Generalization Bounds of Stochastic Gradient Descent for Wide and Deep Neural Networks

Yuan Cao[*] and Quanquan Gu[†]

Abstract

We study the training and generalization of deep neural networks (DNNs) in the over-parameterized regime, where the network width (i.e., number of hidden nodes per layer) is much larger than the number of training data points. We show that, the expected 0-1 loss of a wide enough ReLU network trained with stochastic gradient descent (SGD) and random initialization can be bounded by the training loss of a random feature model induced by the network gradient at initialization, which we call a *neural tangent random feature* (NTRF) model. For data distributions that can be classified by NTRF model with sufficiently small error, our result yields a generalization error bound in the order of $\widetilde{\mathcal{O}}(n^{-1/2})$ that is independent of the network width. Our result is more general and sharper than many existing generalization error bounds for over-parameterized neural networks. In addition, we establish a strong connection between our generalization error bound and the neural tangent kernel (NTK) proposed in recent work.

1 Introduction

Deep learning has achieved great success in a wide range of applications including image processing (Krizhevsky et al., 2012), natural language processing (Hinton et al., 2012) and reinforcement learning (Silver et al., 2016). Most of the deep neural networks used in practice are highly over-parameterized, such that the number of parameters is much larger than the number of training data. One of the mysteries in deep learning is that, even in an over-parameterized regime, neural networks trained with stochastic gradient descent can still give small test error and do not overfit. In fact, a famous empirical study by Zhang et al. (2016) shows the following phenomena:

- Even if one replaces the real labels of a training data set with purely random labels, an over-parameterized neural network can still fit the training data perfectly. However since the labels are independent of the input, the resulting neural network does not generalize to the test dataset.
- If the same over-parameterized network is trained with real labels, it not only achieves small training loss, but also generalizes well to the test dataset.

While a series of recent work has theoretically shown that a sufficiently over-parameterized (i.e., sufficiently wide) neural network can fit random labels (Du et al., 2018b; Allen-Zhu et al., 2018b;

[*] Department of Computer Science, University of California, Los Angeles, CA 90095, USA; e-mail: yuancao@cs.ucla.edu

[†] Department of Computer Science, University of California, Los Angeles, CA 90095, USA; e-mail: qgu@cs.ucla.edu

Du et al., 2018a; Zou et al., 2018), the reason why it can generalize well when trained with real labels is less understood. Existing generalization bounds for deep neural networks (Neyshabur et al., 2015; Bartlett et al., 2017; Neyshabur et al., 2017; Golowich et al., 2017; Dziugaite and Roy, 2017; Arora et al., 2018; Li et al., 2018; Wei et al., 2018; Neyshabur et al., 2018) based on uniform convergence usually cannot provide non-vacuous bounds (Langford and Caruana, 2002; Dziugaite and Roy, 2017) in the over-parameterized regime. In fact, the empirical observation by Zhang et al. (2016) indicates that in order to understand deep learning, it is important to distinguish the true data labels from random labels when studying generalization. In other words, it is essential to quantify the "classifiability" of the underlying data distribution, i.e., how difficult it can be classified.

Certain effort has been made to take the "classifiability" of the data distribution into account for generalization analysis of neural networks. Brutzkus et al. (2017) showed that stochastic gradient descent (SGD) can learn an over-parameterized two-layer neural network with good generalization for linearly separable data. Li and Liang (2018) proved that, if the data satisfy certain structural assumption, SGD can learn an over-parameterized two-layer network with fixed second layer weights and achieve a small generalization error. Allen-Zhu et al. (2018a) studied the generalization performance of SGD and its variants for learning two-layer and three-layer networks, and used the risk of smaller two-layer or three-layer networks with smooth activation functions to characterize the classifiability of the data distribution. There is another line of studies on the algorithm-dependent generalization bounds of neural networks in the over-parameterized regime (Daniely, 2017; Arora et al., 2019b; Cao and Gu, 2019; Yehudai and Shamir, 2019; E et al., 2019), which quantifies the classifiability of the data with a reference function class defined by random features (Rahimi and Recht, 2008, 2009) or kernels[1]. Specifically, Daniely (2017) showed that a neural network of large enough size is competitive with the best function in the conjugate kernel class of the network. Arora et al. (2019b) gave a generalization error bound for two-layer ReLU networks with fixed second layer weights based on a ReLU kernel function. Cao and Gu (2019) showed that deep ReLU networks trained with gradient descent can achieve small generalization error if the data can be separated by certain random feature model (Rahimi and Recht, 2009) with a margin. Yehudai and Shamir (2019) used the expected loss of a similar random feature model to quantify the generalization error of two-layer neural networks with smooth activation functions. A similar generalization error bound was also given by E et al. (2019), where the authors studied the optimization and generalization of two-layer networks trained with gradient descent. However, all the aforementioned results are still far from satisfactory: they are either limited to two-layer networks, or restricted to very simple and special reference function classes.

In this paper, we aim at providing a sharper and generic analysis on the generalization of deep ReLU networks trained by SGD. In detail, we base our analysis upon the key observations that near random initialization, the neural network function is almost a linear function of its parameters and the loss function is locally almost convex. This enables us to prove a cumulative loss bound of SGD, which further leads to a generalization bound by online-to-batch conversion (Cesa-Bianchi et al., 2004). The main contributions of our work are summarized as follows:

- We give a bound on the expected 0-1 error of deep ReLU networks trained by SGD with random initialization. Our result relates the generalization bound of an over-parameterized ReLU network with a random feature model defined by the network gradients, which we call *neural tangent*

[1] Since random feature models and kernel methods are highly related (Rahimi and Recht, 2008, 2009), we group them into the same category. More details are discussed in Section 3.2.

random feature (NTRF) model. It also suggests an algorithm-dependent generalization error bound of order $\widetilde{\mathcal{O}}(n^{-1/2})$, which is independent of network width, if the data can be classified by the NTRF model with small enough error.

- Our analysis is general enough to cover recent generalization error bounds for neural networks with random feature based reference function classes, and provides better bounds. Our expected 0-1 error bound directly covers the result by Cao and Gu (2019), and gives a tighter sample complexity when reduced to their setting, i.e., $\widetilde{\mathcal{O}}(1/\epsilon^2)$ versus $\widetilde{\mathcal{O}}(1/\epsilon^4)$ where ϵ is the target generalization error. Compared with recent results by Yehudai and Shamir (2019); E et al. (2019) who only studied two-layer networks, our bound not only works for deep networks, but also uses a larger reference function class when reduced to the two-layer setting, and therefore is sharper.

- Our result has a direct connection to the neural tangent kernel studied in Jacot et al. (2018). When interpreted in the language of kernel method, our result gives a generalization bound in the form of $\widetilde{\mathcal{O}}(L \cdot \sqrt{\mathbf{y}^\top (\mathbf{\Theta}^{(L)})^{-1}\mathbf{y}/n})$, where $\mathbf{y}$ is the training label vector, and $\mathbf{\Theta}^{(L)}$ is the neural tangent kernel matrix defined on the training input data. This form of generalization bound is similar to, but more general and tighter than the bound given by Arora et al. (2019b).

Notation We use lower case, lower case bold face, and upper case bold face letters to denote scalars, vectors and matrices respectively. For a vector $\mathbf{v} = (v_1, \ldots, v_d)^T \in \mathbb{R}^d$ and a number $1 \leqslant p < \infty$, let $\|\mathbf{v}\|_p = (\sum_{i=1}^d |v_i|^p)^{1/p}$. We also define $\|\mathbf{v}\|_\infty = \max_i |v_i|$. For a matrix $\mathbf{A} = (A_{i,j})_{m\times n}$, we use $\|\mathbf{A}\|_0$ to denote the number of non-zero entries of $\mathbf{A}$, and denote $\|\mathbf{A}\|_F = (\sum_{i,j=1}^d A_{i,j}^2)^{1/2}$ and $\|\mathbf{A}\|_p = \max_{\|\mathbf{v}\|_p=1} \|\mathbf{A}\mathbf{v}\|_p$ for $p \geqslant 1$. For two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$, we define $\langle \mathbf{A}, \mathbf{B} \rangle = \mathrm{Tr}(\mathbf{A}^\top \mathbf{B})$. We denote by $\mathbf{A} \succeq \mathbf{B}$ if $\mathbf{A} - \mathbf{B}$ is positive semidefinite. In addition, we define the asymptotic notations $\mathcal{O}(\cdot)$, $\widetilde{\mathcal{O}}(\cdot)$, $\Omega(\cdot)$ and $\widetilde{\Omega}(\cdot)$ as follows. Suppose that a_n and b_n be two sequences. We write $a_n = \mathcal{O}(b_n)$ if $\limsup_{n\to\infty} |a_n/b_n| < \infty$, and $a_n = \Omega(b_n)$ if $\liminf_{n\to\infty} |a_n/b_n| > 0$. We use $\widetilde{\mathcal{O}}(\cdot)$ and $\widetilde{\Omega}(\cdot)$ to hide the logarithmic factors in $\mathcal{O}(\cdot)$ and $\Omega(\cdot)$.

2 Problem Setup

In this section we introduce the basic problem setup. Following the same standard setup implemented in the line of recent work (Allen-Zhu et al., 2018b; Du et al., 2018a; Zou et al., 2018; Cao and Gu, 2019), we consider fully connected neural networks with width m, depth L and input dimension d. Such a network is defined by its weight matrices at each layer: for $L \geqslant 2$, let $\mathbf{W}_1 \in \mathbb{R}^{m\times d}$, $\mathbf{W}_l \in \mathbb{R}^{m\times m}$, $l = 2, \ldots, L-1$ and $\mathbf{W}_L \in \mathbb{R}^{1\times m}$ be the weight matrices of the network. Then the neural network with input $\mathbf{x} \in \mathbb{R}^d$ is defined as

$$f_{\mathbf{W}}(\mathbf{x}) = \sqrt{m} \cdot \mathbf{W}_L \sigma(\mathbf{W}_{L-1} \sigma(\mathbf{W}_{L-2} \cdots \sigma(\mathbf{W}_1 \mathbf{x}) \cdots)), \tag{2.1}$$

where $\sigma(\cdot)$ is the entry-wise activation function. In this paper, we only consider the ReLU activation function $\sigma(z) = \max\{0, z\}$, which is the most commonly used activation function in applications. It is also arguably one of the most difficult activation functions to analyze, due to its non-smoothess. We remark that our result can be generalized to many other Lipschitz continuous and smooth activation functions. For simplicity, we follow Allen-Zhu et al. (2018b); Du et al. (2018a) and assume that the widths of each hidden layer are the same. Our result can be easily extended to the setting that the widths of each layer are not equal but in the same order, as discussed in Zou et al. (2018); Cao and Gu (2019).

When $L = 1$, the neural network reduces to a linear function, which has been well-studied. Therefore, for notational simplicity we focus on the case $L \geqslant 2$, where the parameter space is defined as

$$\mathcal{W} := \mathbb{R}^{m\times d} \times (\mathbb{R}^{m\times m})^{L-2} \times \mathbb{R}^{1\times m}.$$

We also use $\mathbf{W} = (\mathbf{W}_1, \ldots, \mathbf{W}_L) \in \mathcal{W}$ to denote the collection of weight matrices for all layers. For $\mathbf{W}, \mathbf{W}' \in \mathcal{W}$, we define their inner product as $\langle \mathbf{W}, \mathbf{W}' \rangle := \sum_{l=1}^{L} \mathrm{Tr}(\mathbf{W}_l^\top \mathbf{W}_l')$.

The goal of neural network learning is to minimize the expected risk, i.e.,

$$\min_{\mathbf{W}} L_{\mathcal{D}}(\mathbf{W}) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}} L_{(\mathbf{x},y)}(\mathbf{W}), \tag{2.2}$$

where $L_{(\mathbf{x},y)}(\mathbf{W}) = \ell[y \cdot f_{\mathbf{W}}(\mathbf{x})]$ is the loss defined on any example $(\mathbf{x}, y)$, and $\ell(z)$ is the loss function. Without loss of generality, we consider the cross-entropy loss in this paper, which is defined as $\ell(z) = \log[1 + \exp(-z)]$. We would like to emphasize that our results also hold for most convex and Lipschitz continuous loss functions such as hinge loss. We now introduce stochastic gradient descent based training algorithm for minimizing the expected risk in (2.2). The detailed algorithm is given in Algorithm 1.

Algorithm 1 SGD for DNNs starting at Gaussian initialization

```
Input: Number of iterations $n$, step size $\eta$.
Generate each entry of $\mathbf{W}_l^{(1)}$ independently from $N(0, 2/m)$, $l \in [L-1]$.
Generate each entry of $\mathbf{W}_L^{(1)}$ independently from $N(0, 1/m)$.
for $i = 1, 2, \ldots, n$ do
    Draw $(\mathbf{x}_i, y_i)$ from $\mathcal{D}$.
    Update $\mathbf{W}^{(i+1)} = \mathbf{W}^{(i)} - \eta \cdot \nabla_{\mathbf{W}} L_{(\mathbf{x}_i,y_i)}(\mathbf{W}^{(i)})$.
end for
Output: Randomly choose $\widehat{\mathbf{W}}$ uniformly from $\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(n)}\}$.
```

The initialization scheme for $\mathbf{W}^{(1)}$ given in Algorithm 1 generates each entry of the weight matrices from a zero-mean independent Gaussian distribution, whose variance is determined by the rule that the expected length of the output vector in each hidden layer is equal to the length of the input. This initialization method is also known as He initialization (He et al., 2015). Here the last layer parameter is initialized with variance $1/m$ instead of $2/m$ since the last layer is not associated with the ReLU activation function.

3 Main Results

In this section we present the main results of this paper. In Section 3.1 we give an expected 0-1 error bound against a neural tangent random feature reference function class. In Section 3.2, we discuss the connection between our result and the neural tangent kernel proposed in Jacot et al. (2018).

3.1 An Expected 0-1 Error Bound

In this section we give a bound on the expected 0-1 error $L_{\mathcal{D}}^{0-1}(\mathbf{W}) := \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}}[\mathbb{1}\{y \cdot f_{\mathbf{W}}(\mathbf{x}) < 0\}]$ obtained by Algorithm 1. Our result is based on the following assumption.

Assumption 3.1. The data inputs are normalized: $\|\mathbf{x}\|_2 = 1$ for all $(\mathbf{x}, y) \in \operatorname{supp}(\mathcal{D})$.

Assumption 3.1 is a standard assumption made in almost all previous work on optimization and generalization of over-parameterized neural networks (Du et al., 2018b; Allen-Zhu et al., 2018b; Du et al., 2018a; Zou et al., 2018; Oymak and Soltanolkotabi, 2019; E et al., 2019). As is mentioned in Cao and Gu (2019), this assumption can be relaxed to $c_1 \leqslant \|\mathbf{x}\|_2 \leqslant c_2$ for all $(\mathbf{x}, y) \in \operatorname{supp}(\mathcal{D})$, where $c_2 > c_1 > 0$ are absolute constants.

For any $\mathbf{W} \in \mathcal{W}$, we define its ω-neighborhood as

$$\mathcal{B}(\mathbf{W}, \omega) := \{\mathbf{W}' \in \mathcal{W} : \|\mathbf{W}_l' - \mathbf{W}_l\|_F \leqslant \omega, l \in [L]\}.$$

Below we introduce the neural tangent random feature function class, which serves as a reference function class to measure the "classifiability" of the data, i.e., how easy it can be classified.

Definition 3.2 (Neural Tangent Random Feature)**.** Let $\mathbf{W}^{(1)}$ be generated via the initialization scheme in Algorithm 1. The neural tangent random feature (NTRF) function class is defined as

$$\mathcal{F}(\mathbf{W}^{(1)}, R) = \big\{f(\cdot) = f_{\mathbf{W}^{(1)}}(\cdot) + \langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\cdot), \mathbf{W}\rangle : \mathbf{W} \in \mathcal{B}(\mathbf{0}, R \cdot m^{-1/2})\big\},$$

where $R > 0$ measures the size of the function class, and m is the width of the neural network.

The name "neural tangent random feature" is inspired by the neural tangent kernel proposed by Jacot et al. (2018), because the random features are the gradients of the neural network with random weights. Connections between the neural tangent random features and the neural tangent kernel will be discussed in Section 3.2.

We are ready to present our main result on the expected 0-1 error bound of Algorithm 1.

Theorem 3.3. For any $\delta \in (0, e^{-1}]$ and $R > 0$, there exists

$$m^*(\delta, R, L, n) = \widetilde{\mathcal{O}}\big(\mathrm{poly}(R, L)\big) \cdot n^7 \cdot \log(1/\delta)$$

such that if $m \geqslant m^*(\delta, R, L, n)$, then with probability at least $1 - \delta$ over the randomness of $\mathbf{W}^{(1)}$, the output of Algorithm 1 with step size $\eta = \kappa \cdot R/(m\sqrt{n})$ for some small enough absolute constant κ satisfies

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leqslant \inf_{f \in \mathcal{F}(\mathbf{W}^{(1)}, R)} \left\{\frac{4}{n}\sum_{i=1}^{n} \ell[y_i \cdot f(\mathbf{x}_i)]\right\} + \mathcal{O}\left[\frac{LR}{\sqrt{n}} + \sqrt{\frac{\log(1/\delta)}{n}}\right], \tag{3.1}$$

where the expectation is taken over the uniform draw of $\widehat{\mathbf{W}}$ from $\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(n)}\}$.

The expected 0-1 error bound given by Theorem 3.3 consists of two terms: The first term in (3.1) relates the expected 0-1 error achieved by Algorithm 1 with a reference function class–the NTRF function class in Definition 3.2. The second term in (3.1) is a standard large-deviation error term. As long as $R = \widetilde{\mathcal{O}}(1)$, this term matches the standard $\widetilde{\mathcal{O}}(n^{-1/2})$ rate in PAC learning bounds (Shalev-Shwartz and Ben-David, 2014).

Remark 3.4. The parameter R in Theorem 3.3 is from the NTRF class and introduces a trade-off in the bound: when R is small, the corresponding NTRF class $\mathcal{F}(\mathbf{W}^{(1)}, R)$ is small, making the first term in (3.1) large, and the second term in (3.1) is small. When R is large, the corresponding function class $\mathcal{F}(\mathbf{W}^{(1)}, R)$ is large, so the first term in (3.1) is small, and the second term will be large. In particular, if we set $R = \widetilde{\mathcal{O}}(1)$, the second term in (3.1) will be $\widetilde{\mathcal{O}}(n^{-1/2})$. In this case, the "classifiability" of the underlying data distribution $\mathcal{D}$ is determined by how well its i.i.d. samples can be classified by $\mathcal{F}(\mathbf{W}^{(1)}, \widetilde{\mathcal{O}}(1))$. In other words, Theorem 3.3 suggests that if the data can be classified by a function in the NTRF function class $\mathcal{F}(\mathbf{W}^{(1)}, \widetilde{\mathcal{O}}(1))$ with a small training error, the over-parameterized ReLU network learnt by Algorithm 1 will have a small generalization error.

Remark 3.5. The expected 0-1 error bound given by Theorem 3.3 is in a very general form. It directly covers the result given by Cao and Gu (2019). In Appendix A.1, we show that under the same assumptions made in Cao and Gu (2019), to achieve ϵ expected 0-1 error, our result requires a sample complexity of order $\widetilde{\mathcal{O}}(\epsilon^{-2})$, which outperforms the result in Cao and Gu (2019) by a factor of ϵ^{-2}.

Remark 3.6. Our generalization bound can also be compared with two recent results (Yehudai and Shamir, 2019; E et al., 2019) for two-layer neural networks. When $L = 2$, the NTRF function class $\mathcal{F}(\mathbf{W}^{(1)}, \widetilde{\mathcal{O}}(1))$ can be written as

$$\big\{f_{\mathbf{W}^{(1)}}(\cdot) + \langle \nabla_{\mathbf{W}_1} f_{\mathbf{W}^{(1)}}(\cdot), \mathbf{W}_1 \rangle + \langle \nabla_{\mathbf{W}_2} f_{\mathbf{W}^{(1)}}(\cdot), \mathbf{W}_2 \rangle : \|\mathbf{W}_1\|_F, \|\mathbf{W}_2\|_F \leqslant \widetilde{\mathcal{O}}(m^{-1/2})\big\}.$$

In contrast, the reference function classes studied by Yehudai and Shamir (2019) and E et al. (2019) are contained in the following random feature class:

$$\mathcal{F} = \big\{f_{\mathbf{W}^{(1)}}(\cdot) + \langle \nabla_{\mathbf{W}_2} f_{\mathbf{W}^{(1)}}(\cdot), \mathbf{W}_2 \rangle : \|\mathbf{W}_2\|_F \leqslant \widetilde{\mathcal{O}}(m^{-1/2})\big\},$$

where $\mathbf{W}^{(1)} = (\mathbf{W}_1^{(1)}, \mathbf{W}_2^{(1)}) \in \mathbb{R}^{m \times d} \times \mathbb{R}^{1 \times m}$ are the random weights generated by the initialization schemes in Yehudai and Shamir (2019); E et al. (2019)[2]. Evidently, our NTRF function class $\mathcal{F}(\mathbf{W}^{(1)}, \widetilde{\mathcal{O}}(1))$ is richer than $\mathcal{F}$–it also contains the features corresponding to the first layer gradient of the network at random initialization, i.e., $\nabla_{\mathbf{W}_1} f_{\mathbf{W}^{(1)}}(\cdot)$. As a result, our generalization bound is sharper than those in Yehudai and Shamir (2019); E et al. (2019) in the sense that we can show that neural networks trained with SGD can compete with the best function in a larger reference function class.

3.2 Connection to Neural Tangent Kernel

Besides quantifying the classifiability of the data with the NTRF function class $\mathcal{F}(\mathbf{W}^{(1)}, \widetilde{\mathcal{O}}(1))$, an alternative way to apply Theorem 3.3 is to check how large the parameter R needs to be in order to make the first term in (3.1) small enough (e.g., smaller than $n^{-1/2}$). In this subsection, we show that this type of analysis connects Theorem 3.3 to the neural tangent kernel proposed in Jacot et al. (2018) and later studied by Yang (2019); Lee et al. (2019); Arora et al. (2019a). Specifically, we provide an expected 0-1 error bound in terms of the neural tangent kernel matrix defined over the training data. We first define the neural tangent kernel matrix for the neural network function in (2.1).

[2] Normalizing weights to the same scale is necessary for a proper comparison. See Appendix A.2 for details.

Definition 3.7 (Neural Tangent Kernel Matrix)**.** For any $i, j \in [n]$, define

$$\begin{aligned}
&\widetilde{\boldsymbol{\Theta}}^{(1)}_{i,j} = \boldsymbol{\Sigma}^{(1)}_{i,j} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle, \quad \mathbf{A}^{(l)}_{ij} = \begin{pmatrix} \boldsymbol{\Sigma}^{(l)}_{i,i} & \boldsymbol{\Sigma}^{(l)}_{i,j} \\ \boldsymbol{\Sigma}^{(l)}_{i,j} & \boldsymbol{\Sigma}^{(l)}_{j,j} \end{pmatrix}, \\
&\boldsymbol{\Sigma}^{(l+1)}_{i,j} = 2 \cdot \mathbb{E}_{(u,v)\sim N\left(\mathbf{0}, \mathbf{A}^{(l)}_{ij}\right)}[\sigma(u)\sigma(v)], \\
&\widetilde{\boldsymbol{\Theta}}^{(l+1)}_{i,j} = \widetilde{\boldsymbol{\Theta}}^{(l)}_{i,j} \cdot 2 \cdot \mathbb{E}_{(u,v)\sim N\left(\mathbf{0}, \mathbf{A}^{(l)}_{ij}\right)}[\sigma'(u)\sigma'(v)] + \boldsymbol{\Sigma}^{(l+1)}_{i,j}.
\end{aligned}$$

Then we call $\boldsymbol{\Theta}^{(L)} = [(\widetilde{\boldsymbol{\Theta}}^{(L)}_{i,j} + \boldsymbol{\Sigma}^{(L)}_{i,j})/2]_{n\times n}$ the neural tangent kernel matrix of an L-layer ReLU network on training inputs $\mathbf{x}_1, \ldots, \mathbf{x}_n$.

Definition 3.7 is the same as the original definition in Jacot et al. (2018) when restricting the kernel function on $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, except that there is an extra coefficient 2 in the second and third lines. This extra factor is due to the difference in initialization schemes–in our paper the entries of hidden layer matrices are randomly generated with variance $2/m$, while in Jacot et al. (2018) the variance of the random initialization is $1/m$. We remark that this extra factor 2 in Definition 3.7 will remove the exponential dependence on the network depth L in the kernel matrix, which is appealing. In fact, it is easy to check that under our scaling, the diagonal entries of $\boldsymbol{\Sigma}^{(L)}$ are all 1's, and the diagonal entries of $\widetilde{\boldsymbol{\Theta}}^{(L)}$ are all L's.

The following lemma is a summary of Theorem 1 and Proposition 2 in Jacot et al. (2018), which ensures that $\boldsymbol{\Theta}^{(L)}$ is the infinite-width limit of the Gram matrix $(m^{-1}\langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_j)\rangle)_{n\times n}$, and is positive-definite as long as no two training inputs are parallel.

Lemma 3.8 (Jacot et al. (2018))**.** For an L layer ReLU network with parameter set $\mathbf{W}^{(1)}$ initialized in Algorithm 1, as the network width $m \to \infty$[3], it holds that

$$m^{-1}\langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_j)\rangle \xrightarrow{\mathbb{P}} m^{-1}\mathbb{E}\big[\langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_j)\rangle\big] = \boldsymbol{\Theta}^{(L)}_{i,j},$$

where the expectation is taken over the randomness of $\mathbf{W}^{(1)}$. Moreover, as long as each pair of inputs among $\mathbf{x}_1, \ldots, \mathbf{x}_n \in S^{d-1}$ are not parallel, $\boldsymbol{\Theta}^{(L)}$ is positive-definite.

Remark 3.9. Lemmas 3.8 clearly shows the difference between our neural tangent kernel matrix $\boldsymbol{\Theta}^{(L)}$ in Definition 3.7 and the Gram matrix $\mathbf{K}^{(L)}$ defined in Definition 5.1 in Du et al. (2018a). For any $i, j \in [n]$, by Lemma 3.8 we have

$$\boldsymbol{\Theta}^{(L)}_{i,j} = m^{-1}\textstyle\sum_{l=1}^{L}\mathbb{E}\big[\langle \nabla_{\mathbf{W}_l} f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \nabla_{\mathbf{W}_l} f_{\mathbf{W}^{(1)}}(\mathbf{x}_j)\rangle\big].$$

In contrast, the corresponding entry in $\mathbf{K}^{(L)}$ is

$$\mathbf{K}^{(L)}_{i,j} = m^{-1}\mathbb{E}\big[\langle \nabla_{\mathbf{W}_{L-1}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \nabla_{\mathbf{W}_{L-1}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_j)\rangle\big].$$

It can be seen that our definition of kernel matrix takes all layers into consideration, while Du et al. (2018a) only considered the last hidden layer (i.e., second last layer). Moreover, it is clear that $\boldsymbol{\Theta}^{(L)} \succeq \mathbf{K}^{(L)}$. Since the smallest eigenvalue of the kernel matrix plays a key role in the analysis of

[3] The original result by Jacot et al. (2018) requires that the widths of different layers go to infinity sequentially. Their result was later improved by Yang (2019) such that the widths of different layers can go to infinity simultaneously.

optimization and generalization of over-parameterized neural networks (Du et al., 2018b,a; Arora et al., 2019b), our neural tangent kernel matrix can potentially lead to better bounds than the Gram matrix studied in Du et al. (2018a).

Corollary 3.10. Let $\mathbf{y} = (y_1, \ldots, y_n)^\top$ and $\lambda_0 = \lambda_{\min}(\mathbf{\Theta}^{(L)})$. For any $\delta \in (0, e^{-1}]$, there exists $\widetilde{m}^*(\delta, L, n, \lambda_0)$ that only depends on δ, L, n and λ_0 such that if $m \geqslant \widetilde{m}^*(\delta, L, n, \lambda_0)$, then with probability at least $1 - \delta$ over the randomness of $\mathbf{W}^{(1)}$, the output of Algorithm 1 with step size $\eta = \kappa \cdot \sqrt{\mathbf{y}^\top(\mathbf{\Theta}^{(L)})^{-1}\mathbf{y}}/(m\sqrt{n})$ for some small enough absolute constant κ satisfies

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leqslant \widetilde{\mathcal{O}}\Bigg[L \cdot \sqrt{\frac{\mathbf{y}^\top(\mathbf{\Theta}^{(L)})^{-1}\mathbf{y}}{n}}\Bigg] + \mathcal{O}\Bigg[\sqrt{\frac{\log(1/\delta)}{n}}\Bigg],$$

where the expectation is taken over the uniform draw of $\widehat{\mathbf{W}}$ from $\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(n)}\}$.

Remark 3.11. Corollary 3.10 gives an algorithm-dependent generalization error bound of over-parameterized L-layer neural networks trained with SGD. It is worth noting that recently Arora et al. (2019b) gives a generalization bound $\widetilde{\mathcal{O}}\big(\sqrt{\mathbf{y}^\top(\mathbf{H}^\infty)^{-1}\mathbf{y}/n}\big)$ for two-layer networks with fixed second layer weights, where $\mathbf{H}^\infty$ is defined as

$$\mathbf{H}^\infty_{i,j} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle \cdot \mathbb{E}_{\mathbf{w}\sim N(\mathbf{0},\mathbf{I})}[\sigma'(\mathbf{w}^\top\mathbf{x}_i)\sigma'(\mathbf{w}^\top\mathbf{x}_j)].$$

Our result in Corollary 3.10 can be specialized to two-layer neural networks by choosing $L = 2$, and yields a bound $\widetilde{\mathcal{O}}\big(\sqrt{\mathbf{y}^\top(\mathbf{\Theta}^{(2)})^{-1}\mathbf{y}/n}\big)$, where

$$\mathbf{\Theta}^{(2)}_{i,j} = \mathbf{H}^\infty_{i,j} + 2 \cdot \mathbb{E}_{\mathbf{w}\sim N(\mathbf{0},\mathbf{I})}[\sigma(\mathbf{w}^\top\mathbf{x}_i)\sigma(\mathbf{w}^\top\mathbf{x}_j)].$$

Here the extra term $2 \cdot \mathbb{E}_{\mathbf{w}\sim N(\mathbf{0},\mathbf{I})}[\sigma(\mathbf{w}^\top\mathbf{x}_i)\sigma(\mathbf{w}^\top\mathbf{x}_j)]$ corresponds to the training of the second layer–it is the limit of $\frac{1}{m}\langle \nabla_{\mathbf{W}_2} f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \nabla_{\mathbf{W}_2} f_{\mathbf{W}^{(1)}}(\mathbf{x}_j)\rangle$. Since we have $\mathbf{\Theta}^{(2)} \succeq \mathbf{H}^\infty$, our bound is sharper than theirs. This comparison also shows that, our result generalizes the result in Arora et al. (2019b) from two-layer, fixed second layer networks to deep networks with all parameters being trained.

Remark 3.12. Corollary 3.10 is based on the asymptotic convergence result in Lemma 3.8, which does not show how wide the network need to be in order to make the Gram matrix close enough to the NTK matrix. Very recently, Arora et al. (2019a) provided a non-asymptotic convergence result for the Gram matrix, and showed the equivalence between an infinitely wide network trained by gradient flow and a kernel regression predictor using neural tangent kernel, which suggests that the generalization of deep neural networks trained by gradient flow can potentially be measured by the corresponding NTK. Utilizing this non-asymptotic convergence result, one can potentially specify the detailed dependency of $\widetilde{m}^*(\delta, L, n, \lambda_0)$ on δ, L, n and λ_0 in Corollary 3.10.

4 Proof of Main Theory

In this section we provide the proof of Theorem 3.3 and Corollary 3.10, and explain the intuition behind the proof. For notational simplicity, for $i \in [n]$ we denote $L_i(\mathbf{W}) = L_{(\mathbf{x}_i, y_i)}(\mathbf{W})$.

4.1 Proof of Theorem 3.3

Before giving the proof of Theorem 3.3, we first introduce several lemmas. The following lemma states that near initialization, the neural network function is almost linear in terms of its weights.

Lemma 4.1. There exists an absolute constant κ such that, with probability at least $1 - \mathcal{O}(nL^2) \cdot \exp[-\Omega(m\omega^{2/3}L)]$ over the randomness of $\mathbf{W}^{(1)}$, for all $i \in [n]$ and $\mathbf{W}, \mathbf{W}' \in \mathcal{B}(\mathbf{W}^{(1)}, \omega)$ with $\omega \leqslant \kappa L^{-6}[\log(m)]^{-3/2}$, it holds uniformly that

$$|f_{\mathbf{W}'}(\mathbf{x}_i) - f_{\mathbf{W}}(\mathbf{x}_i) - \langle \nabla f_{\mathbf{W}}(\mathbf{x}_i), \mathbf{W}' - \mathbf{W}\rangle| \leqslant \mathcal{O}\Big(\omega^{1/3}L^2\sqrt{m\log(m)}\Big) \cdot \textstyle\sum_{l=1}^{L-1}\|\mathbf{W}'_l - \mathbf{W}_l\|_2.$$

Since the cross-entropy loss $\ell(\cdot)$ is convex, given Lemma 4.1, we can show in the following lemma that near initialization, $L_i(\mathbf{W})$ is also almost a convex function of $\mathbf{W}$ for any $i \in [n]$.

Lemma 4.2. There exists an absolute constant κ such that, with probability at least $1 - \mathcal{O}(nL^2) \cdot \exp[-\Omega(m\omega^{2/3}L)]$ over the randomness of $\mathbf{W}^{(1)}$, for any $\epsilon > 0$, $i \in [n]$ and $\mathbf{W}, \mathbf{W}' \in \mathcal{B}(\mathbf{W}^{(1)}, \omega)$ with $\omega \leqslant \kappa L^{-6} m^{-3/8}[\log(m)]^{-3/2}\epsilon^{3/4}$, it holds uniformly that

$$L_i(\mathbf{W}') \geqslant L_i(\mathbf{W}) + \langle \nabla_{\mathbf{W}} L_i(\mathbf{W}), \mathbf{W}' - \mathbf{W}\rangle - \epsilon.$$

The locally almost convex property of the loss function given by Lemma 4.2 implies that the dynamics of Algorithm 1 is similar to the dynamics of convex optimization. We can therefore derive a bound of the cumulative loss. The result is given in the following lemma.

Lemma 4.3. For any $\epsilon, \delta, R > 0$, there exists

$$m^*(\epsilon, \delta, R, L) = \widetilde{\mathcal{O}}\big(\mathrm{poly}(R, L)\big) \cdot \epsilon^{-14} \cdot \log(1/\delta)$$

such that if $m \geqslant m^*(\epsilon, \delta, R, L)$, then with probability at least $1 - \delta$ over the randomness of $\mathbf{W}^{(1)}$, for any $\mathbf{W}^* \in \mathcal{B}(\mathbf{W}^{(1)}, Rm^{-1/2})$, Algorithm 1 with $\eta = \nu\epsilon/(Lm)$, $n = L^2R^2/(2\nu\epsilon^2)$ for some small enough absolute constant ν has the following cumulative loss bound:

$$\textstyle\sum_{i=1}^n L_i(\mathbf{W}^{(i)}) \leqslant \sum_{i=1}^n L_i(\mathbf{W}^*) + 3n\epsilon.$$

We now finalize the proof by applying an online-to-batch conversion argument (Cesa-Bianchi et al., 2004), and use Lemma 4.1 to relate the neural network function with a function in the NTRF function class.

Proof of Theorem 3.3. For $i \in [n]$, let $L_i^{0-1}(\mathbf{W}^{(i)}) = \mathbb{1}\big\{y_i \cdot f_{\mathbf{W}^{(i)}}(\mathbf{x}_i) < 0\big\}$. Since cross-entropy loss satisfies $\mathbb{1}\{z \leqslant 0\} \leqslant 4\ell(z)$, we have $L_i^{0-1}(\mathbf{W}^{(i)}) \leqslant 4L_i(\mathbf{W}^{(i)})$. Therefore, setting $\epsilon = LR/\sqrt{2\nu n}$ in Lemma 4.3 gives that, if η is set as $\sqrt{\nu/2}R/(m\sqrt{n})$, then with probability at least $1 - \delta$,

$$\frac{1}{n}\sum_{i=1}^n L_i^{0-1}(\mathbf{W}^{(i)}) \leqslant \frac{4}{n}\sum_{i=1}^n L_i(\mathbf{W}^*) + \frac{12}{\sqrt{2\nu}} \cdot \frac{LR}{\sqrt{n}}. \tag{4.1}$$

Note that for any $i \in [n]$, $\mathbf{W}^{(i)}$ only depends on $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{i-1}, y_{i-1})$ and is independent of $(\mathbf{x}_i, y_i)$. Therefore by Proposition 1 in Cesa-Bianchi et al. (2004), with probability at least $1 - \delta$

we have

$$\frac{1}{n}\sum_{i=1}^{n} L_{\mathcal{D}}^{0-1}(\mathbf{W}^{(i)}) \leqslant \frac{1}{n}\sum_{i=1}^{n} L_{i}^{0-1}(\mathbf{W}^{(i)}) + \sqrt{\frac{2\log(1/\delta)}{n}}. \tag{4.2}$$

By definition, we have $\frac{1}{n}\sum_{i=1}^{n} L_{\mathcal{D}}^{0-1}(\mathbf{W}^{(i)}) = \mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big]$. Therefore combining (4.1) and (4.2) and applying union bound, we obtain that with probability at least $1-2\delta$,

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leqslant \frac{4}{n}\sum_{i=1}^{n} L_i(\mathbf{W}^*) + \frac{12}{\sqrt{2\nu}}\cdot\frac{LR}{\sqrt{n}} + \sqrt{\frac{2\log(1/\delta)}{n}} \tag{4.3}$$

for all $\mathbf{W}^* \in \mathcal{B}(\mathbf{W}^{(1)}, Rm^{-1/2})$. We now compare the neural network function $f_{\mathbf{W}^*}(\mathbf{x}_i)$ with the function $F_{\mathbf{W}^{(1)},\mathbf{W}^*}(\mathbf{x}_i) := f_{\mathbf{W}^{(1)}}(\mathbf{x}_i) + \langle \nabla f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \mathbf{W}^* - \mathbf{W}^{(1)}\rangle \in \mathcal{F}(\mathbf{W}^{(1)}, R)$. We have

$$\begin{aligned}
L_i(\mathbf{W}^*) &\leqslant \ell[y_i \cdot F_{\mathbf{W}^{(1)},\mathbf{W}^*}(\mathbf{x}_i)] + \mathcal{O}\Big((Rm^{-1/2})^{1/3}L^2\sqrt{m\log(m)}\Big)\cdot \textstyle\sum_{l=1}^{L-1}\big\|\mathbf{W}_l^* - \mathbf{W}_l^{(1)}\big\|_2\\
&\leqslant \ell[y_i \cdot F_{\mathbf{W}^{(1)},\mathbf{W}^*}(\mathbf{x}_i)] + \mathcal{O}\Big(L^3\sqrt{m\log(m)}\Big)\cdot R^{4/3}\cdot m^{-2/3}\\
&\leqslant \ell[y_i \cdot F_{\mathbf{W}^{(1)},\mathbf{W}^*}(\mathbf{x}_i)] + LRn^{-1/2},
\end{aligned}$$

where the first inequality is by the 1-Lipschitz continuity of $\ell(\cdot)$ and Lemma 4.1, the second inequality is by $\mathbf{W}^* \in \mathcal{B}(\mathbf{W}^{(1)}, Rm^{-1/2})$, and last inequality holds as long as $m \geqslant C_1R^2L^{12}[\log(m)]^3n^3$ for some large enough absolute constant C_1. Plugging the inequality above into (4.3) gives

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leqslant \frac{4}{n}\sum_{i=1}^{n} \ell[y_i \cdot F_{\mathbf{W}^{(1)},\mathbf{W}^*}(\mathbf{x}_i)] + \bigg(1 + \frac{12}{\sqrt{2\nu}}\bigg)\cdot\frac{LR}{\sqrt{n}} + \sqrt{\frac{2\log(1/\delta)}{n}}.$$

Taking infimum over $\mathbf{W}^* \in \mathcal{B}(\mathbf{W}^{(1)}, Rm^{-1/2})$ and rescaling δ finishes the proof. □

4.2 Proof of Corollary 3.10

In this subsection we prove Corollary 3.10. The following lemma shows that at initialization, with high probability, the neural network function value at all the training inputs are of order $\widetilde{\mathcal{O}}(1)$.

Lemma 4.4. For any $\delta > 0$, if $m \geqslant KL\log(nL/\delta)$ for a large enough absolute constant K, then with probability at least $1-\delta$, $|f_{\mathbf{W}^{(1)}}(\boldsymbol{x}_i)| \leqslant \mathcal{O}(\sqrt{\log(n/\delta)})$ for all $i \in [n]$.

We now present the proof of Corollary 3.10. The idea is to construct suitable target values $\widehat{y}_1, \ldots, \widehat{y}_n$, and then bound the norm of the solution of the linear equations $\widehat{y}_i = \langle \nabla f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \mathbf{W}\rangle$, $i \in [n]$.

Proof of Corollary 3.10. Set $B = \log\{1/[\exp(n^{-1/2}) - 1]\} = \mathcal{O}(\log(n))$, then for cross-entropy loss we have $\ell(z) \leqslant n^{-1/2}$ for $z \geqslant B$. Moreover, let $B' = \max_{i\in[n]}|f_{\mathbf{W}^{(1)}}(\boldsymbol{x}_i)|$. Then by Lemma 4.4, with probability at least $1-\delta$, $B' \leqslant \mathcal{O}(\sqrt{\log(n/\delta)})$ for all $i \in [n]$. Let $\overline{B} = B + B'$ and $\widehat{\mathbf{y}} = \overline{B}\cdot\mathbf{y}$, then it holds that for any $i \in [n]$,

$$y_i \cdot [\widehat{y}_i + f_{\mathbf{W}^{(1)}}(\boldsymbol{x}_i)] = y_i \cdot \widehat{y}_i + y_i \cdot f_{\mathbf{W}^{(1)}}(\boldsymbol{x}_i) \geqslant B + B' - B' \geqslant B,$$

and therefore

$$\ell\{y_i \cdot [\widehat{y}_i + f_{\mathbf{W}^{(1)}}(\boldsymbol{x}_i)]\} \leqslant n^{-1/2},\ \ i \in [n]. \tag{4.4}$$

Denote $\mathbf{F} = m^{-1/2} \cdot (\mathrm{vec}[\nabla f_{\mathbf{W}^{(1)}}(\mathbf{x}_1)], \ldots, \mathrm{vec}[\nabla f_{\mathbf{W}^{(1)}}(\mathbf{x}_n)]) \in \mathbb{R}^{[md+m+m^2(L-2)]\times n}$. Note that entries of $\boldsymbol{\Theta}^{(L)}$ are all bounded by L. Therefore, the largest eigenvalue of $\boldsymbol{\Theta}^{(L)}$ is at most nL, and we have $\mathbf{y}^\top(\boldsymbol{\Theta}^{(L)})^{-1}\mathbf{y} \geqslant n^{-1}L^{-1}\|\mathbf{y}\|_2^2 = L^{-1}$. By Lemma 3.8 and standard matrix perturbation bound, there exists $m^*(\delta, L, n, \lambda_0)$ such that, if $m \geqslant m^*(\delta, L, n, \lambda_0)$, then with probability at least $1-\delta$, $\mathbf{F}^\top\mathbf{F}$ is strictly positive-definite and

$$\|(\mathbf{F}^\top\mathbf{F})^{-1} - (\boldsymbol{\Theta}^{(L)})^{-1}\|_2 \leqslant \mathbf{y}^\top(\boldsymbol{\Theta}^{(L)})^{-1}\mathbf{y}/n. \tag{4.5}$$

Let $\mathbf{F} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{Q}^\top$ be the singular value decomposition of $\mathbf{F}$, where $\mathbf{P} \in \mathbb{R}^{m\times n}, \mathbf{Q} \in \mathbb{R}^{n\times n}$ have orthogonal columns, and $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ is a diagonal matrix. Let $\mathbf{w}_{\mathrm{vec}} = \mathbf{P}\boldsymbol{\Lambda}^{-1}\mathbf{Q}^\top\widehat{\mathbf{y}}$, then we have

$$\mathbf{F}^\top\mathbf{w}_{\mathrm{vec}} = (\mathbf{Q}\boldsymbol{\Lambda}\mathbf{P}^\top)(\mathbf{P}\boldsymbol{\Lambda}^{-1}\mathbf{Q}^\top\widehat{\mathbf{y}}) = \widehat{\mathbf{y}}. \tag{4.6}$$

Moreover, by direct calculation we have

$$\|\mathbf{w}_{\mathrm{vec}}\|_2^2 = \|\mathbf{P}\boldsymbol{\Lambda}^{-1}\mathbf{Q}^\top\widehat{\mathbf{y}}\|_2^2 = \|\boldsymbol{\Lambda}^{-1}\mathbf{Q}^\top\widehat{\mathbf{y}}\|_2^2 = \widehat{\mathbf{y}}^\top\mathbf{Q}\boldsymbol{\Lambda}^{-2}\mathbf{Q}^\top\widehat{\mathbf{y}} = \widehat{\mathbf{y}}^\top(\mathbf{F}^\top\mathbf{F})^{-1}\widehat{\mathbf{y}}.$$

Therefore by (4.5) and the fact that $\|\widehat{\mathbf{y}}\|_2^2 = \overline{B}^2 n$, we have

$$\begin{aligned}\|\mathbf{w}_{\mathrm{vec}}\|_2^2 &= \widehat{\mathbf{y}}^\top[(\mathbf{F}^\top\mathbf{F})^{-1} - (\boldsymbol{\Theta}^{(L)})^{-1}]\widehat{\mathbf{y}} + \widehat{\mathbf{y}}^\top(\boldsymbol{\Theta}^{(L)})^{-1}\widehat{\mathbf{y}}\\ &\leqslant \overline{B}^2 \cdot n \cdot \|(\mathbf{F}^\top\mathbf{F})^{-1} - (\boldsymbol{\Theta}^{(L)})^{-1}\|_2 + \overline{B}^2 \cdot \mathbf{y}^\top(\boldsymbol{\Theta}^{(L)})^{-1}\mathbf{y}\\ &\leqslant 2\overline{B}^2 \cdot \mathbf{y}^\top(\boldsymbol{\Theta}^{(L)})^{-1}\mathbf{y}.\end{aligned}$$

Let $\mathbf{W} \in \mathcal{W}$ be the parameter collection reshaped from $m^{-1/2}\mathbf{w}_{\mathrm{vec}}$. Then clearly

$$\|\mathbf{W}_l\|_F \leqslant m^{-1/2}\|\mathbf{w}_{\mathrm{vec}}\|_2 \leqslant \widetilde{\mathcal{O}}\Big(\sqrt{\mathbf{y}^\top(\boldsymbol{\Theta}^{(L)})^{-1}\mathbf{y}} \cdot m^{-1/2}\Big),$$

and therefore $\mathbf{W} \in \mathcal{B}\big(\mathbf{0}, \mathcal{O}\big(\sqrt{\mathbf{y}^\top(\boldsymbol{\Theta}^{(L)})^{-1}\mathbf{y}}\cdot m^{-1/2}\big)\big)$. Moreover, by (4.6), we have $\widehat{y}_i = \langle\nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\mathbf{x}_i), \mathbf{W}\rangle$. Plugging this into (4.4) then gives

$$\ell\big\{y_i \cdot \big[f_{\mathbf{W}^{(1)}}(\mathbf{x}_i) + \langle\nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\boldsymbol{x}_i), \mathbf{W}\rangle\big]\big\} \leqslant n^{-1/2}.$$

Since $f^*(\cdot) = f_{\mathbf{W}^{(1)}}(\cdot) + \langle\nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\cdot), \mathbf{W}\rangle \in \mathcal{F}\big(\mathbf{W}^{(1)}, \widetilde{\mathcal{O}}\big(\sqrt{\mathbf{y}^\top(\boldsymbol{\Theta}^{(L)})^{-1}\mathbf{y}}\big)\big)$, applying Theorem 3.3 completes the proof. □

5 Conclusions

In this paper we provide an expected 0-1 error bound for wide and deep ReLU networks trained with SGD. This generalization error bound is measured by the NTRF function class. The connection to the neural tangent kernel function studied in Jacot et al. (2018) is also discussed. Our result covers a series of recent generalization bounds for wide enough neural networks, and provides better bounds.

Acknowledgement

We would like to thank Peter Bartlett for a valuable discussion, and Simon S. Du for pointing out the reference Arora et al. (2019a).

A Comparison with Recent Results

In this section we compare our result in Theorem 3.3 with recent generalization error bounds for over-paramerized neural networks by Cao and Gu (2019); Yehudai and Shamir (2019); E et al. (2019), and backup our discussions in Remark 3.5 and Remark 3.6.

A.1 Comparison with Cao and Gu (2019)

In this section we provide direct comparison between our result in Theorem 3.3 and Theorem 4.4 in Cao and Gu (2019). To concretely compare these two results, we apply our result to the setting studied in Cao and Gu (2019), which is based on the following assumption.

Assumption A.1. There exist a constant $\gamma > 0$ and

$$f(\cdot) \in \left\{ f(\mathbf{x}) = \int_{\mathbb{R}^d} c(\mathbf{u})\sigma(\mathbf{u}^\top \mathbf{x})p(\mathbf{u})\mathrm{d}\mathbf{u} : \|c(\cdot)\|_\infty \leqslant 1 \right\},$$

where $p(\mathbf{u})$ the density of standard Gaussian vectors, such that $y \cdot f(\mathbf{x}) \geqslant \gamma$ for all $(\mathbf{x}, y) \in \mathrm{supp}(\mathcal{D})$.

Under Assumption 3.1 and Assumption A.1, in order to train the network to achieve ϵ expected 0-1 loss, Cao and Gu (2019) gave a sample complexity of order $\widetilde{\mathcal{O}}(\mathrm{poly}(2^L, \gamma^{-1}) \cdot \epsilon^{-4})$. In comparison, our result in Theorem 3.3 leads to the following corollary.

Corollary A.2. Under Assumption 3.1 and Assumption A.1, for any $\delta \in (0, e^{-1}]$, there exists

$$m^*(\delta, \gamma, L, n) = \widetilde{\mathcal{O}}\big(\mathrm{poly}(2^L, \gamma^{-1})\big) \cdot n^7 \cdot \log(1/\delta)$$

such that if $m \geqslant m^*(\delta, R, L, n)$, then with probability at least $1 - \delta$ over the randomness of $\mathbf{W}^{(1)}$, the parameters given by Algorithm 1 with $\eta = \kappa \cdot R/(m\sqrt{n})$ for some small enough absolute constant κ satisfies

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leqslant \widetilde{\mathcal{O}}\left(\frac{2^L \cdot \gamma^{-1}}{\sqrt{n}}\right),$$

where the expectation is taken over the draws of training examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ as well as the uniform draw of $\widehat{\mathbf{W}}$ from $\{\mathbf{W}^{(1)}, \ldots, \mathbf{W}^{(n)}\}$.

By setting the expected 0-1 loss bound to ϵ, we obtain a sample complexity of order $\widetilde{\mathcal{O}}(4^L \cdot \gamma^{-2}\epsilon^{-2})$, which is better than the sample complexity given in Cao and Gu (2019) by a factor of ϵ^{-2}.

A.2 Comparison with Yehudai and Shamir (2019); E et al. (2019)

Here we give a detailed explanation to Remark 3.6, where we compare our result with Yehudai and Shamir (2019); E et al. (2019). The reference function classes studied in these two papers share the same general form:

$$\big\{f(x) = \mathbf{W}_2\sigma(\mathbf{W}_1^{(1)}\mathbf{x}) : \|\mathbf{W}_2\|_F \leqslant Cm^{-1/2}\big\},$$

where C is a constant, and $\mathbf{W}_1^{(1)} \in \mathbb{R}^{m\times d}$ is the first layer parameter matrix whose rows are sampled from certain distribution π associated to the initialization scheme. Specifically, Yehudai and Shamir (2019) studied the case where π is the uniform distribution over the d-dimensional cube $[-d^{-1/2}, d^{-1/2}]^d$, while E et al. (2019) studied the uniform distribution over the sphere S^{d-1}. By standard concentration inequality, we can see that in both papers, with high probability, the distribution π gives $\mathbf{W}_1^{(1)}$ with $\|\mathbf{W}_1^{(1)}\|_2 \approx \mathcal{O}(m^{1/2})$. In terms of second layer initialization $\mathbf{W}_2^{(1)}$, the generalization results in both papers require that $\|\mathbf{W}_2^{(1)}\|_2 \leqslant \mathcal{O}(m^{-1/2})$. With such a scaling, we can apply the following lemma.

Lemma A.3. Suppose that $\mathbf{W}^{(1)} = (\mathbf{W}_1^{(1)}, \mathbf{W}_2^{(1)}) \in \mathbb{R}^{m\times d}\times\mathbb{R}^{1\times m}$ be weights satisfying $\|\mathbf{W}_2^{(1)}\|_F \leqslant Km^{-1/2}$ for some $K = \widetilde{\mathcal{O}}(1)$, then

$$\big\{f(x) = \mathbf{W}_2\sigma(\mathbf{W}_1^{(1)}\mathbf{x}) : \|\mathbf{W}_2\|_F \leqslant Cm^{-1/2}\big\} \subseteq \mathcal{F},$$

where

$$\mathcal{F} = \big\{\mathbf{W}_2^{(1)}\sigma(\mathbf{W}_1^{(1)}\mathbf{x}) + \mathbf{W}_2\sigma(\mathbf{W}_1^{(1)}\mathbf{x}) : \|\mathbf{W}_2\|_F \leqslant (C+K)\cdot m^{-1/2}\big\},$$

and $\sigma(\cdot)$ is the activation function of interest.

We compare our result with the bounds given by Yehudai and Shamir (2019); E et al. (2019) by comparing the reference function classes we use. Apparently, a larger reference function class in general gives a better generalization error bound. Such a comparison requires us to adjust the scaling of initialized parameters. Based on our previous discussion, it is easy to see that the initialized second layer weights in our work and Yehudai and Shamir (2019); E et al. (2019) are all of the same scaling. However, the $\|\cdot\|_2$ of first layer weight matrix in Yehudai and Shamir (2019); E et al. (2019) is larger than ours by a factor of $\sqrt{m}$. Adjusting this scaling difference will give an extra factor $\sqrt{m}$, which matches the $\sqrt{m}$ factor in the definition of our neural network function. Note that even after adjusting the scaling of parameters, these random feature function classes are not directly comparable, since the activation functions and the distributions of random weights are different. However, an informal comparison can already clearly show the advantage of our result. Moreover, we remark that at least for two-layer networks, our analysis can be easily generalized to other activation functions and initialization methods, and the resulting NTRF class should be strictly larger than the random feature function classes used in Yehudai and Shamir (2019); E et al. (2019). This justifies our discussion in Remark 3.6.

B Proofs of Technical Lemmas in Section 4

In this section we provide the proofs of the technical lemmas in Section 4. We first introduce some extra notations. Following Allen-Zhu et al. (2018b), for a parameter collection $\mathbf{W}$ and $i \in [n]$, we denote

$$\mathbf{h}_{i,0} = \mathbf{x}_i,\ \ \mathbf{h}_{i,l} = \sigma(\mathbf{W}_l\mathbf{h}_{i,l-1}), l \in [L-1]$$

as the hidden layer outputs of the network. We also define binary diagonal matrices

$$\mathbf{D}_{i,l} = \mathrm{diag}\big(\mathbb{1}\{(\mathbf{W}_l\mathbf{h}_{i,l})_1 > 0\}, \ldots, \mathbb{1}\{(\mathbf{W}_l\mathbf{h}_{i,l})_m > 0\}\big), l \in [L-1].$$

For $i \in [n]$ and $l \in [L-1]$, we use $\mathbf{h}'_{i,l}$, $\mathbf{D}'_{i,l}$ and $\mathbf{h}^{(1)}_{i,l}$, $\mathbf{D}^{(1)}_{i,l}$ to denote the hidden layer outputs and binary diagonal matrices with parameter collections $\mathbf{W}'$ and $\mathbf{W}^{(1)}$ respectively. We also implement the following matrix product notation which is also used in Zou et al. (2018); Cao and Gu (2019):

$$\prod_{r=l_1}^{l_2} \mathbf{A}_r := \begin{cases} \mathbf{A}_{l_2}\mathbf{A}_{l_2-1}\cdots\mathbf{A}_{l_1} & \text{if } l_1 \leqslant l_2 \\ \mathbf{I} & \text{otherwise.} \end{cases}$$

With this notation, we have the following matrix product representation of the neural network gradients:

$$\nabla_{\mathbf{W}_l} f_{\mathbf{W}}(\mathbf{x}_i) = \begin{cases} \sqrt{m}\cdot\big[\mathbf{h}_{i,l-1}\mathbf{W}_L\big(\prod_{r=l+1}^{L-1}\mathbf{D}_{i,r}\mathbf{W}_r\big)\mathbf{D}_{i,l}\big]^\top, & l \in [L-1], \\ \sqrt{m}\cdot\mathbf{h}_{i,L-1}^\top, & l = L. \end{cases}$$

B.1 Proof of Lemma 4.1

The following two lemmas are proved based on several results given by Allen-Zhu et al. (2018b). Note that in their paper, both the first and the last layers of the network are fixed, which is slightly different from our setting. We remark that this difference does not affect the result.

Lemma B.1. If $\omega \leqslant \mathcal{O}(L^{-9/2}[\log(m)]^{-3})$, then with probability at least $1-\mathcal{O}(nL)\cdot\exp[-\Omega(m\omega^{2/3}L)]$, $1/2 \leqslant \|\mathbf{h}_{i,l}\|_2 \leqslant 3/2$ for all $\mathbf{W} \in \mathcal{B}(\mathbf{W}^{(1)}, \omega)$, $i \in [n]$ and $l \in [L-1]$.

Lemma B.2. If $\omega \leqslant \mathcal{O}(L^{-6}[\log(m)]^{-3})$, then with probability at least $1-\mathcal{O}(nL^2)\cdot\exp[-\Omega(m\omega^{2/3}L)]$, uniformly over:

- any $i \in [n]$, $1 \leqslant l_1 < l_2 \leqslant L-1$
- any diagonal matrices $\mathbf{D}''_{i,1}, \ldots, \mathbf{D}''_{i,L-1} \in [-1,1]^{m\times m}$ with at most $\mathcal{O}(m\omega^{2/3}L)$ non-zero entries,

the following results hold:

(i) For all $\mathbf{W} \in \mathcal{B}(\mathbf{W}^{(1)}, \omega)$, $\|\prod_{r=l_1}^{l_2}(\mathbf{D}_{i,r} + \mathbf{D}''_{i,r})\mathbf{W}_r\|_2 \leqslant \mathcal{O}(\sqrt{L})$.

(ii) For all $\mathbf{W} \in \mathcal{B}(\mathbf{W}^{(1)}, \omega)$, $\|\mathbf{W}_L\prod_{r=l_1}^{L-1}(\mathbf{D}_{i,r} + \mathbf{D}''_{i,r})\mathbf{W}_r\|_2 \leqslant \mathcal{O}(1)$.

(iii) For all $\mathbf{W}, \mathbf{W}' \in \mathcal{B}(\mathbf{W}^{(1)}, \omega)$,

$$\bigg\|\mathbf{W}'_L\prod_{r=l_1}^{L-1}(\mathbf{D}'_{i,r} + \mathbf{D}''_{i,r})\mathbf{W}'_r - \mathbf{W}_L\prod_{r=l_1}^{L-1}\mathbf{D}_{i,r}\mathbf{W}_r\bigg\|_2 \leqslant \mathcal{O}\Big(\omega^{1/3}L^2\sqrt{\log(m)}\Big).$$

We are now ready to prove Lemma 4.1.

Proof of Lemma 4.1. Since $f_{\mathbf{W}'}(\mathbf{x}_i) = \sqrt{m} \cdot \mathbf{W}'_L \mathbf{h}'_{i,L-1}$, $f_{\mathbf{W}}(\mathbf{x}_i) = \sqrt{m} \cdot \mathbf{W}_L \mathbf{h}_{i,L-1}$, by direct calculation, we have

$$
\begin{aligned}
f_{\mathbf{W}'}(\mathbf{x}_i) - F_{\mathbf{W},\mathbf{W}'}(\mathbf{x}_i) = -\sqrt{m} \cdot \sum_{l=1}^{L-1} \mathbf{W}_L \Bigg(\prod_{r=l+1}^{L-1} \mathbf{D}_{i,r} \mathbf{W}_r \Bigg) \mathbf{D}_{i,l} (\mathbf{W}'_l - \mathbf{W}_l) \mathbf{h}_{i,l-1} \\
+ \sqrt{m} \cdot \mathbf{W}'_L (\mathbf{h}'_{i,L-1} - \mathbf{h}_{i,L-1}).
\end{aligned}
$$

By Claim 8.2 in Allen-Zhu et al. (2018b) , there exist diagonal matrices $\mathbf{D}''_{i,l} \in \mathbb{R}^{m \times m}$ with entries in $[-1, 1]$ such that $\|\mathbf{D}''_{i,l}\|_0 \leqslant \mathcal{O}(m\omega^{2/3}L)$ and

$$
\mathbf{h}_{i,L-1} - \mathbf{h}'_{i,L-1} = \sum_{l=1}^{L-1} \Bigg[\prod_{r=l+1}^{L-1} (\mathbf{D}'_{i,r} + \mathbf{D}''_{i,r}) \mathbf{W}'_r \Bigg] (\mathbf{D}'_{i,l} + \mathbf{D}''_{i,l}) (\mathbf{W}_l - \mathbf{W}'_l) \mathbf{h}_{i,l-1}
$$

for all $i \in [n]$. Therefore

$$
\begin{aligned}
f_{\mathbf{W}'}(\mathbf{x}_i) - F_{\mathbf{W},\mathbf{W}'}(\mathbf{x}_i) = \sqrt{m} \cdot \sum_{l=1}^{L-1} \mathbf{W}'_L \Bigg[\prod_{r=l+1}^{L-1} (\mathbf{D}'_{i,r} + \mathbf{D}''_{i,r}) \mathbf{W}'_r \Bigg] (\mathbf{D}'_{i,l} + \mathbf{D}''_{i,l}) (\mathbf{W}_l - \mathbf{W}'_l) \mathbf{h}_{i,l-1} \\
- \sqrt{m} \cdot \sum_{l=1}^{L-1} \mathbf{W}_L \Bigg(\prod_{r=l+1}^{L-1} \mathbf{D}_{i,r} \mathbf{W}_r \Bigg) \mathbf{D}_{i,l} (\mathbf{W}'_l - \mathbf{W}_l) \mathbf{h}_{i,l-1}.
\end{aligned}
$$

By (iii) in Lemma B.2, with probability at least $1 - \mathcal{O}(nL^2) \cdot \exp[-\Omega(m\omega^{2/3}L)]$, we have

$$
\begin{aligned}
|f_{\mathbf{W}'}(\mathbf{x}_i) - F_{\mathbf{W},\mathbf{W}'}(\mathbf{x}_i)| &\leqslant \mathcal{O}\Big(\omega^{1/3} L^2 \sqrt{m \log(m)}\Big) \cdot \sum_{l=1}^{L-1} \|\mathbf{h}_{i.l-1}\|_2 \cdot \|\mathbf{W}'_l - \mathbf{W}_l\|_2 \\
&\leqslant \mathcal{O}\Big(\omega^{1/3} L^2 \sqrt{m \log(m)}\Big) \cdot \sum_{l=1}^{L-1} \|\mathbf{W}'_l - \mathbf{W}_l\|_2,
\end{aligned}
$$

where the last inequality follows by Lemma B.1. This inequality finishes the proof. □

B.2 Proof of Lemma 4.2

Intuitively, Lemma 4.2 follows by the fact that the composition of a convex function and an almost linear function is almost convex. The detailed proof is as follows.

Proof of Lemma 4.2. By the convexity of $\ell(z)$, we have

$$
L_i(\mathbf{W}') - L_i(\mathbf{W}) = \ell[y_i f_{\mathbf{W}'}(\mathbf{x}_i)] - \ell[y_i f_{\mathbf{W}}(\mathbf{x}_i)] \geqslant \ell'[y_i f_{\mathbf{W}}(\mathbf{x}_i)] \cdot y_i \cdot [f_{\mathbf{W}'}(\mathbf{x}_i) - f_{\mathbf{W}}(\mathbf{x}_i)].
$$

Since $|\ell'(z)| \leqslant 1$, applying Lemma 4.1 gives

$$
\begin{aligned}
L_i(\mathbf{W}') - L_i(\mathbf{W}) &\geqslant \sum_{l=1}^{L} \langle \nabla_{\mathbf{W}_l} L_i(\mathbf{W}), \mathbf{W}'_l - \mathbf{W}_l \rangle - \mathcal{O}\Big(\omega^{1/3} L^2 \sqrt{m \log(m)}\Big) \sum_{l=1}^{L-1} \|\mathbf{W}'_l - \mathbf{W}_l\|_2 \\
&\geqslant \sum_{l=1}^{L} \langle \nabla_{\mathbf{W}_l} L_i(\mathbf{W}), \mathbf{W}'_l - \mathbf{W}_l \rangle - \epsilon,
\end{aligned}
$$

where the last inequality again follows by $\omega \leqslant \mathcal{O}\big(L^{-9/4}m^{-3/8}[\log(m)]^{-3/8}\epsilon^{3/4}\big)$. □

B.3 Proof of Lemma 4.3

To prove Lemma 4.3, we first introduce the following lemma which provides an upper bound for the gradient of the neural network function near initialization.

Lemma B.3. There exists an absolute constant κ such that, with probability at least $1-\mathcal{O}(nL^2)\cdot\exp[-\Omega(m\omega^{2/3}L)]$, for all $i\in[n]$, $l\in[L]$ and $\mathbf{W}\in\mathcal{B}(\mathbf{W}^{(1)},\omega)$ with $\omega\leqslant\kappa L^{-6}[\log(m)]^{-3}$, it holds uniformly that

$$\|\nabla_{\mathbf{W}_l}f_{\mathbf{W}}(\mathbf{x}_i)\|_F, \|\nabla_{\mathbf{W}_l}L_i(\mathbf{W})\|_F \leqslant \mathcal{O}(\sqrt{m}).$$

We now provide the final proof of Lemma 4.3.

Proof of Lemma 4.3. Let $\omega = C_1L^{-6}m^{-3/8}[\log(m)]^{-3}\epsilon^{3/4}$, where C_1 is a small enough absolute constant such that the conditions on ω given in Lemmas 4.2 and B.3 hold. It is easy to see that as long as $m\geqslant C_1^{-8}R^8L^{48}[\log(m)]^{12}\epsilon^{-6}$, we have $\mathbf{W}^*\in\mathcal{B}(\mathbf{W}^{(1)},\omega)$. We now show that under our parameter choice, $\mathbf{W}^{(1)},\ldots,\mathbf{W}^{(n)}$ are inside $\mathcal{B}(\mathbf{W}^{(1)},\omega)$ as well.

This result follows by simple induction. Clearly we have $\mathbf{W}^{(1)}\in\mathcal{B}(\mathbf{W}^{(1)},\omega)$. Suppose that $\mathbf{W}^{(i)}\in\mathcal{B}(\mathbf{W}^{(1)},\omega)$. Then by Lemma B.3, for $l\in[L]$ we have $\|\nabla_{\mathbf{W}_l}L_i(\mathbf{W}^{(i)})\|_F\leqslant\mathcal{O}(\sqrt{m})$. Therefore

$$\|\mathbf{W}_l^{(i+1)}-\mathbf{W}_l^{(1)}\|_F\leqslant\sum_{j=1}^{i}\|\mathbf{W}_l^{(j+1)}-\mathbf{W}_l^{(j)}\|_F\leqslant\mathcal{O}(\sqrt{m}\eta n).$$

Plugging in our parameter choice $\eta=\nu\epsilon/(Lm)$, $n=L^2R^2/(2\nu\epsilon^2)$ for some small enough absolute constant ν gives

$$\|\mathbf{W}_l^{(i+1)}-\mathbf{W}_l^{(1)}\|_F\leqslant\mathcal{O}\big(\sqrt{m}\cdot LR^2/(2m\epsilon)\big)\leqslant\omega,$$

where the last inequality holds as long as $m\geqslant C_2R^{16}L^{56}[\log(m)]^{12}\epsilon^{-14}$ for some large enough constant C_2. Therefore by induction we see that $\mathbf{W}^{(1)},\ldots,\mathbf{W}^{(n)}\in\mathcal{B}(\mathbf{W}^{(1)},\omega)$. As a result, the conditions of Lemmas 4.2 and B.3 are satisfied for $\mathbf{W}^*$ and $\mathbf{W}^{(1)},\ldots,\mathbf{W}^{(n)}$.

In the following, we utilize the results of Lemmas 4.2 and B.3 to prove the bound of cumulative loss. First of all, by Lemma 4.2, we have

$$\begin{aligned}L_i(\mathbf{W}^{(i)})-L_i(\mathbf{W}^*)&\leqslant\langle\nabla_{\mathbf{W}}L_i(\mathbf{W}^{(i)}),\mathbf{W}^{(i)}-\mathbf{W}^*\rangle+\epsilon\\&=\sum_{l=1}^{L}\frac{\langle\mathbf{W}_l^{(i)}-\mathbf{W}_l^{(i+1)},\mathbf{W}_l^{(i)}-\mathbf{W}_l^*\rangle}{\eta}+\epsilon\end{aligned}$$

Note that for the matrix inner product we have the equality $2\langle\mathbf{A},\mathbf{B}\rangle=\|\mathbf{A}\|_F^2+\|\mathbf{B}\|_F^2-\|\mathbf{A}-\mathbf{B}\|_F^2$. Applying this equality to the right hand side above gives

$$L_i(\mathbf{W}^{(i)})-L_i(\mathbf{W}^*)\leqslant\sum_{l=1}^{L}\frac{\|\mathbf{W}_l^{(i)}-\mathbf{W}_l^{(i+1)}\|_F^2+\|\mathbf{W}_l^{(i)}-\mathbf{W}_l^*\|_F^2-\|\mathbf{W}_l^{(i+1)}-\mathbf{W}_l^*\|_F^2}{2\eta}+\epsilon.$$

By Lemma B.3, for $l \in [L]$ we have $\|\mathbf{W}_l^{(i)} - \mathbf{W}_l^{(i+1)}\|_F \leqslant \eta \|\nabla_{\mathbf{W}_l} L_i(\mathbf{W}^{(i)})\|_F \leqslant \mathcal{O}(\eta\sqrt{m})$. Therefore

$$L_i(\mathbf{W}^{(i)}) - L_i(\mathbf{W}^*) \leqslant \sum_{l=1}^{L} \frac{\|\mathbf{W}_l^{(i)} - \mathbf{W}_l^*\|_F^2 - \|\mathbf{W}_l^{(i+1)} - \mathbf{W}_l^*\|_F^2}{2\eta} + \mathcal{O}(L\eta m) + \epsilon.$$

Telescoping over $i = 1, \ldots, n$, we obtain

$$\begin{aligned}\sum_{i=1}^{n} L_i(\mathbf{W}^{(i)}) &\leqslant \sum_{i=1}^{n} L_i(\mathbf{W}^*) + \sum_{l=1}^{L} \frac{\|\mathbf{W}_l^{(1)} - \mathbf{W}_l^*\|_F^2}{2\eta} + \mathcal{O}(L\eta n m) + n\epsilon \\ &\leqslant \sum_{i=1}^{n} L_i(\mathbf{W}^*) + \frac{LR^2}{2\eta m} + \mathcal{O}(L\eta n m) + n\epsilon,\end{aligned}$$

where in the first inequality we simply remove the term $-\|\mathbf{W}_l^{(n+1)} - \mathbf{W}_l^*\|_F^2/(2\eta)$ to obtain an upper bound, and the second inequality follows by the assumption that $\mathbf{W}^* \in \mathcal{B}(\mathbf{W}^{(1)}, Rm^{-1/2})$. Plugging in the parameter choice $\eta = \nu\epsilon/(Lm)$, $n = L^2R^2/(2\nu\epsilon^2)$ for some small enough absolute constant ν gives

$$\sum_{i=1}^{n} L_i(\mathbf{W}^{(i)}) \leqslant \sum_{i=1}^{n} L_i(\mathbf{W}^*) + 3n\epsilon,$$

which finishes the proof. □

B.4 Proof of Lemma 4.4

Here we prove Lemma 4.4. The proof essentially follows by standard Gaussian tail bound and a bound on the length of last hidden layer output vector.

Proof of Lemma 4.4. By Lemma 4.1 in Allen-Zhu et al. (2018b), with probability at least $1 - \mathcal{O}(nL) \cdot \exp[-\Omega(m/L)] > 1 - \delta/2$ over the randomness of $\mathbf{W}_1^{(1)}, \ldots, \mathbf{W}_{L-1}^{(1)}$, $\|\mathbf{h}_{i,L-1}^{(0)}\|_2 \in [1/2, 3/2]$ for all $i \in [n]$. Condition on $\mathbf{W}_1^{(1)}, \ldots, \mathbf{W}_{L-1}^{(1)}$, $f_{\mathbf{W}^{(1)}}(\mathbf{x}_i) = \sqrt{m} \cdot \mathbf{W}_L^{(1)} \mathbf{h}_{i,L-1}$ is a Gaussian random variable with variance $\|\mathbf{h}_{i,L-1}\|_2^2$. Therefore by standard Gaussian tail bound and union bound, with probability at least $1 - \delta$, $|f_{\mathbf{W}^{(1)}}(\boldsymbol{x}_i)| \leqslant \mathcal{O}(\sqrt{\log(n/\delta)})$ for all $i \in [n]$. □

C Proofs of Results in Section A

In this section we provide the proofs of Corollary A.2 and Lemma A.3.

C.1 Proof of Corollary A.2

The following lemma is a simplified version of Lemma C.2 in Cao and Gu (2019). Since the proof is almost the same as the proof of Lemma C.2 in Cao and Gu (2019), except replacing the ϵ-net argument with a simple union bound over n training examples, we omit the proof detail here.

Lemma C.1. For any $\delta > 0$, if $m \geqslant K \cdot 4^L L^4 \gamma^{-2} \log(nL/\delta)$ for some large enough absolute constant K, then with probability at least $1 - \delta$, there exists $\boldsymbol{\alpha}_{L-1} \in \mathbb{R}^m$ such that $y_i \cdot \langle \boldsymbol{\alpha}_{L-1}, \mathbf{h}_{i,L-1} \rangle \geqslant 2^{-L}\gamma$ for all $i \in [n]$.

Proof of Corollary A.2. Set $B = \log\{1/[\exp(n^{-1/2}) - 1]\} = \mathcal{O}(\log(n))$, then for cross-entropy loss we have $\ell(z) \leqslant n^{-1/2}$ for $z \geqslant B$. Moreover, let $B' = \max_{i \in [n]} |f_{\mathbf{W}^{(1)}}(\boldsymbol{x}_i)|$. Then by Lemma 4.4, with probability at least $1 - \delta$, $B' \leqslant \mathcal{O}(\sqrt{\log(n/\delta)})$ for all $i \in [n]$.

By Lemma C.1, with probability at least $1 - \delta$, there exists $\boldsymbol{\alpha}_{L-1} \in S^{m-1}$ such that $y_i \cdot \langle \boldsymbol{\alpha}_{L-1}, \mathbf{h}_{i,L-1} \rangle \geqslant 2^{-L}\gamma$ for all $i \in [n]$. Therefore, setting $R = (B + B') \cdot 2^L \gamma^{-1} = \widetilde{\mathcal{O}}(2^L \gamma^{-1})$, we have

$$\mathbf{W} = (\mathbf{0}, \ldots, \mathbf{0}, Rm^{-1/2} \cdot \boldsymbol{\alpha}_{L-1}^\top) \in \mathcal{B}(\mathbf{0}, Rm^{-1/2}).$$

Moreover, $f^*(\cdot) := f_{\mathbf{W}^{(1)}}(\cdot) + \langle \nabla_{\mathbf{W}} f_{\mathbf{W}^{(1)}}(\cdot), \mathbf{W} \rangle$ satisfies $f^* \in \mathcal{F}(\mathbf{W}^{(1)}, R)$, and

$$\begin{aligned} y_i \cdot f^*(\mathbf{x}_i) &= y_i \cdot f_{\mathbf{W}^{(1)}}(\mathbf{x}_i) + y_i \cdot \langle \sqrt{m} \cdot \mathbf{h}_{i,L-1}^\top, Rm^{-1/2} \cdot \boldsymbol{\alpha}_{L-1}^\top \rangle \\ &\geqslant (B + B') \cdot 2^L \gamma^{-1} \cdot 2^{-L}\gamma - B' \\ &\geqslant B. \end{aligned}$$

Therefore we have $\ell(y_i \cdot f^*(\mathbf{x}_i)) \leqslant \epsilon$, $i \in [n]$. Applying Theorem 3.3 gives

$$\mathbb{E}\big[L_{\mathcal{D}}^{0-1}(\widehat{\mathbf{W}})\big] \leqslant \widetilde{\mathcal{O}}\bigg(\frac{2^L \cdot \gamma^{-1}}{\sqrt{n}}\bigg) + \mathcal{O}\Bigg[\sqrt{\frac{\log(1/\delta)}{n}}\Bigg] = \widetilde{\mathcal{O}}\bigg(\frac{2^L \cdot \gamma^{-1}}{\sqrt{n}}\bigg).$$

This finishes the proof. □

C.2 Proof of Lemma A.3

Here we give the proof of Lemma A.3. It is based on a simple construction.

Proof of Lemma A.3. For any $f(x) = \mathbf{W}_2 \sigma(\mathbf{W}_1^{(1)} \mathbf{x})$ with $\|\mathbf{W}_2\|_F \leqslant Cm^{-1/2}$, by the assumption that $\|\mathbf{W}_2^{(1)}\|_F \leqslant Km^{-1/2}$ for some $K = \widetilde{\mathcal{O}}(1)$, we have $\mathbf{W}_2' := \mathbf{W}_2 - \mathbf{W}_2^{(1)}$ satisfies $\|\mathbf{W}_2'\|_F \leqslant (C + K) \cdot m^{-1/2}$. Therefore

$$f(x) = \mathbf{W}_2 \sigma(\mathbf{W}_1^{(1)} \mathbf{x}) = \mathbf{W}_2^{(1)} \sigma(\mathbf{W}_1^{(1)} \mathbf{x}) + \mathbf{W}_2' \sigma(\mathbf{W}_1^{(1)} \mathbf{x}) \subseteq \mathcal{F}.$$

This finishes the proof. □

D Proofs of Lemmas in Section B

In this section we give the proofs of lemma B.1, Lemma B.2 and Lemma B.3 in Section B.

D.1 Proof of Lemma B.1

Proof of Lemma B.1. By Lemma 4.1 in Allen-Zhu et al. (2018b), with probability at least $1 - \mathcal{O}(nL) \cdot \exp[-\Omega(m/L)]$, $\|\mathbf{h}_{i,l}^{(1)}\|_2 \in [3/4, 5/4]$ for all $i \in [n]$ and $l \in [L-1]$. Moreover, by Lemma 5.2 in Allen-Zhu et al. (2018b) and the 1-Lipschitz continuity of $\sigma(\cdot)$, with probability at least $1 - \mathcal{O}(nL) \cdot \exp[-\Omega(m\omega^{2/3}L)]$, $\|\mathbf{h}_{i,l} - \mathbf{h}_{i,l}^{(1)}\|_2 \leqslant \mathcal{O}(\omega L^{5/2}\sqrt{\log(m)})$. Therefore by the assumption that $\omega \leqslant \mathcal{O}(L^{-9/2}[\log(m)]^{-3})$, we have $\|\mathbf{h}_{i,l}\|_2 \in [1/2, 3/2]$ for all $i \in [n]$ and $l \in [L-1]$. □

D.2 Proof of Lemma B.2

We first introduce the following lemma characterizing the activation changes between networks with two close enough parameter sets $\mathbf{W}$ and $\mathbf{W}'$. This lemma directly follows by Lemma 5.2 in Allen-Zhu et al. (2018b) and triangle inequality.

Lemma D.1. If $\omega \leqslant \mathcal{O}(L^{-9/2}[\log(m)]^{-3/2})$, then with probability at least $1-\mathcal{O}(nL)\cdot\exp[-\Omega(m\omega^{2/3}L)]$,

$$\|\mathbf{D}_{i,l} - \mathbf{D}'_{i,l}\|_0 \leqslant \mathcal{O}(L\omega^{2/3}m)$$

for all $\mathbf{W}, \mathbf{W}' \in \mathcal{B}(\mathbf{W}^{(1)}, \omega)$, $i \in [n]$ and $l \in [L-1]$.

Proof of Lemma B.2. We first prove (i) and (iii), and then use (iii) to prove (ii).

By Lemma D.1, with probability at least $1 - \mathcal{O}(nL)\cdot\exp(-\Omega(L\omega^{2/3}m))$, $\|\mathbf{D}_{i,l} - \mathbf{D}^{(1)}_{i,l}\|_0 \leqslant \mathcal{O}(L\omega^{2/3}m)$ for all $i \in [n]$ and $l \in [L-1]$. Therefore we have $\|\mathbf{D}_{i,r} + \mathbf{D}''_{i,r} - \mathbf{D}^{(1)}_{i,l}\|_0 \leqslant \mathcal{O}(L\omega^{2/3}m)$ for all $i \in [n]$ and $l \in [L-1]$. Therefore by Lemma 5.6 in Allen-Zhu et al. (2018b), with probability at least $1 - \mathcal{O}(nL^2)\cdot\exp[-\Omega(m\omega^{2/3}L)]$ we have $\big\|\prod_{r=l_1}^{l_2}(\mathbf{D}_{i,r} + \mathbf{D}''_{i,r})\mathbf{W}_r\big\|_2 \leqslant \mathcal{O}(\sqrt{L})$. This completes the proof of (i) in Lemma B.2.

Similarly, to prove (iii), applying Lemma D.1 to $\mathbf{W}'$ gives that with probability at least $1 - \mathcal{O}(nL)\cdot\exp(-\Omega(L\omega^{2/3}m))$, $\|\mathbf{D}'_{i,l} + \mathbf{D}''_{i,r} - \mathbf{D}^{(1)}_{i,l}\|_0 \leqslant \mathcal{O}(L\omega^{2/3}m)$ for all $i \in [n]$ and $l \in [L-1]$. Now by Lemma 5.7 in Allen-Zhu et al. (2018b)[4] with $s = \mathcal{O}(m\omega^{2/3}L)$ to $\mathbf{W}$ and $\mathbf{W}'$, we have

$$\sqrt{m}\cdot\bigg\|\mathbf{W}_L^{(1)}\prod_{r=l_1}^{L-1}(\mathbf{D}'_{i,r} + \mathbf{D}''_{i,r})\mathbf{W}'_r - \mathbf{W}_L^{(1)}\prod_{r=l_1}^{L-1}\mathbf{D}^{(1)}_{i,r}\mathbf{W}^{(1)}_r\bigg\|_2 \leqslant \mathcal{O}\Big(\omega^{1/3}L^2\sqrt{m\log(m)}\Big), \tag{D.1}$$

$$\sqrt{m}\cdot\bigg\|\mathbf{W}_L^{(1)}\prod_{r=l_1}^{L-1}\mathbf{D}_{i,r}\mathbf{W}_r - \mathbf{W}_L^{(1)}\prod_{r=l_1}^{L-1}\mathbf{D}^{(1)}_{i,r}\mathbf{W}^{(1)}_r\bigg\|_2 \leqslant \mathcal{O}\Big(\omega^{1/3}L^2\sqrt{m\log(m)}\Big). \tag{D.2}$$

Moreover, by result (i), we have

$$\bigg\|(\mathbf{W}'_L - \mathbf{W}_L^{(1)})\prod_{r=l_1}^{L-1}(\mathbf{D}'_{i,r} + \mathbf{D}''_{i,r})\mathbf{W}'_r\bigg\|_2 \leqslant \mathcal{O}(\sqrt{L}\omega) \leqslant \mathcal{O}\Big(\omega^{1/3}L^2\sqrt{\log(m)}\Big), \tag{D.3}$$

$$\bigg\|(\mathbf{W}_L - \mathbf{W}_L^{(1)})\prod_{r=l_1}^{L-1}\mathbf{D}_{i,r}\mathbf{W}_r\bigg\|_2 \leqslant \mathcal{O}(\sqrt{L}\omega) \leqslant \mathcal{O}\Big(\omega^{1/3}L^2\sqrt{\log(m)}\Big). \tag{D.4}$$

Combining equations (D.1), (D.2), (D.3), (D.4) and applying triangle inequality gives the desired final result (iii).

[4] Note that $\sqrt{m}\cdot\mathbf{W}_L^{(1)}$ is a random vector following the Gaussian distribution $N(\mathbf{0}, \mathbf{I})$, which matches the distribution of the last layer parameters in Allen-Zhu et al. (2018b) for the binary classification case, where the output dimension of the network is 1.

Finally to prove (ii), we write

$$\left\|\mathbf{W}_L \prod_{r=l_1}^{L-1}(\mathbf{D}_{i,r}+\mathbf{D}''_{i,r})\mathbf{W}_r\right\|_2 \leqslant \left\|\mathbf{W}_L \prod_{r=l_1}^{L-1}(\mathbf{D}_{i,r}+\mathbf{D}''_{i,r})\mathbf{W}_r - \mathbf{W}_L^{(1)}\prod_{r=l_1}^{L-1}\mathbf{D}_{i,r}^{(1)}\mathbf{W}_r^{(1)}\right\|_2 + \left\|\mathbf{W}_L^{(1)}\prod_{r=l_1}^{L-1}\mathbf{D}_{i,r}^{(1)}\mathbf{W}_r^{(1)}\right\|_2.$$

Applying (iii) and (b) in Lemma 4.4 in Allen-Zhu et al. (2018b), with probability at least $1 - \mathcal{O}(nL)\cdot\exp[-\Omega(m/L)]$, we obtain

$$\left\|\mathbf{W}_L \prod_{r=l_1}^{L-1}(\mathbf{D}_{i,r}+\mathbf{D}''_{i,r})\mathbf{W}_r\right\|_2 \leqslant \mathcal{O}\Big(\omega^{1/3}L^2\sqrt{\log(m)}\Big) + \mathcal{O}(1) = \mathcal{O}(1).$$

This gives (ii). □

D.3 Proof of Lemma B.3

Proof of Lemma B.3. By Lemma B.1, clearly we have

$$\|\nabla_{\mathbf{W}_l} f_{\mathbf{W}}(\mathbf{x}_i)\|_F = \|\sqrt{m}\cdot\mathbf{h}_{i,L-1}\|_2 \leqslant \mathcal{O}(\sqrt{m})$$

for all $\mathbf{W}\in\mathcal{B}(\mathbf{W}^{(1)},\omega)$ and $i\in[n]$. For $l\in[L-1]$, by direct calculation we have

$$\begin{aligned}\|\nabla_{\mathbf{W}_l} f_{\mathbf{W}}(\mathbf{x}_i)\|_F &= \sqrt{m}\cdot\left\|\mathbf{h}_{i,l-1}\mathbf{W}_L\left(\prod_{r=l+1}^{L-1}\mathbf{D}_{i,r}\mathbf{W}_r\right)\mathbf{D}_{i,l}\right\|_F \\ &= \sqrt{m}\cdot\|\mathbf{h}_{i,l-1}\|_2\cdot\left\|\mathbf{W}_L\left(\prod_{r=l+1}^{L-1}\mathbf{D}_{i,r}\mathbf{W}_r\right)\mathbf{D}_{i,l}\right\|_2.\end{aligned}$$

Therefore by Lemma B.1 and (ii) in Lemma B.2, we have

$$\|\nabla_{\mathbf{W}_l} f_{\mathbf{W}}(\mathbf{x}_i)\|_F \leqslant \mathcal{O}(\sqrt{m}).$$

Finally, for $\|\nabla_{\mathbf{W}_l} L_i(\mathbf{W}^{(i)})\|_F$ we have

$$\|\nabla_{\mathbf{W}_l} L_i(\mathbf{W}^{(i)})\|_F \leqslant \big|\ell'[y_i\cdot f_{\mathbf{W}^{(i)}}(\mathbf{x}_i)]\cdot y_i\big|\cdot\big\|\nabla_{\mathbf{W}_l} f_{\mathbf{W}^{(i)}}(\mathbf{x}_i)\big\|_F \leqslant \sqrt{m}.$$

This completes the proof. □

References

Allen-Zhu, Z., Li, Y. and Liang, Y. (2018a). Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918* .

Allen-Zhu, Z., Li, Y. and Song, Z. (2018b). A convergence theory for deep learning via over-parameterization. *arXiv preprint arXiv:1811.03962* .

Arora, S., Du, S. S., Hu, W., Li, Z., Salakhutdinov, R. and Wang, R. (2019a). On exact computation with an infinitely wide neural net. *arXiv preprint arXiv:1904.11955* .

Arora, S., Du, S. S., Hu, W., Li, Z. and Wang, R. (2019b). Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584* .

Arora, S., Ge, R., Neyshabur, B. and Zhang, Y. (2018). Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint arXiv:1802.05296* .

Bartlett, P. L., Foster, D. J. and Telgarsky, M. J. (2017). Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*.

Brutzkus, A., Globerson, A., Malach, E. and Shalev-Shwartz, S. (2017). Sgd learns over-parameterized networks that provably generalize on linearly separable data. *arXiv preprint arXiv:1710.10174* .

Cao, Y. and Gu, Q. (2019). A generalization theory of gradient descent for learning over-parameterized deep relu networks. *arXiv preprint arXiv:1902.01384* .

Cesa-Bianchi, N., Conconi, A. and Gentile, C. (2004). On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory* **50** 2050–2057.

Daniely, A. (2017). Sgd learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems*.

Du, S. S., Lee, J. D., Li, H., Wang, L. and Zhai, X. (2018a). Gradient descent finds global minima of deep neural networks. *arXiv preprint arXiv:1811.03804* .

Du, S. S., Zhai, X., Poczos, B. and Singh, A. (2018b). Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054* .

Dziugaite, G. K. and Roy, D. M. (2017). Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008* .

E, W., Ma, C., Wu, L. et al. (2019). A comparative analysis of the optimization and generalization property of two-layer neural network and random feature models under gradient descent dynamics. *arXiv preprint arXiv:1904.04326* .

Golowich, N., Rakhlin, A. and Shamir, O. (2017). Size-independent sample complexity of neural networks. *arXiv preprint arXiv:1712.06541* .

He, K., Zhang, X., Ren, S. and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N. et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* **29** 82–97.

Jacot, A., Gabriel, F. and Hongler, C. (2018). Neural tangent kernel: Convergence and generalization in neural networks. *arXiv preprint arXiv:1806.07572* .

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*.

Langford, J. and Caruana, R. (2002). (not) bounding the true error. In *Advances in Neural Information Processing Systems*.

Lee, J., Xiao, L., Schoenholz, S. S., Bahri, Y., Sohl-Dickstein, J. and Pennington, J. (2019). Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720* .

Li, X., Lu, J., Wang, Z., Haupt, J. and Zhao, T. (2018). On tighter generalization bound for deep neural networks: Cnns, resnets, and beyond. *arXiv preprint arXiv:1806.05159* .

Li, Y. and Liang, Y. (2018). Learning overparameterized neural networks via stochastic gradient descent on structured data. *arXiv preprint arXiv:1808.01204* .

Neyshabur, B., Bhojanapalli, S., McAllester, D. and Srebro, N. (2017). A pac-bayesian approach to spectrally-normalized margin bounds for neural networks. *arXiv preprint arXiv:1707.09564* .

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y. and Srebro, N. (2018). The role of over-parametrization in generalization of neural networks .

Neyshabur, B., Tomioka, R. and Srebro, N. (2015). Norm-based capacity control in neural networks. In *Conference on Learning Theory*.

Oymak, S. and Soltanolkotabi, M. (2019). Towards moderate overparameterization: global convergence guarantees for training shallow neural networks. *arXiv preprint arXiv:1902.04674* .

Rahimi, A. and Recht, B. (2008). Random features for large-scale kernel machines. In *Advances in neural information processing systems*.

Rahimi, A. and Recht, B. (2009). Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in neural information processing systems*.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M. et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature* **529** 484–489.

Wei, C., Lee, J. D., Liu, Q. and Ma, T. (2018). On the margin theory of feedforward neural networks. *arXiv preprint arXiv:1810.05369* .

Yang, G. (2019). Scaling limits of wide neural networks with weight sharing: Gaussian process behavior, gradient independence, and neural tangent kernel derivation. *arXiv preprint arXiv:1902.04760* .

Yehudai, G. and Shamir, O. (2019). On the power and limitations of random features for understanding neural networks. *arXiv preprint arXiv:1904.00687* .

Zhang, C., Bengio, S., Hardt, M., Recht, B. and Vinyals, O. (2016). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530* .

Zou, D., Cao, Y., Zhou, D. and Gu, Q. (2018). Stochastic gradient descent optimizes over-parameterized deep relu networks. *arXiv preprint arXiv:1811.08888* .