Learning One-hidden-layer ReLU Networks via Gradient Descent

Xiao Zhang* University of Virginia

Yaodong Yu* University of Virginia

Lingxiao Wang* University of California, Los Angeles

Quanquan Gu University of California, Los Angeles

Abstract

We study the problem of learning one-hidden-layer neural networks with Rectified Linear Unit (ReLU) activation function, where the inputs are sampled from standard Gaussian distribution and the outputs are generated from a noisy teacher network. We analyze the performance of gradient descent for training such kind of neural networks based on empirical risk minimization, and provide algorithm-dependent guarantees. In particular, we prove that tensor initialization followed by gradient descent can converge to the ground-truth parameters at a linear rate up to some statistical error. To the best of our knowledge, this is the first work characterizing the recovery guarantee for practical learning of one-hidden-layer ReLU networks with multiple neurons. Numerical experiments verify our theoretical findings.

1 INTRODUCTION

Deep neural networks have achieved lots of breakthroughs in the field of artificial intelligence, such as speech recognition (Hinton et al., 2012), image processing (Krizhevsky et al., 2012), statistical machine translation (Bahdanau et al., 2014), and Go games (Silver et al., 2016). The empirical success of neural networks stimulates numerous theoretical studies in this field. For example, in order to explain the superiority of neural networks, a series of work (Hornik, 1991; Barron, 1993; Daniely et al., 2016; Cohen et al., 2016; Arora et al., 2016; Mukherjee and Basu, 2017; Hanin, 2017; Hanin and Sellke, 2017; Yarotsky, 2017, 2018) investigated the expressive power of neural networks. It has been proved that given appropriate weights, neural networks with nonlinear activation function can approximate any continuous function.

In practice, (stochastic) gradient descent remains one of the most widely-used approaches for deep learning. However, due to the nonconvexity and nonsmoothness of the loss function landscape, existing theory in optimization cannot explain why gradient-based methods can effectively learn neural networks. To bridge this gap, a line of research (Tian, 2017; Li and Yuan, 2017; Du et al., 2017a,b) studied (stochastic) gradient descent for learning shallow neural networks from a theoretical perspective. More specifically, by assuming an underlying teacher network, they established recovery guarantees for applying gradient-based learning algorithms to the population loss function (a.k.a., expected risk function). Another line of research (Zhong et al., 2017; Soltanolkotabi, 2017; Soltanolkotabi et al., 2017; Fu et al., 2018; Ge et al., 2019) investigated using (stochastic) gradient descent to minimize the empirical loss function of shallow neural networks, and provided theoretical guarantees on sample complexity, i.e., number of samples required for recovery.

Our work follows the second line of research, where we directly study the empirical risk minimization of one-hidden-layer ReLU networks, and characterize the recovery guarantee using gradient descent. More specifically, we assume the inputs $\{\mathbf{x}_i\}_{i=1}^N \subseteq \mathbb{R}^d$ follow standard multivariate Gaussian distribution, and the outputs $\{y_i\}_{i=1}^N \subseteq \mathbb{R}$ are generated from the following one-hidden-layer ReLU-based teacher network (see Figure 1 for graphical illustration)

$$y_i = \sum_{j=1}^{K} \sigma(\mathbf{w}_j^{*\top} \mathbf{x}_i) + \epsilon_i, \quad \text{for any } i \in [N]. \tag{1.1}$$

Here, $\mathbf{w}_j^* \in \mathbb{R}^d$ is the weight parameter of the j-th neuron, $\sigma(x) = \max\{x, 0\}$ is the ReLU activation function, and $\{\epsilon_i\}_{i=1}^N$ are i.i.d. zero mean sub-Gaussian

*Equal Contribution

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89. Copyright 2019 by the author(s).

random noises[1] with sub-Gaussian norm $\nu > 0$.

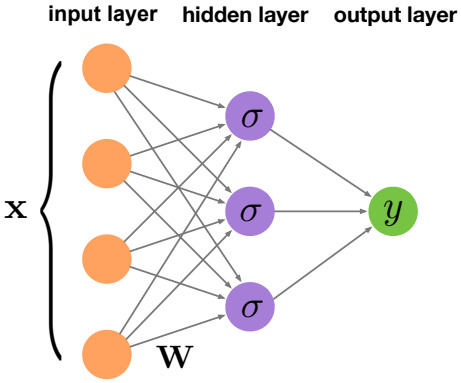


Figure 1: Illustration of one-hidden-layer ReLU-based teacher network (1.1).

Our goal is to recover the unknown parameter matrix $\mathbf{W}^* = [\mathbf{w}_1^*, \ldots, \mathbf{w}_K^*] \in \mathbb{R}^{d\times K}$ based on the observed N examples $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$. The recovery problem can be equivalently formulated as the following empirical risk minimization problem using square loss

$$\min_{\mathbf{W}\in\mathbb{R}^{d\times K}} \widehat{\mathcal{L}}_N(\mathbf{W}) = \frac{1}{2N}\sum_{i=1}^{N}\bigg(\sum_{j=1}^{K}\sigma(\mathbf{w}_j^\top \mathbf{x}_i) - y_i\bigg)^2, \tag{1.2}$$

where $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_K]$. In this paper, we show that with good starting point and sample complexity linear in d, using gradient descent to solve (1.2) is guaranteed to converge to $\mathbf{W}^*$ at a linear rate. To the best of our knowledge, this is the first result of its kind to prove the theoretical guarantee for learning one-hidden-layer ReLU networks with multiple neurons based on the empirical loss function. We believe our analysis on one-hidden-layer ReLU networks can shed light on the understanding of gradient-based methods for learning deeper neural networks. The main contributions of this work are summarized as follows:

- We consider the empirical risk minimization problem (1.2) for learning one-hidden-layer ReLU networks. Compared with existing studies (Tian, 2017; Li and Yuan, 2017; Du et al., 2017a,b) that consider the ideal population risk minimization, our analysis is more aligned with the practice of deep learning that is based on the empirical loss function. More specifically, the empirical optimization problem in (1.2) is nonconvex and nonsmooth (ReLU-activation), which has not been studied in previous work.
- We analyze the performance of gradient descent based algorithm for minimizing the empirical loss function. We demonstrate that, provided an appropriate initial solution, gradient descent can linearly converge to the ground-truth parameters of the underlying teacher network (1.1) up to some statistical error. In particular, the statistical error term depends on the sample size N, the input dimension d, the number of neurons in the hidden layer K, as well as the magnitude of the noise distribution ν. In addition, we show that the sample complexity for recovery required by our algorithm is linear in d up to a logarithmic factor.
- We provide a uniform convergence bound on the gradient of the empirical loss function (1.2). More specifically, we characterize the difference between the gradient of the empirical loss function and the gradient of the population loss function, when the parameters are close to the ground-truth parameters. This result enables us to establish the linear convergence guarantee of gradient descent method without using resampling (i.e., sample splitting) trick adopted in Zhong et al. (2017).

The remainder of this paper is organized as follows: In Section 2, we discuss the most related literature to our work. We introduce the problem setup and our proposed algorithm in Section 3. We present the main theoretical results and their proof in Sections 4 and 5 respectively. In Section 6, we conduct experiments to verify our theory. Finally, we conclude our paper and discuss some future work in Section 7.

Notation. We use $[d]$ to denote the set $\{1, 2, \ldots, d\}$. For any d-dimensional vector $\mathbf{x} = [x_1, ..., x_d]^\top$, let $\|\mathbf{x}\|_2 = (\sum_{i=1}^d |x_i|^2)^{1/2}$ be its ℓ_2 norm. For any matrix $\mathbf{A} = [A_{ij}]$, denote the spectral norm and Frobenius norm of $\mathbf{A}$ by $\|\mathbf{A}\|_2$ and $\|\mathbf{A}\|_F$, respectively. Let $\sigma_{\max}(\mathbf{A})$, $\sigma_{\min}(\mathbf{A})$ be the largest singular value and smallest singular value of $\mathbf{A}$, respectively. Given any two sequences $\{a_n\}$ and $\{b_n\}$, we write $a_n = O(b_n)$ if there exists a constant $0 < C < +\infty$ such that $a_n \le C\, b_n$, and we use $\widetilde{O}(\cdot)$ to hide the logarithmic factors. We use $\mathbb{1}\{\mathcal{E}\}$ to denote the indicator function such that $\mathbb{1}\{\mathcal{E}\} = 1$ if the event $\mathcal{E}$ is true, otherwise $\mathbb{1}\{\mathcal{E}\} = 0$. For two matrices $\mathbf{A}, \mathbf{B}$, we say $\mathbf{A} \succeq \mathbf{B}$ if $\mathbf{A} - \mathbf{B}$ is positive semidefinite. We use $\mathcal{B}_r(\mathbf{B}) = \{\mathbf{A} \in \mathbb{R}^{d\times K} : \|\mathbf{A} - \mathbf{B}\|_F \le r\}$ to denote the Frobenius norm ball centering at $\mathbf{B}$ with radius r.

2 RELATED WORK

To better understand the extraordinary performance of neural networks on different tasks, a line of research (Hornik, 1991; Montufar et al., 2014; Cohen et al., 2016; Telgarsky, 2016; Raghu et al., 2016; Poole et al., 2016; Arora et al., 2016; Daniely et al., 2016; Pan and

[1]The formal definitions of sub-Gaussian random variable and sub-Gaussian norm can be found in Section 4.

Srikumar, 2016; Zhang et al., 2016; Lu et al., 2017) has studied the expressive power of neural networks. In particular, Hornik (1991) showed that, with sufficient number of neurons, shallow networks can approximate any continuous function. Cohen et al. (2016); Telgarsky (2016) proved that a shallow network requires exponential size to realize functions that can be implemented by a deep network of polynomial size. Raghu et al. (2016); Poole et al. (2016); Arora et al. (2016) characterized the exponential dependence on the depth of the network based on different measures of expressivity. Recently, Zhang et al. (2016) empirically demonstrated that neural networks can actually memorize the training samples but still generalize well. Daniely et al. (2016); Lu et al. (2017) showed how the depth and width can affect the expressive power of neural networks.

However, the expressive power of neural networks can only partially explain the empirical success of deep learning. From a theoretical perspective, it is well-known that learning neural networks in general settings is hard in the worst case (Blum and Rivest, 1989; Auer et al., 1996; Livni et al., 2014; Shamir, 2016; Shalev-Shwartz et al., 2017a,b; Zhang et al., 2017; Ge et al., 2017). Nevertheless, a vast literature (Kalai and Sastry, 2009; Kakade et al., 2011; Sedghi and Anandkumar, 2014; Janzamin et al., 2015; Zhang et al., 2015; Goel et al., 2016; Arora et al., 2016) developed ad hoc algorithms that can learn neural networks with provable guarantees. However, none of these algorithms is gradient-based method, which is the most widely-used optimization algorithm for deep learning in practice.

Recently, a series of work (Tian, 2017; Brutzkus and Globerson, 2017; Li and Yuan, 2017; Du et al., 2017a,b) studied the recovery guarantee of gradient-based methods for learning shallow neural networks based on population loss function (i.e., expected risk function). More specifically, Tian (2017) proved that for one-layer one-neuron ReLU networks (i.e., ReLU unit), randomly initialized gradient descent on the population loss function can recover the groundtruth parameters of the teacher network. In a concurrent work, Brutzkus and Globerson (2017) considered the problem of learning a convolution filter and showed that gradient descent enables exact recovery of the true parameters, provided the filters are non-overlapping. Later on, Li and Yuan (2017) studied one-hidden-layer residual networks, and proved that stochastic gradient descent can recover the underlying true parameters in polynomial number of iterations. Du et al. (2017a) studied the convergence of gradient-based methods for learning a convolutional filter. They showed that under certain conditions, performing (stochastic) gradient descent on the expected risk function can recover the underlying true parameters in polynomial time. Du et al. (2017b) further studied the problem of learning the one-hidden-layer ReLU based convolutional neural network in the no-overlap patch setting. More specifically, they established the convergence guarantee of gradient descent with respect to the expected risk function when the input follows Gaussian distribution. Nevertheless, all these studies are based on the population loss function.

In practice, training neural networks is based on the empirical loss function. To the best of our knowledge, only several recent studies (Zhong et al., 2017; Soltanolkotabi, 2017; Soltanolkotabi et al., 2017; Fu et al., 2018) analyzed gradient based methods for training neural networks using empirical risk minimization. More specifically, under condition that the activation function is smooth, Zhong et al. (2017); Soltanolkotabi et al. (2017); Fu et al. (2018) established a locally linear convergence rate for gradient descent with suitable initialization scheme. However, none of their analyses are applicable to ReLU networks since ReLU activation function is nonsmooth[2]. Soltanolkotabi (2017) analyzed the projected gradient descent on the empirical loss function for one-neuron ReLU networks (i.e., ReLU unit). Yet the analysis requires a projection step to ensure convergence, while the constraint set of the projection depends on the unknown ground-truth weight vector, which makes their algorithm less practical. Our work also follows this line of research, where we investigate the theoretical performance of gradient descent for learning one-hidden-layer ReLU networks with multiple neurons.

Inspired by the success of first-order optimization algorithms (Ge et al., 2015; Jin et al., 2017) for solving non-convex optimization problems efficiently, some recent work (Choromanska et al., 2015; Safran and Shamir, 2016; Mei et al., 2016; Kawaguchi, 2016; Hardt and Ma, 2016; Soltanolkotabi et al., 2017; Soudry and Carmon, 2016; Xie et al., 2017; Nguyen and Hein, 2017; Ge et al., 2017; Safran and Shamir, 2017; Yun et al., 2017; Du and Lee, 2018; Gao et al., 2018) attempted to understand neural networks by characterizing their optimization landscape. Choromanska et al. (2015) studied the loss surface of a special random neural network. Safran and Shamir (2016) analyzed the geometric structure of the over-parameterized neural networks. Mei et al. (2016) studied the landscape of the empirical loss of the one-layer neural network given that the third derivative of the activation function is bounded. Kawaguchi (2016) showed that there is

[2]While many activation functions including ReLU are discussed in Zhong et al. (2017), their locally linear convergence result for gradient descent is not applicable to ReLU activation function.

no spurious local minimum for linear deep networks. Hardt and Ma (2016) proved that linear residual networks have no spurious local optimum. Soudry and Carmon (2016); Xie et al. (2017); Nguyen and Hein (2017); Yun et al. (2017) also showed that there is no spurious local minimum for some other neural networks under stringent assumptions. Soltanolkotabi et al. (2017) studied the global optimality of the over-parameterized network with quadratic activation functions. On the other hand, Ge et al. (2017); Safran and Shamir (2017) showed that ReLU neural networks with multiple neurons using square loss actually have spurious local minima. To address this issue, Ge et al. (2017) proposed to modify the objective function of ReLU networks, and showed that the modified objective function has no spurious local minimum, thus perturbed gradient descent can learn the groundtruth parameters. Compared with Ge et al. (2017), we directly analyze the objective function of ReLU networks based on square loss using gradient descent without modifying the loss function, but needing a special initialization.

It is worth noting that very recently there is a line of research that studies the global convergence of gradient descent and stochastic gradient descent for training (deep) neural networks (Li and Liang, 2018; Du et al., 2018, 2019; Allen-Zhu et al., 2018b; Zou et al., 2018), and their generalization performance (Li and Liang, 2018; Allen-Zhu et al., 2018a; Cao and Gu, 2019), in the over-parameterized regime where the width of the neural network is much larger than the training set size. We would like to point out that these results are not directly comparable with our result, because we considered the setting where the training set size is larger than the number of neural network parameters.

3 PROBLEM SETUP AND ALGORITHM

In this section, we present the problem formulation along with a gradient descent-based algorithm for learning one-hidden-layer ReLU networks. Recall that our goal is to recover the unknown parameter matrix $\mathbf{W}^*$ based on the empirical loss function in (1.2). For the ease of later analysis, we define the corresponding population loss function as follows

$$\mathcal{L}(\mathbf{W}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{X}\sim\mathcal{D}_X}\Big(\sum_{j=1}^{K}\sigma(\mathbf{w}_j^\top \boldsymbol{X}) - \sum_{j=1}^{K}\sigma(\mathbf{w}_j^{*\top}\boldsymbol{X})\Big)^2, \tag{3.1}$$

where $\mathcal{D}_X = N(\mathbf{0}, \mathbf{I})$ denotes the standard multivariate Gaussian distribution. In addition, let $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_K > 0$ be the sorted singular values of $\mathbf{W}^*$, and $\kappa = \sigma_1/\sigma_K$ be the condition number of $\mathbf{W}^*$, and $\lambda = (\Pi_{j=1}^K \sigma_j)/\sigma_K^K$.

In this work, we focus on minimizing the empirical loss function in (1.2) instead of the population loss function in (3.1), because in practice one can only get access to the training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^N$. Witnessing the empirical success of the widely-used gradient-based methods for training neural networks, one natural question is whether gradient descent can recover $\mathbf{W}^*$ based on the empirical loss function in (1.2). In later analysis, we will show that the answer to the above question is affirmative. The gradient descent algorithm for solving the nonconvex and nonsmooth optimization problem (1.2) is demonstrated in Algorithm 1.

Algorithm 1 Gradient Descent

Require: empirical loss function $\widehat{\mathcal{L}}_N$; step size η; iteration number T; initial estimator $\mathbf{W}^0$.
  for $t = 1, 2, 3, \ldots, T$ **do**
    $\mathbf{W}^t = \mathbf{W}^{t-1} - \eta\nabla\widehat{\mathcal{L}}_N(\mathbf{W}^{t-1})$
  end for
Ensure: $\mathbf{W}^T$

It is worth noting that the gradient descent algorithm shown in Algorithm 1 does not require any resampling (a.k.a., sample splitting) procedure (Jain et al., 2013) compared with the gradient descent algorithm analyzed in Zhong et al. (2017). More specifically, the gradient descent algorithm in Zhong et al. (2017) requires a fresh subset of the whole training sample at each iteration in order to establish the convergence guarantee. In sharp contrast, Algorithm 1 analyzed in this paper does not need resampling. The reason is that we are able to establish a uniform convergence bound between the gradient of the empirical loss function and the gradient of the population loss function, as will be illustrated in the next section. Furthermore, we lay out the explicit form of the derivative of $\widehat{\mathcal{L}}_N(\mathbf{W})$ with respect to $\mathbf{w}_k$ as follows

$$\Big[\nabla\widehat{\mathcal{L}}_N(\mathbf{W})\Big]_k = \sum_{j=1}^{K}\Big(\widehat{\boldsymbol{\Sigma}}(\mathbf{w}_j, \mathbf{w}_k)\mathbf{w}_j - \widehat{\boldsymbol{\Sigma}}(\mathbf{w}_j^*, \mathbf{w}_k)\mathbf{w}_j^*\Big) - \frac{1}{N}\sum_{i=1}^{N}\epsilon_i\mathbf{x}_i \cdot \mathbb{1}\{\mathbf{w}_k^\top\mathbf{x}_i \geq 0\}, \tag{3.2}$$

where $\widehat{\boldsymbol{\Sigma}}(\mathbf{w}_j, \mathbf{w}_k)$ and $\widehat{\boldsymbol{\Sigma}}(\mathbf{w}_j^*, \mathbf{w}_k)$ are defined as

$$\widehat{\boldsymbol{\Sigma}}(\mathbf{w}_j, \mathbf{w}_k) = \frac{1}{N}\sum_{i=1}^{N}\Big[\mathbf{x}_i\mathbf{x}_i^\top \cdot \mathbb{1}\{\mathbf{w}_j^\top\mathbf{x}_i \geq 0, \mathbf{w}_k^\top\mathbf{x}_i \geq 0\}\Big],$$
$$\widehat{\boldsymbol{\Sigma}}(\mathbf{w}_j^*, \mathbf{w}_k) = \frac{1}{N}\sum_{i=1}^{N}\Big[\mathbf{x}_i\mathbf{x}_i^\top \cdot \mathbb{1}\{\mathbf{w}_j^{*\top}\mathbf{x}_i \geq 0, \mathbf{w}_k^\top\mathbf{x}_i \geq 0\}\Big]. \tag{3.3}$$

4 MAIN THEORY

In this section, we present our main theoretical results, including the local convergence result and the initialization result. Before presenting our main theoretical results, we first lay out the definitions of sub-Gaussian random variable and sub-Gaussian norm.

Definition 4.1. (sub-Gaussian random variable) We say X is a sub-Gaussian random variable with sub-Gaussian norm $K > 0$, if $(\mathbb{E}|X|^p)^{1/p} \leq K\sqrt{p}$ for all $p \geq 1$. In addition, the sub-Gaussian norm of X, denoted $\|X\|_{\psi_2}$, is defined as $\|X\|_{\psi_2} = \sup_{p\geq 1} p^{-1/2}(\mathbb{E}|X|^p)^{1/p}$.

The following theorem shows that as long as the initial estimator $\mathbf{W}^0$ falls in a small neighbourhood of $\mathbf{W}^*$, gradient descent algorithm in Algorithm 1 is guaranteed to converge to $\mathbf{W}^*$ with a linear rate of convergence.

Theorem 4.2. Assume the inputs $\{\mathbf{x}_i\}_{i=1}^N$ are sampled from standard Gaussian distribution, and the outputs $\{y_i\}_{i=1}^N$ are generated from the teacher network (1.1). Suppose the initial estimator $\mathbf{W}^0$ satisfies $\|\mathbf{W}^0 - \mathbf{W}^*\|_F \leq c\sigma_K/(\lambda\kappa^3 K^2)$, where $c > 0$ is a small enough absolute constant. Then there exist absolute constants c_1, c_2, c_3, c_4 and c_5 such that provide the sample size satisfies

$$N \geq \frac{c_1\lambda^4\kappa^{10}K^9 d}{\sigma_K^2}\log\left(\frac{\lambda\kappa K d}{\sigma_K}\right)\cdot(\|\mathbf{W}^*\|_F^2 + \nu^2),$$

the output of Algorithm 1 with step size $\eta \leq 1/(c_2\kappa K^2)$ satisfies

$$\|\mathbf{W}^T - \mathbf{W}^*\|_F^2 \leq \left(1 - \frac{c_3\eta}{\lambda\kappa^2}\right)^T \|\mathbf{W}^0 - \mathbf{W}^*\|_F^2 + \frac{c_4\lambda^2\kappa^4 K^5 d\log N}{N}\cdot(\|\mathbf{W}^*\|_F^2 + \nu^2) \tag{4.1}$$

with probability at least $1 - c_5/d^{10}$.

Remark 4.3. Theorem 4.2 suggests that provided that the initial solution $\mathbf{W}_0$ is sufficiently close to $\mathbf{W}^*$, the output of Algorithm 1 exhibits a linear convergence towards $\mathbf{W}^*$, up to some statistical error. More specifically, the estimation error is bounded by two terms (see the right hand side of (4.1)): the first term is the optimization error, and the second term represents the statistical error. The statistical error depends on the sample size N, the input dimension d, the number of neurons in the hidden layer K and some other problem-specific parameters.

Remark 4.4. In addition, due to the existence of statistical error, we are only able to achieve at best

$$\varepsilon = c\lambda^2\kappa^4 K^5(\|\mathbf{W}^*\|_F^2 + \nu^2)\cdot\frac{d\log N}{N}$$

estimation error, where c is an absolute constant. Because of the linear convergence rate, it is sufficient to perform $T = O\big(\lambda\kappa^3 K^2\cdot\log(1/\varepsilon)\big)$ number of iterations in Algorithm 1 to make sure the optimization error is less than ε. Putting these pieces together gives the overall sample complexity of Algorithm 1 to achieve ε-estimation error:

$$O\bigg(C\cdot d\log\left(\frac{\lambda\kappa K d}{\sigma_K}\right)\log\left(\frac{1}{\varepsilon}\right)\bigg),$$

where $C = \mathrm{poly}\big(\lambda, \kappa, K, \sigma_K, \nu, \|\mathbf{W}^*\|_F\big)$. Apparently, it is in the order of $\widetilde{O}\big(\mathrm{poly}(K)\cdot d\big)$ if we treat other problem-specific parameters as constants. Correspondingly, the statistical error is in the order of $\widetilde{O}\big(\mathrm{poly}(K)\cdot d/N\big)$.

The remaining question is how to find a good initial solution $\mathbf{W}^0$ for Algorithm 1, which satisfies the assumption of Theorem 4.2. We propose to use tensor initialization, which is proposed by Zhong et al. (2017). Here we briefly introduce the procedure of tensor initialization. The basic idea of tensor initialization is to obtain an estimator $\mathbf{W}^0$ that has the same column space as the ground-truth parameter $\mathbf{W}^*$ of the teacher network. In detail, it first constructs two matrices $\mathbf{P}_1 = C\mathbf{I} + \mathbf{P}$ and $\mathbf{P}_2 = C\mathbf{I} - \mathbf{P}$, where $\mathbf{P} = \sum_{i=1}^N y_i(\mathbf{x}_i\mathbf{x}_i^\top - \mathbf{I})$, $C \geq 2\mathbb{E}\|\mathbf{P}\|_2$ and the expectation is taken over the randomness of the input data. Given $\mathbf{P}_1$ and $\mathbf{P}_2$, it then estimates their top-K eigenvalues in terms of magnitude and corresponding eigenvectors. Next, it combines these $2K$ eigenvectors and select K eigenvectors with top K eigenvalues. Finally, it performs an orthogonalization procedure to get the desired initial estimator $\mathbf{W}^0$. The following lemma, proved in Zhong et al. (2017), shows that tensor initialization can give us desired initial estimators.

Lemma 4.5. (Zhong et al., 2017) Consider the empirical risk minimization in (1.2), if the sample size $N \geq \epsilon^{-2}\cdot d\cdot\mathrm{poly}(\kappa, K, \log d)$, with probability at least $1 - d^{-10}$, the output $\mathbf{W}^0 \in \mathbb{R}^{d\times K}$ of the tensor initialization satisfies

$$\|\mathbf{W}^0 - \mathbf{W}^*\|_F \leq \epsilon\cdot\mathrm{poly}(\kappa, K)\|\mathbf{W}^*\|_F.$$

Remark 4.6. According to Lemma 4.5, if we set the approximation error ϵ such that

$$\epsilon \leq \frac{c_1\sigma_K}{\lambda\kappa^3 K^2\mathrm{poly}(\kappa, K)\|\mathbf{W}^*\|_F},$$

where c_1 is an absolute constant, the initial estimator $\mathbf{W}^0$ satisfies the assumption of Theorem 4.2. The corresponding sample complexity requirement for tensor initialization is in the order of $O\big(\mathrm{poly}(\lambda, \kappa, K, \|\mathbf{W}^*\|_F, \log d)\cdot d\big)$. Therefore, combining Lemma 4.5 with Theorem 4.2, we conclude

that tensor initialization followed by gradient descent can learn one-hidden-layer ReLU networks with linear convergence rate and overall sample complexity $\widetilde{O}\big(\text{poly}(K)\cdot d\big)$.

5 PROOF OF THE MAIN THEORY

In this section, we lay out the proof of our main result. To prove Theorem 4.2, we need to make use of the following lemmas. The first lemma characterizes the local strong convexity of the population loss function around $\mathbf{W}^*$.

Lemma 5.1. For any $\mathbf{W}\in\mathbb{R}^{d\times K}$ such that $\|\mathbf{W}-\mathbf{W}^*\|_F\le c\sigma_K/(\lambda\kappa^3K^2)$, the Hessian of the population loss function $\mathcal{L}(\mathbf{W})$ satisfies

$$\nabla^2\mathcal{L}(\mathbf{W})\succeq\mu\mathbf{I},$$

where $\mu=c/(\lambda\kappa^2)$ and $c>0$ is an absolute constant.

Next lemma characterizes the local strong smoothness of the population loss function around $\mathbf{W}^*$.

Lemma 5.2. The gradient of the population loss function $\nabla\mathcal{L}(\cdot)$ is L-Lipschitz within the region $\Omega=\{\mathbf{W}\in\mathbb{R}^{d\times K}\mid\|\mathbf{W}-\mathbf{W}^*\|_F\le\sigma_K/2\}$, i.e., for any $\mathbf{W}_1,\mathbf{W}_2\in\Omega$

$$\|\nabla\mathcal{L}(\mathbf{W}_1)-\nabla\mathcal{L}(\mathbf{W}_2)\|_F\le L\|\mathbf{W}_1-\mathbf{W}_2\|_F,$$

where $L=c\kappa K^2$, and $c>0$ is an absolute constant.

Lemma 5.3. Consider the empirical loss function $\widehat{\mathcal{L}}_N(\mathbf{W})$ in (1.2). For all $\mathbf{W}\in\mathbb{R}^{d\times K}$ such that $\|\mathbf{W}-\mathbf{W}^*\|_F\le c\sigma_K/(\lambda\kappa^3K^2)$, where c is an absolute constant, there exists absolute constants c_1,c_2 such that with probability at least $1-c_1/d^{10}$, we have

$$\|\nabla\widehat{\mathcal{L}}_N(\mathbf{W})-\nabla\mathcal{L}(\mathbf{W})\|_F\le c_2\sqrt{\frac{dK^5\log N}{N}}(\|\mathbf{W}^*\|_F+\nu),$$

where ν is the sub-Gaussian norm of the additive noise in the teacher network.

Lemma 5.3 provides a uniform convergence bound on the difference between the gradient of the empirical loss function and the gradient of the population loss function in terms of Frobenius norm.

Proofs of the above three lemmas can be found in the appendix. Based on these three lemmas, we are ready to prove the main theorem.

Proof of Theorem 4.2. We prove it by induction. We make the following inductive hypothesis

$$\|\mathbf{W}^t-\mathbf{W}^*\|_F\le\frac{c\sigma_K}{\lambda\kappa^3K^2}. \tag{5.1}$$

Note that based on the assumption of Theorem 4.2, the initial estimator $\mathbf{W}^0$ satisfies (5.1), thus it remains to prove the inductive step. In other words, we need to show that $\mathbf{W}^{t+1}$ satisfies (5.1), provided that (5.1) holds for $\mathbf{W}^t$. Consider the gradient-based Algorithm 1 at the $(t+1)$-th iteration, we have

$$\mathbf{W}^{t+1}=\mathbf{W}^t-\eta\nabla\widehat{\mathcal{L}}_N(\mathbf{W}^t),$$

which implies that

$$\begin{aligned}\|\mathbf{W}^{t+1}-\mathbf{W}^*\|_F^2=\|\mathbf{W}^t-\mathbf{W}^*\|_F^2+\eta^2\|\nabla\widehat{\mathcal{L}}_N(\mathbf{W}^t)\|_F^2\\-2\eta\langle\nabla\widehat{\mathcal{L}}_N(\mathbf{W}^t),\mathbf{W}^t-\mathbf{W}^*\rangle.\end{aligned}$$

Therefore, by adding the term $\nabla\mathcal{L}(\mathbf{W}^t)$ into the above inequality, we can obtain

$$\begin{aligned}&\|\mathbf{W}^{t+1}-\mathbf{W}^*\|_F^2-\|\mathbf{W}^t-\mathbf{W}^*\|_F^2\\&\le\underbrace{-2\eta\langle\nabla\mathcal{L}(\mathbf{W}^t),\mathbf{W}^t-\mathbf{W}^*\rangle+2\eta^2\|\nabla\mathcal{L}(\mathbf{W}^t)\|_F^2}_{I_1}\\&\underbrace{-2\eta\langle\nabla\widehat{\mathcal{L}}_N(\mathbf{W}^t)-\nabla\mathcal{L}(\mathbf{W}^t),\mathbf{W}^t-\mathbf{W}^*\rangle}_{I_2}\\&\underbrace{+2\eta^2\|\nabla\widehat{\mathcal{L}}_N(\mathbf{W}^t)-\nabla\mathcal{L}(\mathbf{W}^t)\|_F^2}_{I_2},\end{aligned}$$

where the inequality follows from $(a-b)^2\le2a^2+2b^2$.

In the following discussions, we are going to bound the terms I_1 and I_2, respectively. Consider the first term I_1. Note that according to the population loss function (3.1), we have $\nabla\mathcal{L}(\mathbf{W}^*)=\mathbf{0}$. Thus, we have

$$\text{vec}\big(\nabla\mathcal{L}(\mathbf{W}^t)\big)=\text{vec}\big(\nabla\mathcal{L}(\mathbf{W}^t)-\nabla\mathcal{L}(\mathbf{W}^*)\big),$$

by the fundamental theorem of calculus, we have

$$\begin{aligned}&\text{vec}\big(\nabla\mathcal{L}(\mathbf{W}^t)-\nabla\mathcal{L}(\mathbf{W}^*)\big)\\&=\int_0^1\nabla^2\mathcal{L}\big(\mathbf{W}^*+\theta(\mathbf{W}^t-\mathbf{W}^*)\big)d\theta\cdot\text{vec}(\mathbf{W}^t-\mathbf{W}^*),\end{aligned}$$

which implies

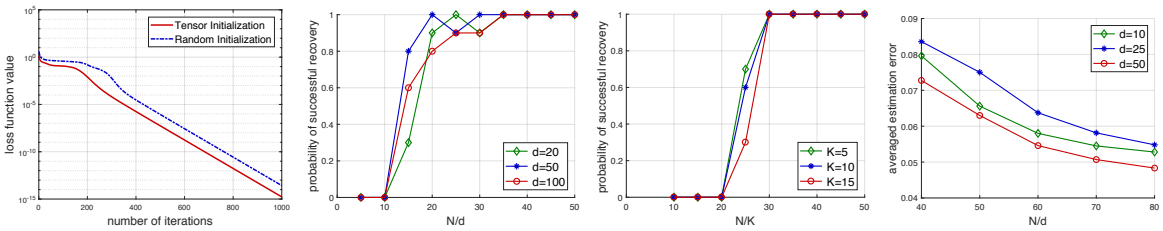
$$\text{vec}\big(\nabla\mathcal{L}(\mathbf{W}^t)\big)=\mathbf{H}_t\text{vec}(\mathbf{W}^t-\mathbf{W}^*),$$

where $\mathbf{H}_t=\int_0^1\nabla^2\mathcal{L}\big(\mathbf{W}^*+\theta(\mathbf{W}^t-\mathbf{W}^*)\big)d\theta\in\mathbb{R}^{dK\times dK}$. Note that by the inductive assumption, we have

$$\|\mathbf{W}^t-\mathbf{W}^*\|_F\le\frac{c\sigma_K}{\lambda\kappa^3K^2}\le\frac{\sigma_K}{2}.$$

Thus, according to Lemma 5.2, we obtain the upper bound of I_1

$$\begin{aligned}I_1&=-2\eta\cdot\text{vec}(\mathbf{W}^t-\mathbf{W}^*)^\top\mathbf{H}_t\text{vec}(\mathbf{W}^t-\mathbf{W}^*)\\&\quad+2\eta^2\cdot\text{vec}(\mathbf{W}^t-\mathbf{W}^*)^\top\mathbf{H}_t^\top\mathbf{H}_t\text{vec}(\mathbf{W}^t-\mathbf{W}^*)\\&\le2(-\eta+L\eta^2)\cdot\text{vec}(\mathbf{W}^t-\mathbf{W}^*)^\top\mathbf{H}_t\text{vec}(\mathbf{W}^t-\mathbf{W}^*),\end{aligned}$$



(a) convergence rate (b) sample complexity d (c) sample complexity K (d) statistical rate

Figure 2: (a) Comparison of convergence rate for gradient descent based algorithm using different initialization procedures. Here, we set input dimension $d = 10$, sample size $N = 5000$ and number of neurons $K = 5$. (b) Plot of successful recovery probability versus the ratio between sample size and input dimension N/d, which illustrates that the sample complexity scales linearly with d. (c) Plot of successful recovery probability versus the ratio between sample size and the number of neurons K, which demonstrates that the sample complexity scales linearly with K. (d) Plot of averaged estimation error versus the rescaled sample size N/d based on our method under different settings.

where the inequality follows from Lemma 5.2 and $L = c_1\kappa K^2$ is the Lipschitz parameter of $\nabla\mathcal{L}(\cdot)$. On the other hand, as for the term I_2, we have with probability at least $1 - c_3/d^{10}$ that

$$I_2 \leq 2(\eta\beta + \eta^2)\|\nabla\widehat{\mathcal{L}}_N(\mathbf{W}^t) - \nabla\mathcal{L}(\mathbf{W}^t)\|_F^2 + \frac{2\eta}{\beta}\|\mathbf{W}^t - \mathbf{W}^*\|_F^2,$$

where the inequality holds due to the Young's inequality, $\beta > 0$ is a constant that will be specified later. Thus according to Lemma 5.3, we can further obtain

$$I_2 \leq (\eta\beta + \eta^2)\frac{2c_2^2K^5d\log N}{N}(\|\mathbf{W}^*\|_F + \nu)^2 + \frac{2\eta}{\beta}\|\mathbf{W}^t - \mathbf{W}^*\|_F^2.$$

If we choose $\eta \leq 1/(2L)$ and $\beta = 4/\mu$, under condition that $\|\mathbf{W}^t - \mathbf{W}^*\|_F \leq c\sigma_K/(\lambda\kappa^3K^2)$, we can obtain

$$\begin{aligned}&\|\mathbf{W}^{t+1} - \mathbf{W}^*\|_F^2 - \left(1 + \frac{2\eta}{\beta}\right)\cdot\|\mathbf{W}^t - \mathbf{W}^*\|_F^2\\ &\leq (-2\eta + 2L\eta^2)\cdot\mathrm{vec}(\mathbf{W}^t - \mathbf{W}^*)^\top\mathbf{H}_t\mathrm{vec}(\mathbf{W}^t - \mathbf{W}^*)\\ &\quad + 2(\eta\beta + \eta^2)\cdot\frac{c_2^2K^5d\log N}{N}(\|\mathbf{W}^*\|_F + \nu)^2,\end{aligned}$$

which implies that

$$\begin{aligned}&\|\mathbf{W}^{t+1} - \mathbf{W}^*\|_F^2 - \left(1 + \frac{2\eta}{\beta}\right)\cdot\|\mathbf{W}^t - \mathbf{W}^*\|_F^2\\ &\leq \frac{9c_2^2\eta K^5d\log N}{\mu N}(\|\mathbf{W}^*\|_F + \nu)^2\\ &\quad + 2(-\eta + L\eta^2)\cdot\mathrm{vec}(\mathbf{W}^t - \mathbf{W}^*)^\top\mathbf{H}_t\mathrm{vec}(\mathbf{W}^t - \mathbf{W}^*),\end{aligned}$$

where μ is the lower bound of the smallest singular value of $\nabla^2\mathcal{L}(\mathbf{W})$ as in Lemma 5.1, the inequality is due to the selection of β, η and the fact that $L > \mu$.

According to Lemma 5.1, we can obtain

$$\|\mathbf{W}^{t+1} - \mathbf{W}^*\|_F^2 \leq \left(1 - \frac{\mu\eta}{2}\right)\cdot\|\mathbf{W}^t - \mathbf{W}^*\|_F^2 + \frac{18c_2^2\eta K^5d\log N}{\mu N}(\|\mathbf{W}^*\|_F^2 + \nu^2) \tag{5.2}$$

holds with probability at least $1 - c_3/d^{10}$.

Hence, as long as the sample size satisfies

$$N \geq \frac{c_4\lambda^2\kappa^6K^9d}{\sigma_K^2\mu^2}\log\left(\frac{\lambda\kappa Kd}{\sigma_K\mu}\right)\cdot(\|\mathbf{W}^*\|_F^2 + \nu^2),$$

we have $\mathbf{W}^{t+1}$ satisfies (5.1). Thus, we proved the inductive hypothesis. Finally, we conclude that with probability at least $1 - c_3/d^{10}$

$$\|\mathbf{W}^T - \mathbf{W}^*\|_F^2 \leq \left(1 - \frac{\mu\eta}{2}\right)^T\|\mathbf{W}^0 - \mathbf{W}^*\|_F^2 + \frac{c_5K^5d\log N}{\mu^2N}\cdot(\|\mathbf{W}^*\|_F^2 + \nu^2).$$

This completes the proof. □

6 EXPERIMENTS

In this section, we perform several experiments on synthetic datasets to justify our theory. In particular, we investigate the convergence rate of our algorithm under different initializations, the sample complexity dependence with respect to dimension d and the number of hidden neurons K, and the statistical error. We sample the input data $\{\mathbf{x}_i\}_{i=1}^N$ from standard Gaussian distribution, and generate the output labels $\{y_i\}_{i=1}^N$ based on the teacher network (1.1). The number of neurons in the hidden layer is set as $K = 5$. We generate the underlying parameter matrix $\mathbf{W}^* \in \mathbb{R}^{d\times K}$

such that $\mathbf{W}^* = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, where $\mathbf{U}$, $\mathbf{V}$ are the left and right singular matrices of a $d \times K$ standard Gaussian matrix, and $\mathbf{\Sigma}$ is a diagonal matrix. The smallest singular value of $\mathbf{W}^*$ is to be 1 and the largest one is set to be 2, and thus the condition number $\kappa = 2$.

To begin with, we study the convergence rate of our proposed Algorithm 1 in the noiseless case using different initialization procedures. In particular, we compare the tensor initialization algorithm proposed in Zhong et al. (2017) and random initialization procedure, where the initial estimator $\mathbf{W}^0$ is generated randomly from a standard Gaussian distribution. We choose the dimension d as 10 and sample size N as 5000. The step size η is set to be 0.5. For each intermediate iterate $\mathbf{W}^t$ returned by Algorithm 1, we compute the empirical loss function value $\widehat{\mathcal{L}}_N(\mathbf{W}^t)$. The logarithm of the empirical loss function value is plotted in Figure 2(a) against the number of iterations. It can be seen that both initialization methods, followed by gradient descent, achieve linear rate of convergence after a certain number of iterations, but tensor initialization leads to faster convergence than random initialization at early stage.

Moreover, we investigate the sample complexity requirement of the gradient descent algorithm in the noiseless setting. In particular, we consider three cases: (i) $d = 20$; (ii) $d = 50$; (iii) $d = 100$. For each case, we vary the sample size N, and repeat Algorithm 1 for 10 trials. A trial is considered to be successful if there exists a permutation matrix[3] $\mathbf{M}_\pi \in \mathbb{R}^{K\times K}$ such that the returned estimator $\mathbf{W}^T$ satisfies

$$\|\mathbf{W}^T - \mathbf{W}^* \cdot \mathbf{M}_\pi\|_F / \|\mathbf{W}^*\|_F \leq 10^{-3}.$$

The results of successful recovery probability of $\mathbf{W}^*$ under different ratio N/d are reported in Figure 2(b). It can be seen from the plot that the sample complexity required by tensor initialization followed by gradient descent for learning one-hidden-layer ReLU networks is linear in the dimension d, which is in agreement with our theory.

We also investigate the dependence of the sample complexity requirement in terms of the number of neurons K. To this end, we generate the underlying parameter matrix $\mathbf{W}^* \in \mathbb{R}^{d\times K}$ with $d = 20$, and we consider three cases: (i) $K = 5$; (ii) $K = 10$; (iii) $K = 15$. Figure 2(c) shows the sample complexity comparison for different K under random initialization. It suggests that the sample complexity is linear to the number of hidden units. Although our theoretical results require higher order dependence on K, the simulation results suggest that our method can achieves the linear dependence on K.

Finally, we study the statistical rate of our method in the noisy setting. In particular, we consider the following three cases: (i) $d = 10$, (ii) $d = 25$, (iii) $d = 50$. Each element of the noise vector $\boldsymbol{\epsilon} = [\epsilon_1, ..., \epsilon_N]^\top$ is generated independently from Gaussian distribution $\mathcal{N}(0, 0.1)$. We run Algorithm 1 with tensor initialization for each case over 10 trials, and report the averaged estimation error of the final output $\mathbf{W}^T$, i.e., $\|\mathbf{W}^T - \mathbf{W}^*\mathbf{M}_\pi\|_F$. Recall that $\mathbf{M}_\pi$ denotes the optimal permutation matrix with respect to $\mathbf{W}^T$ and $\mathbf{W}^*$. The results are displayed in Figure 2(d), which demonstrates that the averaged estimation error is well aligned with the rescaled sample size under different cases, which confirms that the statistical rate of the output of gradient descent for training one-hidden-layer ReLU networks is indeed in the order of $\widetilde{O}(d/N)$.

7 CONCLUSIONS AND FUTURE WORK

In this paper, we studied the empirical risk minimization for training one-hidden-layer ReLU networks using gradient descent. We proved that gradient descent can converge to the ground-truth parameters at a linear rate up to some statistical error with sample complexity $\widetilde{O}(d)$. While the presented results are specific to shallow neural networks, we believe that they can shed light on understanding the learning of deep networks.

As for future work, one important but challenging direction is to study the global optimization landscape and learning guarantees for deeper neural networks with more than one hidden layers. Another future direction is to investigate whether the Gaussian input assumption can be relaxed to more general distribution assumption as done by Du et al. (2017a) for learning one convolutional filter. Last but not least, it would be interesting to study the theoretical guarantee for learning ReLU networks using random initialized gradient descent.

Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments. This research was sponsored in part by the National Science Foundation IIS-1906169 and BIGDATA IIS-1855099. The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing any funding agencies.

[3] A permutation matrix is a square binary matrix that has exactly one entry of 1 in each row and each column and 0's elsewhere.

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., Ge, R., Ma, T. and Risteski, A. (2016). Provable learning of noisy-or networks. *arXiv preprint arXiv:1612.08795* .

Auer, P., Herbster, M. and Warmuth, M. K. (1996). Exponentially many local minima for single neurons. In *Advances in neural information processing systems*.

Bahdanau, D., Cho, K. and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* .

Barron, A. R. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information theory* **39** 930–945.

Blum, A. and Rivest, R. L. (1989). Training a 3-node neural network is np-complete. In *Advances in neural information processing systems*.

Brutzkus, A. and Globerson, A. (2017). Globally optimal gradient descent for a convnet with gaussian inputs. *arXiv preprint arXiv:1702.07966* .

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

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B. and LeCun, Y. (2015). The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*.

Cohen, N., Sharir, O. and Shashua, A. (2016). On the expressive power of deep learning: A tensor analysis. In *Conference on Learning Theory*.

Daniely, A., Frostig, R. and Singer, Y. (2016). Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*.

Du, S. S. and Lee, J. D. (2018). On the power of over-parametrization in neural networks with quadratic activation. *arXiv preprint arXiv:1803.01206* .

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

Du, S. S., Lee, J. D. and Tian, Y. (2017a). When is a convolutional filter easy to learn? *arXiv preprint arXiv:1709.06129* .

Du, S. S., Lee, J. D., Tian, Y., Poczos, B. and Singh, A. (2017b). Gradient descent learns one-hidden-layer cnn: Don't be afraid of spurious local minima. *arXiv preprint arXiv:1712.00779* .

Du, S. S., Zhai, X., Poczos, B. and Singh, A. (2019). Gradient descent provably optimizes over-parameterized neural networks. *ICLR* .

Fu, H., Chi, Y. and Liang, Y. (2018). Local geometry of one-hidden-layer neural networks for logistic regression. *arXiv preprint arXiv:1802.06463* .

Gao, W., Makkuva, A. V., Oh, S. and Viswanath, P. (2018). Learning one-hidden-layer neural networks under general input distributions. *arXiv preprint arXiv:1810.04133* .

Ge, R., Huang, F., Jin, C. and Yuan, Y. (2015). Escaping from saddle pointsonline stochastic gradient for tensor decomposition. In *Conference on Learning Theory*.

Ge, R., Kuditipudi, R., Li, Z. and Wang, X. (2019). Learning two-layer neural networks with symmetric inputs. *In ICLR 2019* .

Ge, R., Lee, J. D. and Ma, T. (2017). Learning one-hidden-layer neural networks with landscape design. *arXiv preprint arXiv:1711.00501* .

Goel, S., Kanade, V., Klivans, A. and Thaler, J. (2016). Reliably learning the relu in polynomial time. *arXiv preprint arXiv:1611.10258* .

Hanin, B. (2017). Universal function approximation by deep neural nets with bounded width and relu activations. *arXiv preprint arXiv:1708.02691* .

Hanin, B. and Sellke, M. (2017). Approximating continuous functions by relu nets of minimal width. *arXiv preprint arXiv:1710.11278* .

Hardt, M. and Ma, T. (2016). Identity matters in deep learning. *arXiv preprint arXiv:1611.04231* .

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.

Hornik, K. (1991). Approximation capabilities of multilayer feedforward networks. *Neural networks* **4** 251–257.

Jain, P., Netrapalli, P. and Sanghavi, S. (2013). Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM.

Janzamin, M., Sedghi, H. and Anandkumar, A. (2015). Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *arXiv preprint arXiv:1506.08473* .

Jin, C., Ge, R., Netrapalli, P., Kakade, S. M. and Jordan, M. I. (2017). How to escape saddle points efficiently. *arXiv preprint arXiv:1703.00887* .

Kakade, S. M., Kanade, V., Shamir, O. and Kalai, A. (2011). Efficient learning of generalized linear and single index models with isotonic regression. In *Advances in Neural Information Processing Systems*.

Kalai, A. T. and Sastry, R. (2009). The isotron algorithm: High-dimensional isotonic regression. In *COLT*. Citeseer.

Kawaguchi, K. (2016). Deep learning without poor local minima. In *Advances in Neural Information Processing Systems*.

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

Li, Y. and Liang, Y. (2018). Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems*.

Li, Y. and Yuan, Y. (2017). Convergence analysis of two-layer neural networks with relu activation. *arXiv preprint arXiv:1705.09886* .

Livni, R., Shalev-Shwartz, S. and Shamir, O. (2014). On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*.

Lu, Z., Pu, H., Wang, F., Hu, Z. and Wang, L. (2017). The expressive power of neural networks: A view from the width. *arXiv preprint arXiv:1709.02540* .

Mei, S., Bai, Y. and Montanari, A. (2016). The landscape of empirical risk for non-convex losses. *arXiv preprint arXiv:1607.06534* .

Montufar, G. F., Pascanu, R., Cho, K. and Bengio, Y. (2014). On the number of linear regions of deep neural networks. In *Advances in neural information processing systems*.

Mukherjee, A. and Basu, A. (2017). Lower bounds over boolean inputs for deep neural networks with relu gates. *arXiv preprint arXiv:1711.03073* .

Nguyen, Q. and Hein, M. (2017). The loss surface of deep and wide neural networks. *arXiv preprint arXiv:1704.08045* .

Pan, X. and Srikumar, V. (2016). Expressiveness of rectifier networks. In *International Conference on Machine Learning*.

Poole, B., Lahiri, S., Raghu, M., Sohl-Dickstein, J. and Ganguli, S. (2016). Exponential expressivity in deep neural networks through transient chaos. In *Advances In Neural Information Processing Systems*.

Raghu, M., Poole, B., Kleinberg, J., Ganguli, S. and Sohl-Dickstein, J. (2016). On the expressive power of deep neural networks. *arXiv preprint arXiv:1606.05336* .

Safran, I. and Shamir, O. (2016). On the quality of the initial basin in overspecified neural networks. In *International Conference on Machine Learning*.

Safran, I. and Shamir, O. (2017). Spurious local minima are common in two-layer relu neural networks. *arXiv preprint arXiv:1712.08968* .

Sedghi, H. and Anandkumar, A. (2014). Provable methods for training neural networks with sparse connectivity. *arXiv preprint arXiv:1412.2693* .

Shalev-Shwartz, S., Shamir, O. and Shammah, S. (2017a). Failures of gradient-based deep learning. In *International Conference on Machine Learning*.

Shalev-Shwartz, S., Shamir, O. and Shammah, S. (2017b). Weight sharing is crucial to succesful optimization. *arXiv preprint arXiv:1706.00687* .

Shamir, O. (2016). Distribution-specific hardness of learning neural networks. *arXiv preprint arXiv:1609.01037* .

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.

Soltanolkotabi, M. (2017). Learning relus via gradient descent. *arXiv preprint arXiv:1705.04591* .

Soltanolkotabi, M., Javanmard, A. and Lee, J. D. (2017). Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *arXiv preprint arXiv:1707.04926* .

Soudry, D. and Carmon, Y. (2016). No bad local minima: Data independent training error guarantees for multilayer neural networks. *arXiv preprint arXiv:1605.08361* .

Telgarsky, M. (2016). Benefits of depth in neural networks. *arXiv preprint arXiv:1602.04485* .

Tian, Y. (2017). An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. *arXiv preprint arXiv:1703.00560* .

Vershynin, R. (2010). Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027* .

Xie, B., Liang, Y. and Song, L. (2017). Diverse neural network learns true target functions. In *Artificial Intelligence and Statistics*.

Yarotsky, D. (2017). Error bounds for approximations with deep relu networks. *Neural Networks* **94** 103–114.

Yarotsky, D. (2018). Optimal approximation of continuous functions by very deep relu networks. *arXiv preprint arXiv:1802.03620* .

Yun, C., Sra, S. and Jadbabaie, A. (2017). Global optimality conditions for deep neural networks. *arXiv preprint arXiv:1707.02444* .

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

Zhang, Y., Lee, J., Wainwright, M. and Jordan, M. (2017). On the learnability of fully-connected neural networks. In *Artificial Intelligence and Statistics*.

Zhang, Y., Lee, J. D., Wainwright, M. J. and Jordan, M. I. (2015). Learning halfspaces and neural networks with random initialization. *arXiv preprint arXiv:1511.07948* .

Zhong, K., Song, Z., Jain, P., Bartlett, P. L. and Dhillon, I. S. (2017). Recovery guarantees for one-hidden-layer neural networks. *arXiv preprint arXiv:1706.03175* .

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