}) = \int_{\mathbb{R}^d} c(\mathbf{w})^\top \mathbf{x}\sigma'(\mathbf{w}^\top \mathbf{x})d\mathbf{w} \in \mathcal{F}_{RF}$. $\sigma(0) = 0$. *Then for any* $\delta > 0$*, with probability at least* $1-\delta$ *over* $\mathbf{w}_1, \cdots, \mathbf{w}_M$ *drawn i.i.d. from* $\mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I}_d)$*, there exists* $c_1, \cdots, c_M$ *where* $c_i \in \mathbb{R}^d$ *and* $\|c_i\|_2 \leq \frac{\|h\|_{RF}}{M}$*, so that the function* $\hat{h} = \sum_{i=1}^M c_i^\top \mathbf{x}\sigma'(\mathbf{w}_i^\top \mathbf{x})$*, satisfies*

$$\left\|\hat{h} - h\right\|_{\infty,\mathcal{S}} \leq \frac{\|h\|_{RF}}{\sqrt{M}}\left(4C + C\log\frac{1}{\delta}\right).$$

Proof. Firstly, we construct $\hat{h}$ with $c_i = \frac{c(\mathbf{w}_i)}{Mp_0(\mathbf{w}_i)}$. We first notice that $\|c_i\|_2 = \frac{\|c(\mathbf{w}_i)\|_2}{Mp_0(\mathbf{w}_i)} \leq \frac{\|h\|_{\mathrm{RF}}}{M}$ which satisfies the condition of the theorem. We then define the random variable

$$v(\mathbf{w}_1, \cdots, \mathbf{w}_M) = \left\|\hat{h} - h\right\|_{\infty,\mathcal{S}}.$$

We bound this deviation from its expectation using McDiarmid's inequality.

To do so, we should first show that v is robust to the perturbation of one of its arguments. In fact, for $\mathbf{w}_1, \cdots, \mathbf{w}_M$ and $\tilde{\mathbf{w}}_i$ we have

$$\begin{aligned}
&|v(\mathbf{w}_1, \cdots, \mathbf{w}_M) - v(\mathbf{w}_1, \cdots, \tilde{\mathbf{w}}_i, \cdots, \mathbf{w}_M|) \\
=&\frac{1}{M}\left\|\frac{c(\mathbf{w}_i)^\top \mathbf{x}\sigma'(\mathbf{w}_i^\top \mathbf{x})}{p_0(\mathbf{w}_i)} - \frac{c(\tilde{\mathbf{w}}_i)^\top \mathbf{x}\sigma'(\tilde{\mathbf{w}}_i{}^\top \mathbf{x})}{p_0(\tilde{\mathbf{w}}_i)}\right\|_{\infty,\mathcal{S}} \\
\leq&\frac{1}{M}\|h\|_{\mathrm{RF}}\left(\left\|\sigma'(\mathbf{w}_i^\top \mathbf{x})\right\|_{\infty,\mathcal{S}} + \left\|\sigma'(\tilde{\mathbf{w}}_i{}^\top \mathbf{x})\right\|_{\infty,\mathcal{S}}\right) \\
\leq&\frac{2C\|h\|_{\mathrm{RF}}}{M},
\end{aligned}$$

by using triangle, Cauchy-Schwartz inequality and noticing that $|\sigma'(\cdot)|$ is bounded and $\|\mathbf{x}\|_2 = 1$. We call the final quantity ξ.

Next, we bound the expectation of v. First, observe that the choice of $\mathbf{w}_1, \cdots, \mathbf{w}_M$ ensures that $\mathbb{E}_w\hat{h} = h$. By symmetrization [36], we have

$$\begin{aligned}
\mathbb{E}v =&\mathbb{E}\sup_{\mathbf{x}\in\mathcal{S}}\left|\hat{h}(\mathbf{x}) - \mathbb{E}\hat{h}(\mathbf{x})\right| \\
\leq&2\mathbb{E}_{\mathbf{w},\epsilon}\sup_{\mathbf{x}\in\mathcal{S}}\left|\sum_{i=1}^M \epsilon_i c_i^\top \mathbf{x}\sigma'(\mathbf{w}_i^\top \mathbf{x})\right|,
\end{aligned}$$

where $\epsilon_1, \cdots, \epsilon_M$ is a sequence of Rademacher random variables.

Since $\left|c_i^\top \mathbf{x}\right| \leq \|c_i\|_2 \leq \frac{\|h\|_{\mathrm{RF}}}{M}$ and σ' is C-Lipschitz, we have that $c_i^\top \mathbf{x}\sigma'(\cdot)$ is $\frac{C\|h\|_{\mathrm{RF}}}{M}$-Lipschitz in the scalar argument and zero when the scalar argument is zero. By Rademacher complexity inequality for Lipschitz function (Thm 12 part (4) in [7]) together with Cauchy-Schwartz, Jensen's inequality, we have

$$\begin{aligned}
\mathbb{E}v \leq & 2\mathbb{E}_{\mathbf{w},\epsilon} \sup_{\mathbf{x}\in\mathcal{S}} \left| \sum_{i=1}^{M} \epsilon_i c_i^\top \mathbf{x} \sigma'(\mathbf{w}_i^\top \mathbf{x}) \right| \\
\leq & \frac{4C\left\|h\right\|_{\mathrm{RF}}}{M} \mathbb{E} \sup_{\mathbf{x}\in\mathcal{S}} \left| \sum_{i=1}^{M} \epsilon_i \mathbf{w}_i^\top \mathbf{x} \right| \\
\leq & \frac{4C\left\|h\right\|_{\mathrm{RF}}}{M} \mathbb{E} \left\| \sum_{i=1}^{M} \epsilon_i \mathbf{w}_i \right\|_2 \\
\leq & \frac{4C\left\|h\right\|_{\mathrm{RF}}}{\sqrt{M}} \sqrt{\mathbb{E}_{\mathbf{w}\sim\mathcal{N}(\mathbf{0},\frac{1}{d}\mathbf{I}_d)} \left\|\mathbf{w}\right\|_2^2},
\end{aligned}$$

call this quantity μ. Then McDiarmid's inequailty implies

$$\mathbb{P}[v \geq \mu + \epsilon] \leq \mathbb{P}[v \geq \mathbb{E}v + \epsilon] \leq \exp(-\frac{2\epsilon^2}{M\xi}).$$

The theorem is proved by solving the ϵ while setting the right hand to the given δ. $\square$

Finally, we construct $\mathbf{W}^*$ within a ball of the initialization $\mathbf{W}_0$ that suffers little robust loss $L_*(\mathbf{W}^*)$. Using the symmetric initialization, we have $f(\mathbf{W}_0, \mathbf{x}) = 0$ for all $\mathbf{x}$. We then use the neural Taylor expansion w.r.t. the parameters:

$$f(\mathbf{W}, \mathbf{W}_0) \approx \underbrace{\frac{1}{\sqrt{2m}} \left(\sum_{i=1}^{m} a_i (\mathbf{w}_i - \mathbf{w}_{i0})^\top \mathbf{x} \sigma'(\mathbf{w}_{i0}^\top \mathbf{x}) + \sum_{i=1}^{m} a_i' (\bar{\mathbf{w}}_i - \bar{\mathbf{w}}_{i0})^\top \mathbf{x} \sigma'(\bar{\mathbf{w}}_{i0}^\top \mathbf{x}) \right)}_{(i)},$$

where we omitted the second order term. The term (i) has the form of the random feature approximation, and so Proposition B.1 can be used to construct a robust interpolant.

In summarize, we have give the entire proof of Theorem 5.1 as follow.

Proof. Let L be the Lipschitz coefficient of the loss function ℓ. $\bar{\epsilon} = \frac{2}{3L}$.

By Assumption 5.1 with $\bar{\epsilon}$, there exists $g_1 \in \mathcal{H}(K_\sigma)$ such that

$$|f(\mathbf{x}) - y_i| \leq \epsilon,$$

for every $\mathbf{x} \in \mathcal{B}(\mathbf{x}_i)$ where $\mathcal{B}(\mathbf{x}_i)$ is the perturbation set.

By Lemma B.1, for $\bar{\epsilon}$ there is $g_2 \in \mathcal{F}_{\text{RF}}$ such that $\|g_1 - g_2\|_{\infty,\mathcal{S}} \leq \bar{\epsilon}$. Then, by Theorem B.1, we have $c_1, \cdots, c_m$ where $c_i \in \mathbb{R}^d$ and

$$\|c_i\|_2 \leq \frac{\|g_2\|_{RF}}{m},$$

such that $g_3 = \sum_{i=1}^m c_i^\top \mathbf{x}\sigma'(\mathbf{w}_i^\top \mathbf{x})$ satisfies

$$\|g_2 - g_3\|_{\infty,\mathcal{S}} \leq \frac{\|g_2\|_{RF}}{\sqrt{m}} \left(4C + C\log\frac{1}{\delta}\right),$$

with probability at least $1-\delta$ on the initialization $\mathbf{w}_i$'s.

We use the neural Taylor expansion w.r.t. the parameters:

$$\begin{aligned}
f(\mathbf{W},\mathbf{x}) &= \underbrace{(\mathbf{W}-\mathbf{W}_0)^\top \nabla_{\mathbf{W}} f(\mathbf{W}_0,\mathbf{x})}_{(i)} + \underbrace{(\mathbf{W}-\mathbf{W}_0)^\top \nabla^2_{\mathbf{W}} f(\mathbf{W}_1,\mathbf{x})(\mathbf{W}-\mathbf{W}_0)}_{(ii)} \quad (16)\\
&= \underbrace{\frac{1}{\sqrt{2m}}\left(\sum_{i=1}^m a_i(\mathbf{w}_i - \mathbf{w}_{i0})^\top \mathbf{x}\sigma'(\mathbf{w}_{i0}^\top \mathbf{x}) + \sum_{i=1}^m a_i'(\bar{\mathbf{w}}_i - \bar{\mathbf{w}}_{i0})^\top \mathbf{x}\sigma'(\bar{\mathbf{w}}_{i0}^\top \mathbf{x})\right)}_{(i)}\\
&\quad + \underbrace{\frac{1}{\sqrt{2m}}\left(\sum_{i=1}^m a_i\sigma''(\mathbf{w}_{i1}^\top \mathbf{x})\left((\mathbf{w}_i - \mathbf{w}_{i0})^\top \mathbf{x}\right)^2 + \sum_{i=1}^m a_i'\sigma''(\bar{\mathbf{w}}_{i1}^\top \mathbf{x})\left((\bar{\mathbf{w}}_i - \bar{\mathbf{w}}_{i0})^\top \mathbf{x}\right)^2\right)}_{(ii)},
\end{aligned} \tag{17}$$

for some $\mathbf{W}_1$.

Then set $\mathbf{w}_i = \mathbf{w}_{i0} + \sqrt{\frac{m}{2}}c_i, \bar{\mathbf{w}}_i = \sqrt{\frac{m}{2}}c_i - \bar{\mathbf{w}}_{i0}$ in Equation 16, we have

$$\begin{aligned}
&\|\mathbf{w}_r - \mathbf{w}_{r0}\|_2 \leq \frac{\|g_2\|_{RF}}{\sqrt{2m}},\\
&\frac{1}{\sqrt{2m}}\left(\sum_{i=1}^m a_i(\mathbf{w}_i - \mathbf{w}_{i0})^\top \mathbf{x}\sigma'(\mathbf{w}_{i0}^\top \mathbf{x}) + \sum_{i=1}^m a_i'(\bar{\mathbf{w}}_i - \bar{\mathbf{w}}_{i0})^\top \mathbf{x}\sigma'(\bar{\mathbf{w}}_{i0}^\top \mathbf{x})\right)\\
=&\frac{1}{\sqrt{2m}}\left(\sum_{i=1}^m a_i\sqrt{\frac{m}{2}}c_i^\top \mathbf{x}\sigma'(\mathbf{w}_{i0}^\top \mathbf{x}) - \sum_{i=1}^m a_i'\sqrt{\frac{m}{2}}c_i^\top \mathbf{x}\sigma'(\bar{\mathbf{w}}_{i0}^\top \mathbf{x})\right)\\
=&\sum_{i=1}^m c_i^\top \mathbf{x}\sigma'(\mathbf{w}_i^\top \mathbf{x})\\
=&g_3,
\end{aligned}$$

So

$$\|f(\mathbf{W},x)-g_3\|_{\infty,\mathcal{S}} =\Big\|\frac{1}{\sqrt{2m}}\Big(\sum_{i=1}^{m} a_i\sigma''(\mathbf{w}_{i1}^\top\mathbf{x})\left((\mathbf{w}_i-\mathbf{w}_{i0})^\top\mathbf{x}\right)^2+ \sum_{i=1}^{m} a_i'\sigma''(\bar{\mathbf{w}}_{i1}^\top\mathbf{x})\left((\bar{\mathbf{w}}_i-\bar{\mathbf{w}}_{i0})^\top\mathbf{x}\right)^2\Big)\Big\|_{\infty,\mathcal{S}} \le\frac{C}{\sqrt{2m}}\|g_2\|_{RF}^2,$$

and therefore

$$\|f(\mathbf{W},x)-g_2\|_{\infty,\mathcal{S}}\le\frac{C}{\sqrt{2m}}\|g_2\|_{RF}^2+\frac{\|g_2\|_{RF}}{\sqrt{m}}\left(4C+C\log\frac{1}{\delta}\right). \tag{18}$$

Finally, set m to be larger than $\frac{C\|g_2\|_{\mathrm{RF}}}{\epsilon}\left(\frac{\|g_2\|_{\mathrm{RF}}}{\sqrt{2}}+4+\log\frac{1}{\delta}\right)$ which makes the left hand in Equation (18) no more than $\bar{\epsilon}$ and let $B_{\mathcal{D},\mathcal{B},\epsilon}$ to be $\sqrt{2}\,\|g_2\|_{\mathrm{RF}}$. Then

$$\begin{aligned}
L_*(\mathbf{W}) =&\frac{1}{2n}\sum_{i=1}^{n}\sup_{\mathbf{x}\in\mathcal{B}(\mathbf{x}_i)}\ell\left(f(\mathbf{W},\mathbf{x}),y_i\right)\\
\le&\frac{1}{2}\sup_{i\in[n],\mathbf{x}\in\mathcal{B}(\mathbf{x}_i)}\ell\left(f(\mathbf{W},\mathbf{x}),y_i\right)\\
\le&\frac{L}{2}\sup_{i\in[n],\mathbf{x}\in\mathcal{B}(\mathbf{x}_i)}\left(|f(\mathbf{W},\mathbf{x})-g_3(\mathbf{x})|+|g_3(\mathbf{x})-g_2(\mathbf{x})|+|g_2(\mathbf{x})-g_1(\mathbf{x})|+|g_1(\mathbf{x})-y_i|\right)\\
\le&\frac{3L}{2}\bar{\epsilon}\\
=&\epsilon.
\end{aligned}$$

The theorem follows by set $\delta=0.01$. $\square$

B.2 Proof of Theorem 5.2

We use the ℓ_2 approximation result in [5] and translate it to an ℓ_∞ approximation result by using Lipshitz continuity. We first state Proposition 1 in [5].

Lemma B.2 (Approximation of unit ball of $\mathcal{H}(K_\sigma)$, Corollary of Proposition 1 in [5])**.** *Let $h\in\mathcal{H}(K_\sigma)$. For $\epsilon>0$, let $d\rho$ be the uniform distribution on $\mathcal{S}$. Let $\mathbf{w}_1,\cdots,\mathbf{w}_M$ be* *sampled i.i.d. from* $\mathcal{N}(\mathbf{0}, \frac{1}{d}\mathbf{I}_d)$ *then for any* $\delta \in (0,1)$*, if*

$$M = \Omega\left(\frac{\|h\|_{\mathcal{H}}^2}{\epsilon}\log\frac{\|h\|_{\mathcal{H}}^2}{\epsilon\delta}\right)$$

with probability at least $1-\delta$*, there exists* $c_1, \cdots, c_M \in \mathbb{R}^d$ *such that* $\hat{h} = \sum_{r=1}^{M} c_r^\top \mathbf{x}\mathrm{ReLU}\left(\mathbf{w}_r^\top \mathbf{x}\right)$ *satisfies*

$$\sum_{r=1}^{M} \|c_r\|_2^2 \leq \frac{4\|h\|_{\mathcal{H}}^2}{M}, \tag{19}$$

$$\left\|h - \hat{h}\right\|_{L_2(d\rho)}^2 = \int_{\mathcal{S}} \left(h - \hat{h}\right)^2 d\rho \leq \epsilon. \tag{20}$$

Then we can give the proof of Theorem 5.2.

Proof. We first show that $\hat{h}$ is $2\|h\|_{\mathcal{H}}$-Lipschitz with high probability. We will use a well-known concentration inequality of the norm of Gaussian vector which state as follow:

Fact B.1. *Let* $z \in \mathbb{R}^{md}$ *be drawn from a centered spherical Gaussian, i.e.* $z \sim \mathcal{N}(0, \sigma^2 I)$ *where* $\sigma > 0$*. Then we have* $\mathbb{P}[\|z\|_2 \geq \sigma\sqrt{md} + t] \leq e^{-t^2/(2\sigma^2)}$.

Proof. We refer the reader to Example 5.7 in [9] for a reference of this standard result. Combined with $\mathbb{E}[\|z\|_2] \leq \sigma\sqrt{md}$, which is obtained from Jensen's Inequality, the aforementioned example gives the desired upper tail bound. $\square$

Let $Lip(f)$ denote the Lipschitz coefficient of f. Then by the property of Lipschitz coefficient, we have

$$
\begin{aligned}
Lip(\hat{h}) =& Lip\left(\sum_{r=1}^{M} c_r^\top \mathbf{x}\mathrm{ReLU}\left(\mathbf{w}_r^\top \mathbf{x}\right)\right) \\
\leq& \sum_{r=1}^{M} Lip\left(c_r^\top \mathbf{x}\mathrm{ReLU}\left(\mathbf{w}_r^\top \mathbf{x}\right)\right) \\
\leq& \sum_{r=1}^{M} \left\|c_r\right\|_2 \left\|\mathbf{x}\right\|_2 Lip\left(\mathrm{ReLU}\left(\mathbf{w}_r^\top \mathbf{x}\right)\right) \\
\leq& \sum_{r=1}^{M} \left\|c_r\right\|_2 \left\|\mathbf{w}_r\right\|_2 \\
\leq& \sqrt{\left(\sum_{r=1}^{M} \left\|c_r\right\|_2^2\right)\left(\sum_{r=1}^{M} \left\|\mathbf{w}_r\right\|_2^2\right)} \\
\leq& \sqrt{\frac{4\left\|h\right\|_{\mathcal{H}}^2}{M}\left(\sum_{r=1}^{M} \left\|\mathbf{w}_r\right\|_2^2\right)} \\
\leq& 2\left\|h\right\|_{\mathcal{H}}\left(1 + O\left(\sqrt{\frac{\log\frac{1}{\delta}}{M}}\right)\right), \qquad \text{(With probability } 1-\delta/2 \text{ by Fact B.1)}
\end{aligned}
$$

which means $\hat{h}$ has finite Lipschitz coefficient and therefore so does $h - \hat{h}$, and the upper bound of Lipschitz constant c_L only depends on the data and the perturbation. Then we can bound the ℓ_∞ approximation error. Suppose for some $\mathbf{x} \in \mathcal{S}$, $\left|h(\mathbf{x}) - \hat{h}(\mathbf{x})\right| > \epsilon$, since $h - \hat{h}$ is Lipschitz, it is not hard to see that, when ϵ is small,

$$
\int_{\mathcal{S}} \left(h - \hat{h}\right)^2 \gtrsim \frac{\pi^{\frac{d}{2}}\epsilon^{d+1}}{\Gamma(d/2+1)c_L^d} \asymp \frac{\epsilon^{d+1}}{c_L^d}\frac{(2\pi e)^{\frac{d}{2}}}{d^{\frac{d+1}{2}}}. \tag{21}
$$

By Lemma B.2, for some constant $C_{\mathcal{D},\mathcal{B}}$, when $M = \Omega\left(\frac{C_{\mathcal{D},\mathcal{B}}}{\epsilon^{d+1}}\log\frac{1}{\epsilon^{d+1}\delta}\right)$, Equation (21) fails, so $\left\|h - \hat{h}\right\|_{\infty,\mathcal{S}} \leq \epsilon$ holds and at the same time we have $\sum_{r=1}^{M}\left\|c_r\right\|_2^2 = O\left(\frac{1}{M}\right)$. $\square$

C A convergence Theorem for Two-Layer Networks Using Gradient Descent without Projection

Theorem C.1 (Convergence of Gradient Descent without Projection for Optimizing Surrogate Loss For Two-layer Networks)**.** *Suppose the input* $\mathbf{x}$ *is bounded, and the loss function satisfies Assumption 3.2. For the two-layer network defined in Equation (5), if we run gradient descent with a small stepsize* α*, then for any* $\epsilon > 0$*, if* $m = \Omega\left(\frac{B^4}{\epsilon^2}\right)$*, we have*

$$\min_{k=1,\cdots,T} L_{\mathcal{A}}(\mathbf{W}^k) - L_*(\mathbf{W}^*) \leq \epsilon, \tag{22}$$

where $\mathbf{W}^* = \min_{\mathbf{W} \in \hat{\mathcal{R}}(\mathbf{W}_0, B)} L_*(\mathbf{W})$ *and* $T = \Omega(\frac{\sqrt{m}}{\alpha})$.

Proof. Now each step we take an update $\mathbf{W}^{k+1} = \mathbf{W}^k - \alpha \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k)$. We can compute an upper bound on the Lipschitz of $\nabla_{\mathbf{W}} f$:

$$\|\nabla_{\mathbf{W}} f(\mathbf{W}) - \nabla_{\mathbf{W}} f(\mathbf{W}')\|_F \leq \frac{\max_r |a_r|}{\sqrt{m}} \|\sigma'(\mathbf{W}\mathbf{x})) - \sigma'(\mathbf{W}'\mathbf{x})\|_2 \|\mathbf{x}\|_2 = O\left(\frac{\|\mathbf{W} - \mathbf{W}'\|_F}{\sqrt{m}}\right),$$

which implies that $f(\mathbf{W})$ is $O\left(\frac{1}{\sqrt{m}}\right)$-weakly convex, i.e. adding a term $O\left(\frac{1}{\sqrt{m}} \|\mathbf{W} - \mathbf{W}'\|_F^2\right)$ would make it convex. Denote $\mathcal{R} = \{\mathbf{W} | \|\mathbf{W} - \mathbf{W}^0\|_F \leq 3B\}$. Since $L_{\mathcal{A}}(\mathbf{W}) = \ell(f(\mathbf{W}))$ and ℓ is convex and has bounded derivative, we know that for $\mathbf{W}, \mathbf{W}' \in \mathcal{R}$,

$$\begin{aligned}
& L_{\mathcal{A}}(\mathbf{W}') - L_{\mathcal{A}}(\mathbf{W}) \\
\geq & \ell'(f(\mathbf{W}))(f(\mathbf{W}') - f(\mathbf{W})) \\
\geq & \ell'(f(\mathbf{W}))(\nabla_{\mathbf{W}} f(\mathbf{W})(\mathbf{W}' - \mathbf{W})) - |\ell'(f(\mathbf{W}))| \cdot O(\frac{1}{\sqrt{m}}) \|\mathbf{W} - \mathbf{W}'\|_F^2 \\
= & \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}) - O(\frac{1}{\sqrt{m}}) \|\mathbf{W} - \mathbf{W}'\|_F^2
\end{aligned}$$

which means $L_{\mathcal{A}}(\mathbf{W})$ is also $O\left(\frac{1}{\sqrt{m}}\right)$-weakly convex.

Denote $d_k = \|\mathbf{W}^k - \mathbf{W}^*\|_F$. There are two situations during the optimization process:

Case 1. $\mathbf{W}^k \in \mathcal{R}$ holds for all $k = 1, \cdots, T$. We have

$$
\begin{aligned}
d_{k+1}^2 &= \|\mathbf{W}^{k+1} - \mathbf{W}^*\|_F^2 \\
&= \|\mathbf{W}^k - \mathbf{W}^*\|_F^2 + 2(\mathbf{W}^{k+1} - \mathbf{W}^k) \cdot (\mathbf{W}^k - \mathbf{W}^*) + \|\mathbf{W}^{k+1} - \mathbf{W}^k\|_F^2 \\
&= d_k^2 + 2\alpha \nabla_{\mathbf{W}} L_{\mathcal{A}}(\mathbf{W}^k) \cdot (\mathbf{W}^* - \mathbf{W}^k) + \alpha^2 \|\nabla_{\mathbf{W}} L_{\mathcal{A}}\|_F^2 \\
&\le d_k^2 + 2\alpha \sum_i [\ell(\mathbf{W}^*, \mathcal{A}(w^k, \mathbf{x}_i)) - \ell(\mathbf{W}^k, \mathcal{A}(\mathbf{W}^k, \mathbf{x}_i)) + O(\frac{1}{\sqrt{m}}) \|\mathbf{W}^* - \mathbf{W}^k\|_F^2] + M\alpha^2 \\
&\le d_k^2 + 2\alpha (L_*(\mathbf{W}^*) - L_{\mathcal{A}}(\mathbf{W}^k)) + O(\frac{1}{\sqrt{m}}) \|\mathbf{W}^* - \mathbf{W}^k\|_F^2) + M\alpha^2 \\
&\le (1 + O(\frac{\alpha}{\sqrt{m}})) d_k^2 - 2\alpha D_k + \alpha^2 M,
\end{aligned}
$$

where the first inequality is based on the above convexity result, and $D_k = \min_{i=1,\cdots,k}(L_{\mathcal{A}}(\mathbf{W}^i) - L_*(\mathbf{W}^*))$. Also, by Lemma G.2 in [19] with probability 0.99 $\|\mathbf{W}^0\|_2 = O(1)$ and then $\|\mathbf{W}\|_2 = O(1)$ for $\mathbf{W} \in \mathcal{R}$, so

$$
M = \max_{\mathbf{W} \in \mathcal{R}} \|\nabla_{\mathbf{W}} L_{\mathcal{A}}\|^2 = \max_{\mathbf{W} \in \mathcal{R}} \|(\mathbf{W}\mathbf{x}) \circ \mathcal{I}(\mathbf{W}\mathbf{x} \ge 0)\|_2^2 \le \max_{\mathbf{W} \in \mathcal{R}} \|\mathbf{W}\mathbf{x}\|_2^2 = O(1).
$$

Let $T_k = (1 + O(\frac{\alpha}{\sqrt{m}}))^k$ which is a geometric series, we have

$$
\begin{aligned}
\frac{d_{k+1}^2}{T_k} &\le \frac{d_k^2}{T_{k-1}} - 2\alpha \frac{D_k}{T_k} + \frac{O(\alpha^2)}{T_k} \le \cdots \le \\
&\le d_1^2 - 2\alpha \sum_{i=1}^k \frac{D_i}{T_i} + O(\alpha^2) \sum_{i=1}^k \frac{1}{T_i} \\
&\le d_1^2 - D_k O(\sqrt{m})(\frac{1}{T_1} - \frac{1}{T_{k+1}}) + O(\alpha\sqrt{m})(\frac{1}{T_1} - \frac{1}{T_{k+1}}),
\end{aligned}
$$

which gives

$$
D_k \le O(\alpha) + \frac{d_1^2 - \frac{d_{k+1}^2}{T_k}}{O(\sqrt{m})(\frac{1}{T_1} - \frac{1}{T_{k+1}})}.
$$

Choosing a small α and and $T = 1/O(\alpha/\sqrt{m})$, we will have $T_1 \approx 1$, $T_K \approx e$. So when α is small enough, we have $D_K = O(\frac{B^2}{\sqrt{m}})$.

Case 2. If $\mathbf{W}^k \notin \mathcal{R}$ happens, say first at step $k+1$, since we have $T_{k+1} = O(1)$, which makes $d_1^2 - \frac{d_{k+1}^2}{T_k} \le B^2 - 4B^2 \le 0$. So we have $D_k \le O(\alpha)$. Then for a small α, $D_k = O(\frac{B^2}{\sqrt{m}})$ still holds.

So in any case the result is correct, thus we have proved the convergence without the need of projection. $\square$

D Proof of Theorem 6.1

Proof. We prove this theorem by an explicit construction of $\left[\frac{n}{2}\right] \times d$ data points that $\mathcal{F}$ is guaranteed to be able to shatter. Consider the following data points

$$\mathbf{x}_{i,j} = \mathbf{c}_i + \epsilon \mathbf{e}_j \text{ for } i \in \{1, \cdots, \left[\frac{n}{2}\right]\}, j \in [d],$$

where $\mathbf{c}_i = (5i\delta, 0, \cdots, 0)^\top \in \mathbb{R}^d$, ϵ is a small constant, and $\mathbf{e}_j = (0, \cdots, 1, \cdots, 0) \in \mathbb{R}_d$ is the j-th unit vector. For any labeling $y_{i,j} \in \{1, -1\}$, we let $P_i = \{j : y_{i,j} = 1\}$, $N_i = \{j : y_{i,j} = -1\}$, and let $\#P_i = k_i$. The idea is that for every cluster of points $\{\mathbf{x}_{i,j}\}_{j=1}^n$, we use 2 disjoint balls with radius δ to separate the positive and negative data points. In fact, for every such cluster, the hyperplane

$$\mathcal{M}_i = \{\mathbf{x} : (y_{i,1}, \cdots, y_{i,d}) \cdot \mathbf{x} = 0\},$$

clearly separates the points into $\{\mathbf{x}_{i,j} : j \in P_i\}$ and $\{\mathbf{x}_{i,j} : j \in N_i\}$. Then we can see easily that there exists a constant $\gamma_{k_i} > 0$ such that for any $r > \gamma_{k_i}\epsilon$, there exist two Euclidean balls $\mathcal{B}_r(\mathbf{x}'_{i,1}), \mathcal{B}_r(\mathbf{x}'_{i,2})$ in $\mathbb{R}^d$ with radius r, such that they contain the set $\{\mathbf{x}_{i,j} : j \in P_i\}$ and $\{\mathbf{x}_{i,j} : j \in N_i\}$ respectively, and that $\mathcal{B}_r(\mathbf{x}'_{i,1})$ and $\mathcal{B}_r(\mathbf{x}'_{i,2})$ are also separated by $\mathcal{M}_i$. Therefore, as long as we take

$$\epsilon < \delta \max\left(\frac{1}{\gamma_1}, \cdots, \frac{1}{\gamma_d}, 1\right),$$

we can always put $r = \delta$. In the case that P_i or N_i is empty, we can simply put one ball centered at $\mathbf{c}_i$ and the other anywhere far away. Such balls are disjoint since $\epsilon \leq \delta$, $\left\|\mathbf{x}'_{i,l} - \mathbf{c}_i\right\|_2 \leq 2\delta$ for $l = 1, 2$, and $\|\mathbf{c}_i - \mathbf{c}_{i'}\| \geq 5\delta$ for $i \neq i'$. In this way, we can use the fact that there exists a function $f \in F$ such that for any $i \in \{1, \cdots, \left[\frac{n}{2}\right]\}$, $f(\mathbf{x}) = 1$ for $\mathbf{x} \in \mathcal{B}_\delta(\mathbf{x}_{i,1})$ and $f(\mathbf{x}) = -1$ for $\mathbf{x} \in \mathcal{B}_\delta(\mathbf{x}_{i,2})$. In this way, $f(\mathbf{x}_{i,j}) = y_{i,j}$ holds for all i, j. Since the labels $y_{i,j}$ can be picked at will, by the definition of the VC-dimension, we know that the VC-dimension of $\mathcal{F}$ is always at least $\left[\frac{n}{2}\right] \times d$. $\square$