Regularization Matters: A Nonparametric Perspective on Overparametrized Neural Network Trained by Gradient Descent

Tianyang Hu*
hu478@purdue.edu
Purdue University

Wenjia Wang*
wenjiawang@ust.hk
Hong Kong University of Science and Technology

Cong Lin
52174404011@stu.ecnu.edu.cn
East China Normal University

Guang Cheng
chengg@purdue.edu
Purdue University

Abstract

Overparametrized neural networks trained by gradient descent (GD) can provably overfit any training data. However, the generalization guarantee may not hold for noisy data. From a nonparametric perspective, this paper studies how well overparametrized neural networks can recover the true target function in the presence of random noises. We establish a lower bound on the L_2 estimation error with respect to the GD iterations, which is away from zero without a delicate scheme of early stopping. In turn, through a comprehensive analysis of ℓ_2-regularized GD trajectories, we prove that for overparametrized one-hidden-layer ReLU neural network with the ℓ_2 regularization: (1) the output is close to that of the kernel ridge regression with the corresponding neural tangent kernel; (2) minimax optimal rate of the L_2 estimation error can be achieved. Numerical experiments confirm our theory and further demonstrate that the ℓ_2 regularization approach improves the training robustness and works for a wider range of neural networks.

1 INTRODUCTION

Deep learning has shown outstanding empirical successes and demonstrates superior performance in many standard machine learning tasks, such as image classification [Krizhevsky et al., 2012, LeCun et al., 2015, He et al., 2016], generative modeling [Goodfellow et al., 2014, Arjovsky et al., 2017], etc. Despite common accusations of being a black box with no theoretical guarantee, deep neural network (DNN) tends to achieve higher accuracy than other classical methods in various prediction tasks, which attracts plenty of interests from researchers. In contrast to the huge empirical success, little is yet settled from the theoretical side why DNN outperforms other methods. Without enough understanding, practical use of deep learning models could be inefficient and unreliable.

Recently, many efforts have been devoted to provable deep learning methods with algorithmic guarantees, particularly training overparametrized neural networks by gradient descent (GD) or other gradient-based optimization. It has been shown that with enough overparametrization, e.g., neural network width tends to infinity, training DNN resembles a kernel method with a specific kernel called as "neural tangent kernel" (NTK) [Jacot et al., 2018]. In the NTK regime, GD can provably minimize the training error to zero in both regression [Du et al., 2018, Li and Liang, 2018, Arora et al., 2019, Zou and Gu, 2019] and classification [Ji and Telgarsky, 2019a,b, Lyu and Li, 2019] settings. Corresponding generalization error bounds are developed to ensure prediction performance on unseen data. However, a closer inspection of these generalization results reveals that they only hold under the noiseless assumption, i.e., the response variable is deterministic given the explanatory variables. For overparametrized neural networks, the training loss can be minimized to zero so that the generalization error equals the population loss, which cannot be zero in the presence of noises. As random noises are ubiquitous in the real world, theoretical guarantees and provable learning algorithms that take into account of random noises are much needed in practice.

In contrast, classic nonparametric statistics literature demonstrate that in the presence of noises, the L_2 estimation error can still go to zero with possibly optimal rates as established in Stone [1982]. To further investi-

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

* These authors contributed equally to this work.

gate how overparametrized neural networks trained via GD work and how well they can learn the underlying true function with noisy data, we consider the classic nonparametric regression setting. Suppose we observe data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, given by

$$y_i = f^*(\boldsymbol{x}_i) + \epsilon_i, \tag{1.1}$$

where f^* is the ground truth, $\boldsymbol{x}_i \in \mathbb{R}^d$, and ϵ_i's are i.i.d. random noises with mean 0 and finite variance σ^2. In this work, we consider neural network estimators $\widehat{f}$ produced by overparametrized one-hidden-layer ReLU neural networks, where the number of neurons can be much larger than the sample size, and investigate how fast the L_2 estimation error $\left\|\widehat{f} - f^*\right\|_2$ converges to zero as sample size grows.

Note that the L_2 convergence rate critically depends on the assumptions of the true function, e.g., linearity, smoothness, boundedness, etc., based on which minimax lower bounds are established [Siegel, 1957]. An estimation method is said to be *minimax-optimal* if its convergence rate achieves the lower bound, indicating that it performs the best in the worst possible scenario. The above nonparametric perspective provides a sharp characterization of the employed estimation method and complements the existing optimization/generalization framework.

The main contributions of this paper are:

- We prove that overparametrized one-hidden-layer ReLU neural networks trained using GD do not recover the true function in the classic nonparametric regression setting (1.1), i.e., the L_2 estimation error is bounded away from zero as sample size goes to infinity. To predict well on unseen data, a delicate early stopping rule has to be deployed.
- We analyze the ℓ_2-regularized GD trajectory and show that the ℓ_2 penalty on network weights amounts to penalizing the reproducing kernel Hilbert space (induced by NTK) norm of the associated neural network. With ℓ_2 regularization, overparametrized neural network trained by GD resembles the solution of kernel ridge regression.
- We further prove that by adding proper ℓ_2 regularization, overparametrized neural network trained by GD achieves the *minimax-optimal* L_2 convergence rate $n^{-d/(4d-2)}$, in recovering the ground truth in (1.1).

The correspondence between overparametrized neural network trained by ℓ_2-regularized GD and kernel ridge regression is nontrivial and technically challenging. In spite of the well-established equivalence between NTK and infinite-width DNN trained by GD, there is a huge technical gap for finite-width overparametrized neural networks, especially when the training objective includes explicit regularization terms.

To sum up, this work broadens the current scope of the NTK literature and connects the recent advances in deep learning theory, e.g., analyzing the trajectory of GD updates, implicit bias of overparametrization, etc., to the classical results in nonparametric statistics. More specifically, our findings not only contribute to the theoretical (in particular, nonparametric) understanding of training overparametrized DNN on noisy data but also promotes the use of ℓ_2 penalty or weight decay in practice for better theoretical guarantees.

2 RELATED WORKS

Neural Tangent Kernel The seminal paper [Jacot et al., 2018] proves that the evolution of DNNs during training can be described by the so-called neural tangent kernel (NTK), which is central to characterize the convergence and generalization behaviors. Du et al. [2018], Arora et al. [2019], Li and Liang [2018] investigate specifically for one-hidden-layer ReLU neural networks and show explicitly that with enough overparametrization, the weight vectors and the corresponding NTK do not change much during GD training. Similar investigations have been done for other neural networks and other settings [Zou and Gu, 2019, Ji and Telgarsky, 2019b]. Among others, Arora et al. [2019], Cao and Gu [2019] provide generalization error bounds and provable learning scenarios, but only hold for noiseless data.

For noisy data, explicit regularizations have recently been considered in the NTK literature. Wei et al. [2019] promote the ℓ_2 penalty when using NTK by showing that in a constructed classification example, sample efficiency can benefit from the regularization. Hu et al. [2020] consider classification with noisy labels and propose to add ℓ_2 regularization to ensure robustness. However, their analyses only apply to the kernel estimator directly using NTK and only relate to infinite width neural networks, which greatly restricts the model class capacity. As pointed out before, bridging the technical gap between NTK and finite-width overparametrized neural networks is technically challenging when the training objective includes an ℓ_2 regularization term and we should not take it for granted. Geifman et al. [2020] demonstrate the similarity between the Laplace kernels and ReLU NTKs. However, in order for NTK to be a good characterization of neural network training, how wide is wide enough remains an active field of research [Nitanda et al., 2019]. In comparison, we directly analyze GD

trajectories of training finite-width neural networks (with and without ℓ_2 regularization) and prove that the corresponding NTK solutions can be well-approximated after a polynomial number of GD iterations. To the best of our knowledge, we are among the first to rigorously establish the L_2 convergence rate for trained neural networks under noisy data. Nitanda and Suzuki [2020] recently provide similar convergence rate analysis by considering a particular penalized stochastic gradient descent algorithm but they require the neural network width to be exponential with n.

Nonparametric Regression In nonparametric statistics, Stone [1982] shows that when f^* is d-variate and β-time differentiable, the optimal rate of convergence for the L_2 estimation error is $n^{-\beta/(2\beta+d)}$. Many popular methods such as kernel methods, Gaussian process, splines, etc., achieve this rate. It has been recently shown that DNN (with certain structures) can also achieve optimal convergence rates [Yarotsky, 2017, Schmidt-Hieber, 2017, Bauer et al., 2019, Liu et al., 2019] and even for non-smooth functions [Imaizumi and Fukumizu, 2018]. However, this type of results has two limitations. Firstly, they only apply to the empirical risk minimizer or some specially constructed DNNs without any algorithmic guarantee. Secondly, the theoretical analysis relies on delicate complexity control of the DNN family and cannot handle overparametrization, which is very common in practice. Therefore, the aforementioned results are less helpful in understanding deep neural network models with overparametrization and highly non-convex optimization properties.

Our algorithm-dependent statistical analysis bridges the gap between these two types of research. Based on the GD trajectories and the corresponding NTK, we are able to analyze the trained overparametrized neural networks within the nonparametric framework and show they can also achieve the optimal convergence rate with proper regularizations.

3 PRELIMINARIES

Notation For any function $f(\boldsymbol{x}) : \mathcal{X} \to \mathbb{R}$, denote $\|f\|_\infty = \sup_{\boldsymbol{x}\in\mathcal{X}} |f(\boldsymbol{x})|$ and $\|f\|_p = (\int_{\mathcal{X}} |f(\boldsymbol{x})|^p d\boldsymbol{x})^{1/p}$. For any vector $\boldsymbol{x}$, $\|\boldsymbol{x}\|_p$ denotes its p-norm, for $1 \le p \le \infty$. L_p and l_p are used to distinguish function norms and vector norms. For two given sequences $\{a_n\}_{n\in\mathbb{N}}$ and $\{b_n\}_{n\in\mathbb{N}}$ of real numbers, we write $a_n \lesssim b_n$ if there exists a constant $C > 0$ such that $a_n \le C b_n$ for all sufficiently large n. Let $\Omega(\cdot)$ be the counterpart of $O(\cdot)$ that $a_n = \Omega(b_n)$ means $a_n \gtrsim b_n$. Further, $a_n = \widetilde{O}(b_n)$ and $a_n = \widetilde{\Omega}(b_n)$ are used to indicate there are specific requirements for the multiplicative constants. We write $a_n \asymp b_n$ if $a_n \lesssim b_n$ and $a_n \gtrsim b_n$. Let $[N] = \{1, \ldots, N\}$ for $N \in \mathbb{N}$ and let $\lambda_{\min}(\boldsymbol{A})$ be the minimum eigenvalue of a symmetric matrix $\boldsymbol{A}$. We use $\mathbb{I}$ to denote the indicator function and $\boldsymbol{I}_d$ to denote the $d \times d$ identity matrix. $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ represents Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ and $\mathrm{poly}(t_1, t_2, \ldots)$ denotes some polynomial function with arguments $t_1, t_2, \ldots$.

Neural Network Setup Consider the one-hidden-layer ReLU neural network family $\mathcal{F}$ with m nodes in the hidden layer, expressed as

$$f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}) = \frac{1}{\sqrt{m}} \sum_{r=1}^{m} a_r \sigma(\boldsymbol{w}_r^\top \boldsymbol{x}),$$

where $\boldsymbol{x} \in \mathbb{R}^d$ denotes the input, $\boldsymbol{W} = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_m) \in \mathbb{R}^{d\times m}$ is the weight matrix in the hidden layer, $\boldsymbol{a} = (a_1, \cdots, a_m)^\top \in \mathbb{R}^m$ is the weight vector in the output layer, $\sigma(z) = \max\{0, z\}$ is the rectified linear unit (ReLU). The initial values of the weights are independently generated from

$$\boldsymbol{w}_r(0) \sim N(\boldsymbol{0}, \tau^2 \boldsymbol{I}_m), \quad a_r \sim \mathrm{unif}\{-1, 1\}, \quad \forall r \in [m].$$

When $m \gg n$, the neural network is highly overparametrized. As is usually assumed in the NTK literature [Arora et al., 2019, Hu et al., 2020, Bietti and Mairal, 2019], we consider data on the unit sphere $\mathbb{S}^{d-1}$, i.e., $\|\boldsymbol{x}_i\|_2 = 1$ for any $i \in [n]$. Throughout this work, we further assume that $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are uniformly distributed on $\mathbb{S}^{d-1}$ so that $\mathbb{E}_{\boldsymbol{x}\sim\mathrm{unif}(\mathbb{S}^{d-1})}(\widehat{f}(\boldsymbol{x}) - f^*(\boldsymbol{x}))^2$ and $\|f - f^*\|_2^2$ are equal up to a constant multiplier and thus will be used interchangeably.

Gradient Descent Let $\boldsymbol{y} = (y_1, \cdots, y_n)^\top$ and $\boldsymbol{\epsilon} = (\epsilon_1, \cdots, \epsilon_n)^\top$. Denote $u_i = f_{\boldsymbol{W},\boldsymbol{a}}(\boldsymbol{x}_i)$ to be the network's prediction on $\boldsymbol{x}_i$ and let $\boldsymbol{u} = (u_1, ..., u_n)^\top$. Without loss of generality, we consider fixing the second layer $\boldsymbol{a}$ after initialization and only training the first layer $\boldsymbol{W}$ by GD. Fixing the last layer is not a strong restriction since $a \cdot \sigma(z) = \mathrm{sign}(a) \cdot \sigma(|a|z)$ and we can always reparametrize the network to have all a_i's to be either 1 or -1. Denote the empirical squared loss as $\Phi(\boldsymbol{W}) = \frac{1}{2}\|\boldsymbol{y} - \boldsymbol{u}\|_2^2$. The gradient of $\Phi(\boldsymbol{W})$ w.r.t. $\boldsymbol{w}_r$ can be written as

$$\frac{\partial \Phi(\boldsymbol{W})}{\partial \boldsymbol{w}_r} = \frac{1}{\sqrt{m}} a_r \sum_{i=1}^{n} (u_i - y_i) \mathbb{I}_{r,i} \boldsymbol{x}_i, \quad r \in [m],$$

where $\mathbb{I}_{r,i} = \mathbb{I}\{\boldsymbol{w}_r^\top \boldsymbol{x}_i \ge 0\}$. Then the GD update rule at the k-th iteration is given by

$$\boldsymbol{w}_r(k+1) = \boldsymbol{w}_r(k) - \eta \frac{\partial \Phi(\boldsymbol{W})}{\partial \boldsymbol{w}_r}\bigg|_{\boldsymbol{W}=\boldsymbol{W}(k)},$$

where $\eta > 0$ is the step size (a.k.a. learning rate). In the rest of this work, we use k to index variables at the

k-th iteration, e.g., $u_i(k) = f_{\boldsymbol{W}(k),\boldsymbol{a}}(\boldsymbol{x}_i)$, etc. Define $\mathbb{I}_{r,i}(k) = \mathbb{I}\{\boldsymbol{w}_r(k)^\top \boldsymbol{x}_i \geq 0\}$, $\boldsymbol{Z}(k) \in \mathbb{R}^{md\times n}$ that

$$\boldsymbol{Z}(k) = \frac{1}{\sqrt{m}} \begin{pmatrix} a_1\mathbb{I}_{1,1}(k)\boldsymbol{x}_1 & \dots & a_1\mathbb{I}_{1,n}(k)\boldsymbol{x}_n \\ \vdots & \ddots & \vdots \\ a_m\mathbb{I}_{m,1}(k)\boldsymbol{x}_1 & \dots & a_m\mathbb{I}_{m,n}(k)\boldsymbol{x}_n \end{pmatrix}$$

and $\boldsymbol{H}(k) = \boldsymbol{Z}(k)^\top \boldsymbol{Z}(k)$. It is shown that matrices $\boldsymbol{Z}(k)$ and $\boldsymbol{H}(k)$ are close to $\boldsymbol{Z}(0)$ and $\boldsymbol{H}(0)$, respectively for any k, when m is sufficiently large [Arora et al., 2019]. We can rewrite the GD update rule as

$$\mathrm{vec}(\boldsymbol{W}(k+1)) = \mathrm{vec}(\boldsymbol{W}(k)) - \eta \boldsymbol{Z}(k)(\boldsymbol{u}(k) - \boldsymbol{y}), \tag{3.1}$$

where $\mathrm{vec}(\boldsymbol{W}) = (\boldsymbol{w}_1^\top, \cdots, \boldsymbol{w}_m^\top)^\top \in \mathbb{R}^{md\times 1}$ is the vectorized weight matrix.

Kernel Ridge Regression with NTK The study of one-hidden-layer ReLU neural networks is closely related to the NTK defined as

$$\begin{aligned} h(\boldsymbol{s},\boldsymbol{t}) =& \mathbb{E}_{\boldsymbol{w}\sim N(0,\boldsymbol{I}_d)} \left(\boldsymbol{s}^\top \boldsymbol{t}\, \mathbb{I}\{\boldsymbol{w}^\top \boldsymbol{s} \geq 0, \boldsymbol{w}^\top \boldsymbol{t} \geq 0\}\right) \\ =& \frac{\boldsymbol{s}^\top \boldsymbol{t}(\pi - \arccos(\boldsymbol{s}^\top \boldsymbol{t}))}{2\pi}, \end{aligned} \tag{3.2}$$

where $\boldsymbol{s}, \boldsymbol{t}$ are d-dimensional vectors. It can be shown that h is positive definite on the unit sphere $\mathbb{S}^{d-1}$ [Bietti and Mairal, 2019]. Let the Mercer decomposition of h be $h(\boldsymbol{s},\boldsymbol{t}) = \sum_{j=0}^{\infty} \lambda_j \varphi_j(\boldsymbol{s})\varphi_j(\boldsymbol{t})$, where $\lambda_1 \geq \lambda_2 \geq ... \geq 0$ are the eigenvalues, and $\{\varphi_j\}_{j=1}^{\infty}$ is an orthonormal basis.

The following lemma states the decay rate of eigenvalues of the NTK associated with one-hidden-layer ReLU neural networks, as a key technical contribution of this work.

Lemma 3.1. Let λ_j be the eigenvalues of NTK h defined above. Then we have $\lambda_j \asymp j^{-\frac{d}{d-1}}$.

Let $\mathcal{N}$ denote the reproducing kernel Hilbert space (RKHS) generated by h on $\mathbb{S}^{d-1}$, equipped with norm $\|\cdot\|_{\mathcal{N}}$. For an unknown function $f^* \in \mathcal{N}$, the kernel ridge regression minimizes

$$\min_{f\in\mathcal{N}} \frac{1}{2}\sum_{i=1}^{n}(y_i - f(\boldsymbol{x}_i))^2 + \frac{\mu}{2}\|f\|_{\mathcal{N}}^2, \tag{3.3}$$

where $\mu > 0$ is a tuning parameter controlling the regularization strength. The representer theorem says that the solution to (3.3) can be written as

$$\widehat{f}(\boldsymbol{x}) = h(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{H}^\infty + \mu \boldsymbol{I}_n)^{-1}\boldsymbol{y} \tag{3.4}$$

for any point $\boldsymbol{x} \in \mathbb{R}^d$, where $h(\boldsymbol{x},\boldsymbol{X}) = (h(\boldsymbol{x},\boldsymbol{x}_1), ..., h(\boldsymbol{x},\boldsymbol{x}_n)) \in \mathbb{R}^{1\times n}$ and $\boldsymbol{H}^\infty = (h(\boldsymbol{x}_i,\boldsymbol{x}_j))_{n\times n}$ ($\boldsymbol{H}^\infty$ is usually called the NTK matrix). In the following theorem, we show that the function $\widehat{f}$ is close to the true function f^* under the L_2 metric.

Theorem 3.2. Let $\widehat{f}$ be as in (3.4). By choosing $\mu \asymp n^{(d-1)/(2d-1)}$, we have

$$\left\|\widehat{f} - f^*\right\|_2^2 = O_{\mathbb{P}}\left(n^{-\frac{d}{2d-1}}\right), \quad \left\|\widehat{f}\right\|_{\mathcal{N}}^2 = O_{\mathbb{P}}(1).$$

The proof of the convergence rate requires an accurate characterization of the complexity of $\mathcal{N}$, which is determined by the eigenvalues and eigenfunction expansion of the NTK h. If the eigenvalues decay at rate $\lambda_j \asymp j^{-2\nu}$, the corresponding minimax optimal rate is $n^{-2\nu/(2\nu+1)}$ [Yuan et al., 2016, Raskutti et al., 2014]. Building on the the eigenvalue decay rate established in Lemma 3.1, it can be shown that the L_2 estimation rate in Theorem 3.2 is minimax-optimal.

In the rest of this work, we assume that $f^* \in \mathcal{N}$.

4 PROBLEMS OF GRADIENT DESCENT FROM THE NONPARAMETRIC PERSPECTIVE

In this section, we consider training overparametrized neural networks with the GD update rule (3.1). Among others, Arora et al. [2019], Du et al. [2018] prove that as iteration $k \to \infty$, the training data are interpolated, achieving zero training loss. However, in the presence of noises, i.e., ϵ_i in (1.1), such an overfitting to the training data can be harmful for recovering the ground truth. The following theorem shows that if k is too small or too large, the L_2 estimation error of the trained neural network is bounded away from zero.

Theorem 4.1. Fix a failure probability $\delta \in (0,1)$. Let λ_0 be the largest number that with probability at least $1-\delta$, $\lambda_{\min}(\boldsymbol{H}^\infty) \geq \lambda_0$. Suppose $m \geq \tau^{-2}\mathrm{poly}\left(n, \frac{1}{\lambda_0}, \frac{1}{\delta}\right)$, $\eta = \widetilde{O}\left(\frac{\lambda_0}{n^2}\right)$, and $\tau = \widetilde{O}\left(\frac{\lambda_0\delta}{n}\right)$. For sufficiently large n, if the iteration $k = \widetilde{\Omega}\left(\frac{\log n}{\eta\lambda_0}\right)$ or $k = \widetilde{O}\left(\frac{1}{n\eta}\right)$, then with probability at least $1-2\delta$, we have

$$\mathbb{E}_{\boldsymbol{\epsilon}}\left\|f_{\boldsymbol{W}(k),\boldsymbol{a}} - f^*\right\|_2^2 = \Omega(1).$$

The conditions on m, η, and τ have the same rates as those in Theorem 5.1 of Arora et al. [2019], but the constants requirements are different. The probability $1-2\delta$ in Theorem 4.1 comes from the randomness of $\lambda_{\min}(\boldsymbol{H}^\infty)$ and $(\boldsymbol{W}(0), \boldsymbol{a})$.

Theorem 4.1 states that the estimation error for non-regularized one-hidden-layer neural networks is bounded away from zero by some constant if trained for too short or too long. The latter scenario indicates that overfitting is harmful in terms of the L_2 estimation error. Similar results have been shown in

Kohler and Krzyzak [2019] for specifically designed overparametrized DNNs that is a linear combination of $\Omega(n^{10d^2})$ smaller neural networks, which is much more restrictive than ours.

In order to have low L_2 estimation errors, Theorem 4.1 implies that the iteration number k must satisfy $(\eta\lambda_0)^{-1}\log n \lesssim k \lesssim (n\eta)^{-1}$. However, deriving a precise order of k, which leads to the optimal rate of convergence, could be extremely challenging. Alternatively, we consider the infinite-width limit of one-hidden-layer ReLU networks, i.e., directly using the NTK (3.2) in kernel regression. This may shed some light on the optimal stopping time for practical overparametrized neural networks.

In kernel regression, the objective becomes

$$\min_{f\in\mathcal{N}} \frac{1}{2}\sum_{i=1}^{n}(y_i - f(\boldsymbol{x}_i))^2, \tag{4.1}$$

whose solution can be explicitly expressed as $h(\boldsymbol{x},\boldsymbol{X})(\boldsymbol{H}^{\infty})^{-1}\boldsymbol{y}$, by setting $\mu = 0$ in (3.4). However, inverting the kernel matrix can be computationally intensive. In practice, gradient-based methods are often applied to solve (4.1) [Raskutti et al., 2014]. The following theorem establishes estimation error results for the NTK estimators trained by GD, complementary to Theorem 4.1.

Theorem 4.2. Consider using GD to optimize (4.1) with a sufficiently small step size η depending on n (but not on k). There exists a stopping time k^* depending on data, such that

$$\mathbb{E}\left\|\widehat{f}_{k^*} - f^*\right\|_2^2 = O\left(n^{-\frac{d}{2d-1}}\right),$$

where $\widehat{f}_k$ is the predictor obtained at the k-th iteration. Moreover, if $k\to\infty$, the interpolated estimator $\widehat{f}_{\infty}$ satisfies

$$\mathbb{E}\left\|\widehat{f}_{\infty} - f^*\right\|_2^2 = \Omega(1).$$

To specify the optimal stopping time k^* in Theorem 4.2, we first introduce the local empirical Rademacher complexity defined as

$$\widehat{\mathcal{R}}_{\boldsymbol{H}^{\infty}}(\varepsilon) := \left(\frac{1}{n}\sum_{i=1}^{n}\min\left\{\widehat{\lambda}_i/n, \varepsilon^2\right\}\right)^{1/2},$$

which relies on the eigenvalues $\widehat{\lambda}_1 \geq \cdots \geq \widehat{\lambda}_n > 0$ of $\boldsymbol{H}^{\infty}$. Then, the stopping time k^* is defined to be

$$k^* := \operatorname{argmin}\left\{k\in\mathbb{N} \mid \widehat{\mathcal{R}}_{\boldsymbol{H}^{\infty}}\left(\frac{1}{\sqrt{\eta k}}\right) > \frac{1}{2e\sigma\eta k}\right\} - 1. \tag{4.2}$$

In essence, the optimal stopping time decreases with the noise level σ and increases with the model complexity, measured by the eigenvalues of $\boldsymbol{H}^{\infty}$.

Remark 1. (k^* for neural networks) To derive the order of k^* for overparametrized neural network, a sharp characterization of the eigen-distribution of $\boldsymbol{H}^{\infty}$ is needed. To the best of the authors' knowledge, no such results are available yet. Even though as $m\to\infty$, neural network resembles its linearization (NTK), it doesn't necessarily mean such a stopping rule can be easily derived for finite-width neural networks. In general, theoretical guarantees of an early stopping rule for training overparametrized neural networks is challenging and left for future work.

Besides early stopping, explicit regularizations are usually employed in deep learning models to balance the bias-variance trade-off and prevent overfitting, for example, weight decay [Krogh and Hertz, 1992], batch normalization [Ioffe and Szegedy, 2015], dropout [Srivastava et al., 2014], etc., to prevent overfitting. In the next section, we investigate the ℓ_2 regularization [Bilgic et al., 2014, Van Laarhoven, 2017, Phaisangittisagul, 2016] and demonstrate its effectiveness in the nonparametric regression setting.

5 ℓ_2-REGULARIZED GRADIENT DESCENT FOR NOISY DATA

Without any regularization, GD overfits the training data and the estimation error is bounded away from zero. Instead, we propose using the ℓ_2-regularized gradient descent defined as

$$\begin{aligned}\operatorname{vec}(\boldsymbol{W}_D(k+1)) =& \operatorname{vec}(\boldsymbol{W}_D(k)) - \eta_1\boldsymbol{Z}_D(k)(\boldsymbol{u}_D(k)-\boldsymbol{y})\\ &- \eta_2\mu\operatorname{vec}(\boldsymbol{W}_D(k)),\end{aligned} \tag{5.1}$$

where $\eta_1,\eta_2 > 0$ are step sizes, and $\mu > 0$ is a tuning parameter. It can be easily seen that (5.1) is the GD update rule on the following loss function

$$\Phi_1(\boldsymbol{W}) = \frac{1}{2}\|\boldsymbol{y}-\boldsymbol{u}\|_2^2 + \frac{\mu}{2}\|\operatorname{vec}(\boldsymbol{W})\|_2^2. \tag{5.2}$$

The ℓ_2 regularization has long been used in practical training neural networks and is equivalent to "weight decay" [Krogh and Hertz, 1992] when using GD [Loshchilov and Hutter, 2017]. In the NTK literature, ℓ_2 regularization is also considered as a way to improve generalization [Wei et al., 2019, Hu et al., 2020]. However, we are among the first to directly analyze the ℓ_2-regularized GD trajectories of overparametrized neural networks and show its connection to kernel ridge regression using NTK. In the rest of this work, we use subscript D to denote the variables under the regularized GD (5.1), e.g., $\boldsymbol{u}_D(k)$ for the predictions at the k-th iteration.

Theorem 5.1. Let λ_0 be the largest number such that with probability at least $1-\delta_n$, $\lambda_{\min}(\boldsymbol{H}^{\infty}) \geq \lambda_0$,

and $\delta_n \to 0$ as n goes to infinity[1]. For sufficiently large n, suppose $\mu \asymp n^{\frac{d-1}{2d-1}}$, $\eta_1 \asymp \eta_2 = o(n^{-\frac{3d-1}{2d-1}})$, $\tau = O(1)$, $m \geq \tau^{-2}\text{ploy}(n, \lambda_0^{-1})$, and the iteration number k satisfies $\log\left(\text{ploy}_1(n, \tau, 1/\lambda_0)\right) \lesssim \eta_2\mu k \lesssim \log\left(\text{ploy}_2(\tau, 1/n, \sqrt{m})\right)$. Then we have

$$\left\|\boldsymbol{u}_D(k) - \boldsymbol{H}^{\infty}(C\mu I + \boldsymbol{H}^{\infty})^{-1}\boldsymbol{y}\right\|_2 = O_{\mathbb{P}}\left(\sqrt{n}(1-\eta_2\mu)^k\right), \tag{5.3}$$

$$\left\|\text{vec}(\boldsymbol{W}_D(k)) - (1-\eta_2\mu)^k\text{vec}(\boldsymbol{W}_D(0))\right\|_2 = O_{\mathbb{P}}(1), \tag{5.4}$$

for some constant $C > 0$. Moreover, during the training process, the mean squared loss satisfies

$$\Phi(\boldsymbol{W}_D(k))/n \leq (1-\eta_2\mu)^k\Phi(\boldsymbol{W}_D(0))/n + O_{\mathbb{P}}(1). \tag{5.5}$$

In the above theorem, three upper bounds are provided. In (5.3), we provide an upper bound on the difference between the prediction using one-hidden-layer neural networks and that obtained by (3.4), which converges to zero as the sample size goes to infinity. This indicates that the ℓ_2 penalty on neural network weights has similar effects to penalizing the RKHS norm as in (3.3). Combining (5.3) and Theorem 3.2, we can conclude that the ℓ_2-regularized one-hidden-layer ReLU neural network recovers the true function on the training data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$.

In (5.4), we provide an upper bound on the distance between the weight matrix at the k-th iteration and the "decayed" initialization $\boldsymbol{W}_D(0)$. Under the conditions in Theorem 5.1, their distance measured in Frobenius norm is bounded by some constant depending on the underlying true function. Unlike the results in Arora et al. [2019], the upper bound presented in (5.4) does not depend on data. Therefore, as long as the underlying function is within the RKHS generated by NTK, the total movement of all the weights is not large even if the data observed are corrupted by noises.

In (5.5), we give a characterization of how the training objective decreases over iterations, which is reminiscent of Theorem 4.1 in Du et al. [2018]. Unlike the results without regularization, our ℓ_2-regularized objective is not expected to converge to zero, i.e., no data interpolation, which is essential to ensure the best trade-off between the bias and variance.

Remark 2. (More iterations) The required iteration number k in Theorem 5.1 is approximately $(\eta_2\mu)^{-1}$, up to a logarithmic term. We believe the upper bound on k is not necessary and may be relaxed. The stated results are expected to hold if $k \to \infty$ and we conjecture that the output will converge to the optimal solution of kernel ridge regression as in (3.4). Simulation results in Section 6 support our conjecture and we leave the technical proof for future work.

Next, we extend the results in Theorem 5.1 and establish the L_2 convergence rate for neural networks trained with ℓ_2-regularized GD.

Theorem 5.2. Suppose the assumptions of Theorem 5.1 hold. Then we have

$$\left\|f_{\boldsymbol{W}_D(k),\boldsymbol{a}} - f^*\right\|_2^2 = O_{\mathbb{P}}(n^{-\frac{d}{2d-1}}).$$

The above theorem states that with probability tending to one, the neural network estimator can still recover the true function with the optimal convergence rate of $n^{-\frac{d}{2(2d-1)}}$, demonstrating the effectiveness of the ℓ_2 regularization for noisy data. Unlike other optimality results established for neural networks [Schmidt-Hieber, 2017, Bauer et al., 2019], our convergence rate result applies to overparametrized networks and is obtainable using the ℓ_2-regularized GD.

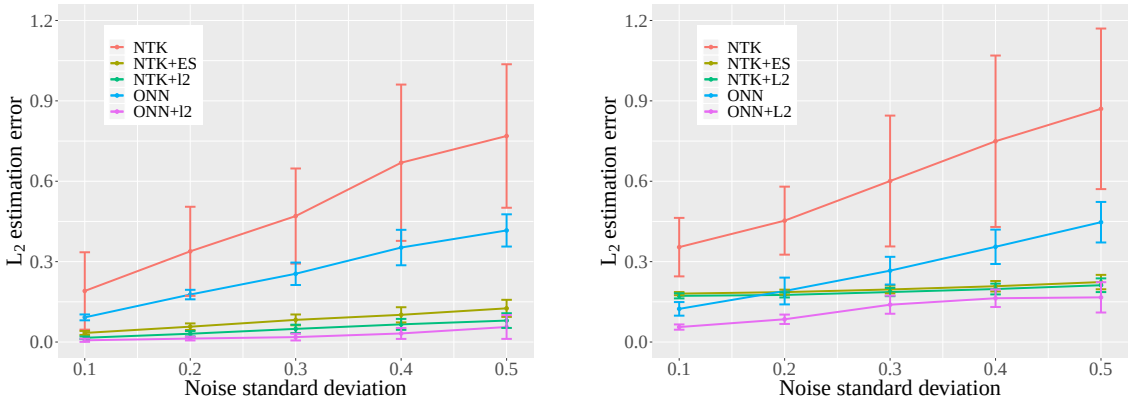
6 NUMERICAL STUDIES

In practice, regularization techniques are widely used in training deep learning models. Among others, Van Laarhoven [2017], Caruana et al. [2001], Prechelt [1998], Zhang et al. [2016], Lewkowycz and Gur-Ari [2020] have investigated the effectiveness of ℓ_2 regularization and early stopping in training DNNs, and comprehensive comparisons have been made empirically against other regularization techniques. Therefore, one major goal of this section is not to show state-of-the-art performance using ℓ_2 regularization, but to use it as an example to illustrate, from a nonparametric perspective, the necessity of regularization in training overparametrized neural networks with GD. Another goal is to demonstrate the robustness of our theory when some underlying assumptions are violated, e.g., one hidden layer, ReLU activation function and data on a sphere, etc.

Specifically, we consider NTK without regularization (NTK), NTK with early stopping[2] (NTK+ES), NTK with ℓ_2 regularization (NTK+ℓ_2), overparametrized neural network with and without ℓ_2 regularization, denoted as ONN and ONN+ℓ_2, respectively. For ONN, we use two-hidden-layer ReLU neural networks and $m = 500$ for each layer. To train the neural networks, instead of GD, we consider the more popular RMSProp

[1] Potential dependency of λ_0 on n is suppressed for notational simplicity.

[2] As specified in Theorem 4.2, the optimal stopping time k^* in (4.2) depends on σ, which is to be estimated from data. In our simulation, we directly use the true value. The GD algorithm can found in Appendix G



(a) f_1^* (b) f_2^*

Figure 1: The L_2 estimation errors are shown for all methods vs. σ, with their standard deviations plotted as vertical bars. Similarly for both f_1^* and f_2^*, we observe that NTK and ONN do not recover the true function well. Early stopping and ℓ_2 regularization perform similarly for NTK, especially for f_2^*. ONN+ℓ_2 performs the best in both cases.

optimizer [Hinton et al.] with the default setting. For ONN+ℓ_2 and NTK+ℓ_2, the tuning parameter μ is selected by cross-validation.

6.1 Simulated Data

Consider the $d = 2$ case where the training data points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ are i.i.d. sampled from unif$([-1,1]^2)$. We set $n = 100$ and let noises follow $N(0, \sigma^2)$. Two target functions are considered: $f_1^*(\boldsymbol{x}) = 0$ and $f_2^*(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{x}$. The L_2 estimation error is approximated using a noiseless test dataset $\{(\bar{\boldsymbol{x}}_i, f^*(\bar{\boldsymbol{x}}_i))\}_{i=1}^{1000}$ where $\bar{\boldsymbol{x}}_i$'s are new samples i.i.d. from unif$([-1,1]^2)$. We choose $\sigma = 0.1, 0.2, ..., 0.5$ and for each σ value, 100 replications are run to estimate the mean and standard deviation of the L_2 estimation error. Results are presented in Figure 1. More details and results can be found in Appendix G.

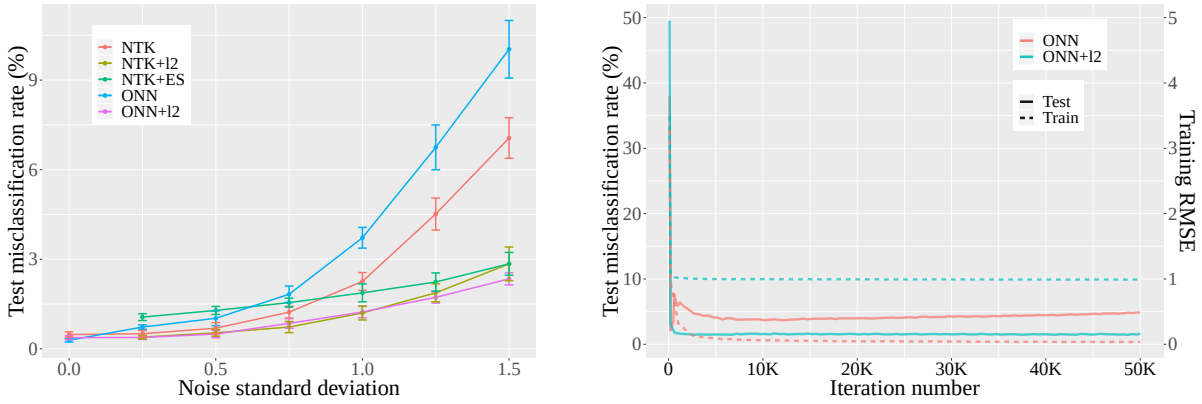
6.2 Real Data

To showcase our results on the L_2 estimation, an ideal dataset is one that can be well-fitted by neural networks so that we can treat it as noiseless and then manually inject random noises. Inspired by the numerical studies in Hu et al. [2020], we consider the MNIST dataset (digits 5 vs. 8 relabeled as -1 and 1), where the test accuracy can reach over 99% by shallow fully connected neural networks [LeCun et al., 1998]. Even though the dataset is for classification, we can treat the labels as continuous and learn the true function under the proposed regression setting. We use $\boldsymbol{y}^*$ to denote the true labels and manually add noises $\boldsymbol{\epsilon}$ to the training data, where each element of $\boldsymbol{\epsilon}$ follows $N(0, \sigma^2)$ independently. The perturbed labels are denoted by $\boldsymbol{y} = \boldsymbol{y}^* + \boldsymbol{\epsilon}$. By gradually increase σ, we investigate how ONN and ONN+ℓ_2 perform under the additive label noises setting.

Remark 3. (Additive label noises) To manually inject noises to classification data, many works consider replacing part of the labels by random labels [Zhang et al., 2016, Arora et al., 2019]. However, such noises are not i.i.d. and cannot be applied to the regression setting. Similar additive label noises are also considered in Hu et al. [2020].

The training dataset contains $n = 11272$ vectorized images of dimension $d = 784$. The test dataset size is 1866. For ONN+ℓ_2, our training objective function is Φ_1 as in (5.2) and setting $\mu = 0$ corresponds to the objective function of training ONN. On test dataset, which is *not contaminated* by noises, we use the sign of the output for classification and calculate the misclassification rate as a measure of estimation performance. To be more specific, a test image $\bar{\boldsymbol{x}}$ is classified as label 8 if $\widehat{f}(\bar{\boldsymbol{x}}) \geq 0$, and label 5 if $\widehat{f}(\bar{\boldsymbol{x}}) < 0$, where $\widehat{f}$ is the neural network estimator. The misclassification rate is the percentage of incorrect classifications on the test images. We choose $\sigma = 0, 0.25, ..., 1.5$ and for each σ value, 100 replications are run to estimate the mean and standard deviation of the test misclassification rate. How the training root mean square error (RMSE) and test misclassification rate evolve during training when $\sigma = 1$ for ONN and ONN+ℓ_2 is also investigated. The results are reported in Figure 2. More details and results can be found in Appendix G.

Remark 4. (NTK+ES) The performance of NTK+ES is shown in Figure 2(a). Unlike in the simulated dataset where NTK+ES and NTK+ℓ_2 perform almost iden-



(a) (b)

Figure 2: Figure (a) shows the test misclassification rates for all methods vs. σ with their standard deviations plotted as vertical bars. NTK+ES for $\sigma = 0$ is omitted since k^* is not well-defined when $\sigma = 0$ and NTK+ES in this case should be the same as NTK, i.e. $k^* = \infty$. As σ increases, all misclassification rates increase but NTK+ℓ_2 and ONN+ℓ_2 perform significantly better than NTK and ONN with smaller misclassification rate and better stability, i.e., the standard deviation is smaller. The NTK+ES is the green line and it performs the worst when $\sigma \leq 0.5$ but better than NTK and ONN when $\sigma \geq 1$. Figure (b) shows how the training RMSE and test misclassification rate evolve across iterations for ONN and ONN+ℓ_2 when $\sigma = 1$. For both methods, the training RMSEs decrease fast in the first 1K iterations. However, as the ONN training RMSE flattens after 10K iterations, its test misclassification rate goes up while that for ONN+ℓ_2 remains flat even after 50K iterations, which supports our conjecture in Remark 2. Figure (b) also reveals the potential early stopping time for ONN around iteration 10K, which has test misclassification rate comparable to that of ONN+ℓ_2.

tically, NTK+ES performs noticeably worst for the MNIST dataset, especially when σ is small. One possible explanation lies in our additive label noise setting. Even though we treat the labels as continuous during training, the reported misclassification rate only depends on the sign of the label. If σ is small, the probability of changing signs is small. This may be one of the reasons that NTK, ONN perform relatively well for small σ's, since if the signs remain the same, it is not very harmful to overfit the labels. Note that NTK+ℓ_2 and ONN+ℓ_2 choose small μ's such that it is not very different from NTK and ONN. The stopping rule in NTK+ES, on the other hand, doesn't take the classification setting into consideration and tends to underestimate the stopping time when the additive label noises are small. Nonetheless, we don't recommend NTK+ES for handling large datasets. Firstly, the noise level σ needs to be estimated, which brings extra instability to the algorithm. Secondly, NTK+ES is very computationally intensive, especially for the eigenvalues of the NTK matrix.

7 CONCLUSION AND DISCUSSION

From a nonparametric perspective, this paper studies overparametrized neural networks trained with GD and establishes optimal L_2 convergence rates for trained neural network estimators under the ℓ_2 regularization. On one hand, our result broadens the NTK literature by incorporating an explicit penalty term in the training objective. On the other hand, our convergence analysis extends the statistical theory of deep neural networks by bringing algorithmic guarantees into the network estimator and offsetting the extra complexity from overparametrization through delicate GD analysis. Our simulation results corroborate the theoretical analysis and imply that the assumptions of our theory may be relaxed. More investigations along this direction would advance our statistical understandings of deep learning. For example, our work can be further improved by relaxing the sphere assumption on the input data and the iteration number k imposed in Theorems 5.1 and 5.2. Additionally, although our theoretical analysis depends on the exact formula of the NTK associated with one-hidden layer ReLU neural network, it is possible to extend our theory to multi-layer DNNs as empirically shown in numerical experiments. In fact, it has been shown that the RKHS generated by the multi-layer NTK is equivalent to the one-hidden NTK [Chen and Xu, 2020]. Therefore, one possible approach for generalizing our theory is based on this equivalence.

The nonparametric perspective is potentially helpful in understanding other popular regularization techniques, e.g., batch normalization [Ioffe and Szegedy, 2015], data

augmentation [Dao et al., 2019], knowledge distillation [Hinton et al., 2015], etc. On the other hand, novel and problem-specific regularization approaches may be motivated during the convergence analysis that inspires better performance in practice.

References

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pages 2672–2680, 2014.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. *arXiv preprint arXiv:1701.07875*, 2017.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems*, pages 8571–8580, 2018.

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

Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*, pages 8157–8166, 2018.

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.

Difan Zou and Quanquan Gu. An improved analysis of training over-parameterized deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2053–2062, 2019.

Ziwei Ji and Matus Telgarsky. The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pages 1772–1798, 2019a.

Ziwei Ji and Matus Telgarsky. Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow ReLU networks. *arXiv preprint arXiv:1909.12292*, 2019b.

Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *arXiv preprint arXiv:1906.05890*, 2019.

Charles J Stone. Optimal global rates of convergence for nonparametric regression. *The Annals of Statistics*, pages 1040–1053, 1982.

Sidney Siegel. Nonparametric statistics. *The American Statistician*, 11(3):13–19, 1957.

Yuan Cao and Quanquan Gu. Generalization error bounds of gradient descent for learning overparameterized deep ReLU networks. *arXiv preprint arXiv:1902.01384*, 2019.

Colin Wei, Jason D Lee, Qiang Liu, and Tengyu Ma. Regularization matters: Generalization and optimization of neural nets vs their induced kernel. In *Advances in Neural Information Processing Systems*, pages 9709–9721, 2019.

W Hu, Z Li, and D Yu. Simple and effective regularization methods for training on noisily labeled data with generalization guarantee. In *International Conference on Learning Representations*, 2020.

Amnon Geifman, Abhay Yadav, Yoni Kasten, Meirav Galun, David Jacobs, and Ronen Basri. On the similarity between the laplace and neural tangent kernels. *NeurIPS 2020*, 2020.

Atsushi Nitanda, Geoffrey Chinot, and Taiji Suzuki. Gradient descent can learn less over-parameterized two-layer neural networks on classification problems. *arXiv preprint arXiv:1905.09870*, 2019.

Atsushi Nitanda and Taiji Suzuki. Optimal rates for averaged stochastic gradient descent under neural tangent kernel regime. *arXiv preprint arXiv:2006.12297*, 2020.

Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.

Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with ReLU activation function. *arXiv preprint arXiv:1708.06633*, 2017.

Benedikt Bauer, Michael Kohler, et al. On deep learning as a remedy for the curse of dimensionality in nonparametric regression. *The Annals of Statistics*, 47(4):2261–2285, 2019.

Ruiqi Liu, Ben Boukai, and Zuofeng Shang. Optimal nonparametric inference via deep neural network. *arXiv preprint arXiv:1902.01687*, 2019.

Masaaki Imaizumi and Kenji Fukumizu. Deep neural networks learn non-smooth functions effectively. *arXiv preprint arXiv:1802.04474*, 2018.

Alberto Bietti and Julien Mairal. On the inductive bias of neural tangent kernels. In *Advances in Neural Information Processing Systems*, pages 12873–12884, 2019.

Ming Yuan, Ding-Xuan Zhou, et al. Minimax optimal rates of estimation in high dimensional additive models. *The Annals of Statistics*, 44(6):2564–2593, 2016.

Garvesh Raskutti, Martin J Wainwright, and Bin Yu. Early stopping and non-parametric regression: an optimal data-dependent stopping rule. *The Journal of Machine Learning Research*, 15(1):335–366, 2014.

Michael Kohler and Adam Krzyzak. Overparametrized deep neural networks do not generalize well. *arXiv preprint arXiv:1912.03925*, 2019.

Anders Krogh and John A Hertz. A simple weight decay can improve generalization. In *Advances in Neural Information Processing Systems*, pages 950–957, 1992.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1): 1929–1958, 2014.

Berkin Bilgic, Itthi Chatnuntawech, Audrey P Fan, Kawin Setsompop, Stephen F Cauley, Lawrence L Wald, and Elfar Adalsteinsson. Fast image reconstruction with l2-regularization. *Journal of magnetic resonance imaging*, 40(1):181–191, 2014.

Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017.

Ekachai Phaisangittisagul. An analysis of the regularization between l2 and dropout in single hidden layer neural network. In *2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)*, pages 174–179. IEEE, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Rich Caruana, Steve Lawrence, and C Lee Giles. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In *Advances in Neural Information Processing Systems*, pages 402–408, 2001.

Lutz Prechelt. Early stopping-but when? In *Neural Networks: Tricks of the Trade*, pages 55–69. Springer, 1998.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Aitor Lewkowycz and Guy Gur-Ari. On the training dynamics of deep networks with l_2 regularization. *arXiv preprint arXiv:2006.08643*, 2020.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86 (11):2278–2324, 1998.

Lin Chen and Sheng Xu. Deep neural tangent kernel and laplace kernel have the same rkhs. *arXiv preprint arXiv:2009.10683*, 2020.

Tri Dao, Albert Gu, Alexander J Ratner, Virginia Smith, Christopher De Sa, and Christopher Ré. A kernel theory of modern data augmentation. *Proceedings of machine learning research*, 97:1528, 2019.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Yuan Cao, Zhiying Fang, Yue Wu, Ding-Xuan Zhou, and Quanquan Gu. Towards understanding the spectral bias of deep learning. *arXiv preprint arXiv:1912.01198*, 2019.

Sara van de Geer. *Empirical Processes in M-estimation*. Cambridge University Press, 2000.

Sara van de Geer. On the uniform convergence of empirical norms and inner products, with application to causal inference. *Electronic Journal of Statistics*, 8 (1):543–574, 2014.

George Kimeldorf and Grace Wahba. Some results on tchebycheffian spline functions. *Journal of mathematical analysis and applications*, 33(1):82–95, 1971.

Richard S Varga. *Gershgorin and His Circles*, volume 36. Springer Science & Business Media, 2010.

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

Kendall Atkinson and Weimin Han. *Spherical Harmonics and Approximations on the Unit Sphere: An Introduction*, volume 2044. Springer Science & Business Media, 2012.

Efthimiou Costas and Frye Christopher. *Spherical Harmonics in p Dimensions*. World Scientific, 2014.

Johann S Brauchart and Josef Dick. A characterization of Sobolev spaces on the sphere and an extension of Stolarsky's invariance principle to arbitrary smoothness. *Constructive Approximation*, 38(3):397–445, 2013.

He Ping Wang, Kai Wang, and Jing Wang. Entropy numbers of Besov classes of generalized smoothness on the sphere. *Acta Mathematica Sinica, English Series*, 30(1):51–60, 2014.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.