Improved Linear Convergence of Training CNNs With Generalizability Guarantees: A One-Hidden-Layer Case

Shuai Zhang, *Graduate Student Member, IEEE*, Meng Wang, *Member, IEEE*, Jinjun Xiong, *Senior Member, IEEE*, Sijia Liu, *Member, IEEE*, and Pin-Yu Chen, *Member, IEEE*

***Abstract*— We analyze the learning problem of one-hidden-layer nonoverlapping convolutional neural networks with the rectified linear unit (ReLU) activation function from the perspective of model estimation. The training outputs are assumed to be generated by the neural network with the unknown ground-truth parameters plus some additive noise, and the objective is to estimate the model parameters by minimizing a nonconvex squared loss function of the training data. Assuming that the training set contains a finite number of samples generated from the Gaussian distribution, we prove that the accelerated gradient descent (GD) algorithm with a proper initialization converges to the ground-truth parameters (up to the noise level) with a linear rate even though the learning problem is nonconvex. Moreover, the convergence rate is proved to be faster than the vanilla GD. The initialization can be achieved by the existing tensor initialization method. In contrast to the existing works that assume an infinite number of samples, we theoretically establish the sample complexity of the required number of training samples. Although the neural network considered here is not deep, this is the first work to show that accelerated GD algorithms can find the global optimizer of the nonconvex learning problem of neural networks. This is also the first work that characterizes the sample complexity of gradient-based methods in learning convolutional neural networks with the nonsmooth ReLU activation function. This work also provides the tightest bound so far of the estimation error with respect to the output noise.**

***Index Terms*— Accelerated gradient descent (GD), convolutional neural networks, generalizability, global optimality, linear convergence.**

I. Introduction

NEURAL networks, especially convolutional neural networks (CNNs), have demonstrated superior performance in machine learning for image classification [16] and recognition [19], natural language processing [5], and strategic game program [33]. Compared with fully connected neural networks, CNNs require fewer coefficients and can better capture local features [20], and thus, perform well in applications, such as image and video processing.

Learning a neural network needs to find appropriate parameters for the hidden layers using the training data and is achieved by minimizing a nonconvex empirical loss function over the choices of the model parameters. The nonconvex learning problem is usually solved by a first-order gradient descent (GD) algorithm. The convergence to the global optimal, however, is not guaranteed naturally due to the existence of spurious local minima. Another major hurdle to the widespread acceptance of deep learning is the lack of analytical performance guarantees about whether the parameters learned from the training data perform well on the testing data, i.e., the generalizability of the learned model. A learned model generalizes well to the testing data provided that it is a global minimizer of the population loss function, which takes the expectation over the distribution of testing samples. Since the distribution is unknown, one minimizes the empirical loss function of the training data assuming that the training data are drawn from the same distribution. Moreover, a large number of training samples are required to obtain a network model with powerful feature representation capability [6], while the method may perform poorly when the number of training samples is small [4]. The theoretical characterization of the required size of the training data for given network architecture is vastly unavailable.

To analyze the learning performance, one line of research focuses on the overparameterized case that the number of parameters in the neural network is larger than the number of training samples [1], [2], [14], [15], [24], [27], [30], [34]. In particular, the optimization problem has no spurious local minima [24], [34], [43], and GD methods can indeed find the global minimum of the empirical loss function. Nevertheless, the overparameterized models may experience overfitting issues in practice [42], [43]. Moreover, when overparameterized, there is no guarantee by Vapnik–Chervonenkis (VC)-dimension learning theory that the empirical loss function is close to the population loss, and thus, the generalizability of the learned model to the testing data is unknown. Reference [1] develops a new analysis tool to explore the generalizability under overparameterization assumption. The convergence rate provided by [1] is sublinear, and the sizes of neural networks increase as a polynomial function of the inverse of the desired testing error, which implies a high computational cost. Moreover, the training error and the generalization error are analyzed separately, and

Manuscript received August 25, 2019; revised March 25, 2020; accepted June 28, 2020. Date of publication July 29, 2020; date of current version June 2, 2021. This work was supported in part by the Air Force Office of Scientific Research (AFOSR) under Grant FA9550-20-1-0122, in part by NSF 1932196, and in part by the Rensselaer-IBM AI Research Collaboration, a part of the IBM AI Horizons Network. *(Corresponding author: Meng Wang.)*

Shuai Zhang and Meng Wang are with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: zhangs21@rpi.edu; wangm7@rpi.edu).

Jinjun Xiong, Sijia Liu, and Pin-Yu Chen are with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: jinjun@us.ibm.com; sijia.liu@ibm.com; pin-yu.chen@ibm.com).

This article has supplementary downloadable material available at https://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2020.3007399

it is not clear if both a small training error and a small generalization error can be achieved simultaneously.

References [18] and [39] study the convergence to the global optimal for shallow neural networks when the data is linearly separable. Assuming the rectified linear unit (ReLU) activation function and the hinge loss function, [39] can detect all the spurious local minima and saddle points, and the generalization error of the learned model approaches zero when the number of samples goes to infinite. However, if the data are linearly separable, simple algorithms, such as perceptron [29], can find a classifier in finite steps. Moreover, the detection method of the spurious local minima and saddle points in [39] only apply the ReLU activation function and hinge loss function, and the method does not extend to other activation functions and loss functions.

One recent line of research assumes the existence of a ground-truth model that maps the input data to the output data. Then the set of the ground-truth model parameters is a global minimizer of both the population and the empirical risk functions. The learning problem can be viewed as a model estimation problem. If the parameters are accurately estimated, the generalizability of the testing data is guaranteed. This article follows this line of research.

To simplify the analysis, one standard trick in this line of research is to assume that the number of input data is infinite so that the empirical loss function is simplified to the population loss function that is easier to analyze. Most existing theoretical results are centered on one-hidden-layer shallow neural networks as the analyses quickly become intractable when the number of layers increases. The input data are usually assumed to follow the Gaussian distribution [32] or some distributions that are rationally invariant [7]. References [3], [8], and [36] analyze the landscape of the population loss function of a simple one-hidden-layer neural network with only one or two nodes and show that there exists a considerably large convex region near the global optimum. Then, a random initial point lies in this region with a constant probability, and GD algorithms converge to the global minimum. This result does not easily generalize to general neural networks as spurious local minima are fairly common for neural networks with even one hidden layer but multiple nodes [31]. Some works [10], [22], [23] seek to obtain a good optimization landscape through changing the neural network structure. Reference [22] adds an identity mapping after the hidden layer to improve the convergence of the GD algorithm. An additional regularization term is added to the loss function in [10] such that the ground-truth parameters are still close to the global minimum, and spurious local minima are excluded. An exponential node is added in each layer of an arbitrary neural network such that all local minima are global minima [23]. Another work [12] developed a new iterative algorithm named Convotron, which applied a modified gradient descent update in each iteration and did not require initialization.

In the practical case of a finite number of samples, the nice properties of the population loss function do not directly generalize to the empirical loss function. Some recent works study the training performance with a finite number of samples [9], [40], [44]–[46]. If the number of samples is greater than a certain threshold, referred to the sample complexity, [40] shows that iterates converge to the ground-truth parameters for one-hidden-layer neural networks. However, the sample complexity is suboptimal as it is a high order polynomial with respect to the dimension of the input data. With the tensor initialization method [46], GD algorithms are proved to converge to the ground-truth parameters linearly in one-hidden-layer neural networks, and the sample complexity is nearly linear in the dimension of the input data [9], [44]–[46]. However, the analyses in [9], [45], and [46] are limited to smooth activation functions and exclude the widely used nonsmooth activation function, ReLU. Among them, only [44] studies the ReLU activation function but focuses on fully connected neural networks. Reference [44] can only guarantee the convergence to the ground truth up to some nonzero estimation error, even when the data are noiseless.

The majority of the above-mentioned works assume that data are noiseless, which may not be realistic in practice. Only [10] and [44] consider the cases that the output data contain additive noise that is independent of the input. The noise is assumed to be zero mean by Ge *et al.* [10], and they analyze the stochastic GD through expectation. Thus, the noise does not affect their analyses and results. The result in [44] guarantees the convergence of GD provided that the initialization is sufficiently close to the ground-truth parameters, but no discussion is provided about whether the initialization in [44] satisfies this assumption or not.

All the aforementioned works analyze standard GD algorithms. It is well known that Accelerated GD (AGD) methods, such as the Nesterov accelerated gradient (NAG) method [26] and the heavy ball method [28], converge faster than vanilla GD. However, the analyses for GD do not generalize directly to AGD because of the additional momentum term introduced in AGD. Only [35] and [41] explore the numerical performance of AGD in neural networks. No theoretical analysis of AGD is reported in [35]. Reference [41] analyzes AGD from a general optimization perspective, and it is not clear whether the neural network learning problem satisfies the assumptions in [41].

This article provides novel contributions to the theoretical analyses of neural networks in three aspects. First, this article provides the first theoretical analysis of AGD methods in learning neural networks. We prove analytically that the AGD method can converge to the ground-truth parameters linearly, and its convergence rate is faster than vanilla GD. Second, it is the first work that explicitly proves the convergence of the proposed learning algorithm to the ground-truth parameters (or nearby) when the data contain noise. We characterize the relationship between the learning accuracy and noise level quantitatively. Our error bound is much tighter than that in [44], and [44] makes assumptions about the initialization without any justification. In the special case of noiseless data, our parameter estimation is exact, while the method in [44] is not. Third, it provides the first tight generalizability analysis of the widely used convolutional neural networks with the nonsmooth ReLU activation functions. Specifically, we prove that for one-hidden-layer nonoverlapping convolutional neural networks, if initialized using the tensor method, and the number of samples exceeds our characterized sample complexity, both GD and ADG converge to a global minimum linearly up to the noise level. Our sample complexity is orderwise optimal with respect to the dimension of the node parameters. Our estimation error bound of the ground-truth parameters is much tighter than a direct application of the existing results for fully connected neural networks, such as [44] to CNN.

The rest of this article is organized as follows. Section II introduces the problem formulation. The algorithm and major theorems are presented in Section III. Section IV shows the

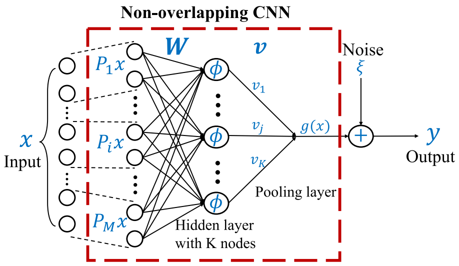


Fig. 1. One-hidden-layer nonoverlapping CNN.

simulation results, and Section V concludes this article. All the proofs are in the Appendix.

Notation: Vectors are bold lowercase, matrices and tensors are bold uppercase, and scalars are in normal font. For instance, $\boldsymbol{Z}$ is a matrix, and $\boldsymbol{z}$ is a vector. z_i denotes the ith entry of $\boldsymbol{z}$, and Z_{ij} denotes the (i, j)th entry of $\boldsymbol{Z}$. $\boldsymbol{I}$ and $\boldsymbol{e}_i$ denote the identity matrix and the ith standard basis vector. $\boldsymbol{Z}^T$ denotes the transpose of $\boldsymbol{Z}$, similarly for $\boldsymbol{z}^T$. $\|\boldsymbol{z}\|$ denotes the ℓ_2-norm of a vector $\boldsymbol{z}$, and $\|\boldsymbol{Z}\|_2$ and $\|\boldsymbol{Z}\|_F$ denote the spectral norm and Frobenius norm of a matrix $\boldsymbol{Z}$, respectively. We use $\sigma_i(\boldsymbol{Z})$ to denote the ith largest singular value of $\boldsymbol{Z}$. The outer product of a group of vectors $\boldsymbol{z}_i \in \mathbb{R}^{n_i}, 1 \leq i \leq l$ and $l \in \mathbb{N}^+$, is defined as $\boldsymbol{T} = \boldsymbol{z}_1 \otimes \cdots \otimes \boldsymbol{z}_l \in \mathbb{R}^{n_1\times\cdots\times n_l}$ with $T_{j_1,\ldots,j_l} = (\boldsymbol{z}_1)_{j_1}\cdots(\boldsymbol{z}_l)_{j_l}$. Let $\mathcal{L}_i$ be a linear operator from $\mathbb{R}^{n_i}$ to $\mathbb{R}^{d_i}$ with $1 \leq i \leq l$, then $\boldsymbol{T}(\mathcal{L}_1,\ldots,\mathcal{L}_l) = \mathcal{L}_1(\boldsymbol{z}_1)\otimes\cdots\otimes\mathcal{L}_l(\boldsymbol{z}_l) \in \mathbb{R}^{d_1\times\cdots\times d_l}$. Moreover, $f(d) = O(g(d))$ means that if for some constant $C > 0$, $f(d) \leq Cg(d)$ holds when d is sufficiently large. $f(d) = \Theta(g(d))$ means that for some constants $c > 0$ and $C > 0$, $cg(d) \leq f(d) \leq Cg(d)$ holds when d is sufficiently large. In the Appendix, we use $f(d) \gtrsim (\lesssim) g(d)$ to denote there exists some positive constant C such that $f(d) \geq (\leq) C \cdot g(d)$ when d is sufficiently large.

II. Problem Formulation

Following [45], we consider the regression setup in this article as follows. Given N input data $\boldsymbol{x}_n \in \mathbb{R}^p$, $n = 1, 2, \ldots, N$ that are independent and identically distributed (i.i.d.) from the standard Gaussian distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{p\times p})$, the resulting outputs $y_n \in \mathbb{R}$, $n = 1, 2, \ldots, N$ are generated from $\{\boldsymbol{x}_n\}_{n=1}^N$ by a one-hidden-layer nonoverlapping convolutional neural network shown in Fig. 1. The hidden layer has K nodes. We use the vector $\boldsymbol{w}_j^* \in \mathbb{R}^d$ to denote the weight parameters for the jth node in the hidden layer and define the weight matrix $\boldsymbol{W}^* = \left[\boldsymbol{w}_1^*, \boldsymbol{w}_2^*, \ldots, \boldsymbol{w}_K^*\right] \in \mathbb{R}^{d\times K}$. Followed by the hidden layer, there is a pooling layer with ground-truth parameters $\boldsymbol{v}^* \in \mathbb{R}^d$. We assume $K < d$ throughout this article because K is the constant, while d increases as the dimension of the input data increases. $\sigma_i = \sigma_i(\boldsymbol{W}^*)$ denotes the ith largest singular value of $\boldsymbol{W}^*$. We define $\kappa = \sigma_1(\boldsymbol{W}^*)/\sigma_K(\boldsymbol{W}^*)$ as the conditional number of $\boldsymbol{W}^*$ and $\gamma = \Pi_{j=1}^K(\sigma_j(\boldsymbol{W}^*)/\sigma_K(\boldsymbol{W}^*))$.

Each input data $\boldsymbol{x}_n$ is partitioned into M nonoverlapping patches, denoted by $\boldsymbol{P}_i\boldsymbol{x}_n \in \mathbb{R}^d$, $i = 1, \ldots, M$. $\boldsymbol{P}_i \in \mathbb{R}^{d\times p}$, $i = 1, \ldots, M$, are a series of matrices that satisfy the following properties: (1) there exists one and only one nonzero entry with value 1 in each row of $\boldsymbol{P}_i$; (2) $\langle \boldsymbol{P}_{i_1}, \boldsymbol{P}_{i_2}\rangle = 0$ for $i_1 \neq i_2$.[1] A simple example of $\{\boldsymbol{P}_i\}_{i=1}^M$ is

$$\boldsymbol{P}_i = \Bigg[\underbrace{\boldsymbol{0}_{d\times d} \cdots \boldsymbol{0}_{d\times d}}_{(i-1)\text{submatrices}} \quad \boldsymbol{I}_{d\times d} \quad \underbrace{\boldsymbol{0}_{d\times d} \cdots \boldsymbol{0}_{d\times d}}_{(M-i)\text{submatrices}}\Bigg].$$

The output y_n can be written as

$$y_n = g(\boldsymbol{x}_n) + \xi_n = \sum_{j=1}^{K}\sum_{i=1}^{M} v_j^*\phi\Big(\boldsymbol{w}_j^{*T}\boldsymbol{P}_i\boldsymbol{x}_n\Big) + \xi_n \qquad (1)$$

for $1 \leq n \leq N$, where ξ_n is the additive stochastic noise.

Throughout this article, we assume bounded noise with zero mean and use $|\xi|$ to denote the upper bound such that $|\xi_n| \leq |\xi|$ for all n. In practice, the mapping from the input to output data may not be modeled exactly by a neural network due to the random fluctuations or measurement errors in the data. The additive noise better characterizes the relations in real data sets.

The activation function $\phi(z) = \max\{z, 0\}$ is the ReLU function, which is widely used in various applications [11], [13], [21], [25]. Note that if the activation function is homogeneous, such as ReLU, one can assume v_j^* to be either $+1$ or -1 without loss of generality. That is because $v_j^*\phi(\boldsymbol{w}_j^{*T}\boldsymbol{P}_i\boldsymbol{x}_n) = \text{sign}(v_j^*)\phi(|v_j^*|\boldsymbol{w}_j^{*T}\boldsymbol{P}_i\boldsymbol{x}_n)$ for a homogeneous ϕ. We can just let $\widetilde{\boldsymbol{w}}_j^* = |v_j^*|\boldsymbol{w}_j^*$ and $\widetilde{v}_j^* = \text{sign}(v_j^*)$ and use $\{\widetilde{\boldsymbol{w}}_j^*\}_{j=1}^K$ and $\{\widetilde{v}_j^*\}_{j=1}^K$ as ground-truth parameters equivalently. Therefore, we assume $v_j^* \in \{+1, -1\}$ for any $1 \leq j \leq K$ throughout this article.

Given any estimated $\boldsymbol{W} \in \mathbb{R}^{d\times K}$ and $\boldsymbol{v} \in \mathbb{R}^K$ of the weight matrix $\boldsymbol{W}^*$ and $\boldsymbol{v}^*$, the empirical squared loss function[2] of the training set $\mathcal{D} = \{\boldsymbol{x}_n, y_n\}_{n=1}^N$ is defined as

$$\hat{f}_{\mathcal{D}}(\boldsymbol{W}, \boldsymbol{v}) = \frac{1}{2N}\sum_{n=1}^{N}\left(\sum_{j=1}^{K}\boldsymbol{v}_j\sum_{i=1}^{M}\phi(\boldsymbol{w}_j^T\boldsymbol{P}_i\boldsymbol{x}_n) - y_n\right)^2. \qquad (2)$$

Our goal is to estimate the ground-truth weight matrix $\boldsymbol{W}^*$ and $\boldsymbol{v}^*$ via solving the following problem:

$$\min_{\boldsymbol{W}\in\mathbb{R}^{d\times K}, \boldsymbol{v}\in\mathbb{R}^K} : \hat{f}_{\mathcal{D}}(\boldsymbol{W}, \boldsymbol{v}). \qquad (3)$$

Clearly $(\boldsymbol{W}^*, \boldsymbol{v}^*)$ is a global minimizer to (3) when measurements are noiseless, i.e., $\xi_n = 0$ for all n. However, (3) is a nonconvex optimization problem and is not easy to solve.

III. Algorithm and Theoretical Results

We propose to solve the nonconvex problem (3) via the heavy ball method [28]. The algorithm is initialized via the tensor method [46]. Although the tensor initialization is designed for fully connected neural networks in [46], we can extend it to nonoverlapping convolutional neural networks with minor changes. $\hat{\boldsymbol{v}}$ is estimated through the tensor initialization. During each iteration, we update $\boldsymbol{W}$ through the AGD algorithm. Compared with the vanilla GD, in the $(t + 1)$th iteration, an additional momentum term, denoted by $\beta(\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t-1)})$, is added to the update, where $\boldsymbol{W}^{(t)}$ is the estimation in iteration t. The momentum represents the direction of the previous iterations. Hence, besides moving along the GD direction with a step size of η, $\boldsymbol{W}^{(t)}$ is further moved along the direction of previous steps with a parameter of β. During each iteration, a fresh subset of data is applied to estimate the GD. Such disjoint subsets guarantee the independence of $\hat{f}_{\mathcal{D}_t}$ over the iterations. This is a standard analysis technique [45], [46], and not necessarily in numerical experiments. The initialization algorithm is summarized in Section III-A, and Algorithm 1 summarizes our proposed algorithm to solve (3).

[1] Such requirement on $\boldsymbol{P}_i$ guarantees the independence of each patches and will be used in the proofs.

[2] Besides the mean squared error, another choice of the loss function, especially in classification problems, is the cross entropy, see [9].

Algorithm 1 Accelerated GD Algorithm With Tensor Initialization

```
Input: training data $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, gradient step size $\eta$, momentum parameter $\beta$, and thresholding error parameter $\varepsilon$;
Initialization: $\boldsymbol{W}^{(0)}, \hat{\boldsymbol{v}}$ through Tensor Initialization via Subroutine 1;
Partition $\mathcal{D}$ into $T = \log(1/\varepsilon)$ disjoint subsets, denoted as $\{\mathcal{D}_i\}_{i=1}^T$;
for $t = 1, 2, \ldots, T$ do
    $\boldsymbol{W}^{(t+1)} = \boldsymbol{W}^{(t)} - \eta\nabla\hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)}, \hat{\boldsymbol{v}}) + \beta(\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t-1)})$
end forreturn $\boldsymbol{W}^{(T)}$ and $\hat{\boldsymbol{v}}$.
```

A. Initialization via Tensor Method

In this section, we first briefly introduce the tensor initialization method that is built upon [46, Algorithm 1]. We then provide the first theoretical performance guarantee of the tensor initialization method when the output contains noise in Lemma 1, while the result in [46] only applies to noiseless measurements.

The tensor initialization method in [46] is designed for the fully connected neural networks. To handle the convolutional neural networks, the definitions of the high-order moments [see (5)–(7)] are modified by replacing $\boldsymbol{x}$ in [46, Definition 5.1] with $\boldsymbol{P}_i\boldsymbol{x}$. All the other steps mainly follow [46].

Following [46], we define a special outer product, denoted by $\widetilde{\otimes}$. For any vector $\boldsymbol{v} \in \mathbb{R}^{d_1}$ and $\boldsymbol{Z} \in \mathbb{R}^{d_1 \times d_2}$

$$\boldsymbol{v}\widetilde{\otimes}\boldsymbol{Z} = \sum_{i=1}^{d_2}(\boldsymbol{v} \otimes \boldsymbol{z}_i \otimes \boldsymbol{z}_i + \boldsymbol{z}_i \otimes \boldsymbol{v} \otimes \boldsymbol{z}_i + \boldsymbol{z}_i \otimes \boldsymbol{z}_i \otimes \boldsymbol{v}) \quad (4)$$

where $\otimes$ is the outer product and $\boldsymbol{z}_i$ is the ith column of $\boldsymbol{Z}$. Next, we pick any $i \in \{1, 2, \ldots, K\}$ and define

$$\boldsymbol{M}_{i,1} = \mathbb{E}_{\boldsymbol{x}}\{y\boldsymbol{x}\} \in \mathbb{R}^d \quad (5)$$

$$\boldsymbol{M}_{i,2} = \mathbb{E}_{\boldsymbol{x}}\{y[(\boldsymbol{P}_i\boldsymbol{x}) \otimes (\boldsymbol{P}_i\boldsymbol{x}) - \boldsymbol{I}]\} \in \mathbb{R}^{d \times d} \quad (6)$$

$$\boldsymbol{M}_{i,3} = \mathbb{E}_{\boldsymbol{x}}\big\{y\big[(\boldsymbol{P}_i\boldsymbol{x})^{\otimes 3} - (\boldsymbol{P}_i\boldsymbol{x})\widetilde{\otimes}\boldsymbol{I}\big]\big\} \in \mathbb{R}^{d \times d \times d} \quad (7)$$

where $\boldsymbol{z}^{\otimes 3} := \boldsymbol{z} \otimes \boldsymbol{z} \otimes \boldsymbol{z}$, and $\mathbb{E}_{\boldsymbol{x}}$ is the expectation over $\boldsymbol{x}$.

From [46, Claim 5.2], there exist some known constants $\psi_i, i = 1, 2, 3$, such that

$$\boldsymbol{M}_{i,1} = \sum_{j=1}^K \psi_1 \cdot v_j^*\left\|\boldsymbol{w}_j^*\right\| \cdot \overline{\boldsymbol{w}}_j^* \quad (8)$$

$$\boldsymbol{M}_{i,2} = \sum_{j=1}^K \psi_2 \cdot v_j^*\left\|\boldsymbol{w}_j^*\right\| \cdot \overline{\boldsymbol{w}}_j^*\overline{\boldsymbol{w}}_j^{*T} \quad (9)$$

$$\boldsymbol{M}_{i,3} = \sum_{j=1}^K \psi_3 \cdot v_j^*\left\|\boldsymbol{w}_j^*\right\| \cdot \overline{\boldsymbol{w}}_j^{*\otimes 3} \quad (10)$$

where $\overline{\boldsymbol{w}}_j^* = \boldsymbol{w}_j^*/\|\boldsymbol{w}_j^*\|_2$ in (5)–(7) is the normalization of $\boldsymbol{w}_j^*$.

$\boldsymbol{M}_{i,1}$, $\boldsymbol{M}_{i,2}$, and $\boldsymbol{M}_{i,3}$ can be estimated through the samples $\{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, and let $\widehat{\boldsymbol{M}}_{i,1}$, $\widehat{\boldsymbol{M}}_{i,2}$, and $\widehat{\boldsymbol{M}}_{i,3}$ denote the corresponding estimates. First, we will decompose the rank-k tensor $\boldsymbol{M}_{i,3}$ and obtain the $\{\overline{\boldsymbol{w}}_j^*\}_{j=1}^K$. By applying the tensor decomposition method [17] to $\widehat{\boldsymbol{M}}_{i,3}$, the outputs, denoted by $\widehat{\overline{\boldsymbol{w}}}_j^*$, are the estimations of $\{\boldsymbol{s}_j\overline{\boldsymbol{w}}_j^*\}_{j=1}^K$, where s_j is an unknown sign. Second, we will estimate s_j, $\boldsymbol{v}_j^*$ and $\|\boldsymbol{w}_j^*\|_2$ through $\boldsymbol{M}_{i,1}$ and $\boldsymbol{M}_{i,2}$. Note that $\boldsymbol{M}_{i,2}$ does not contain the information of $\boldsymbol{s}_j$ because $\boldsymbol{s}_j^2$ is always 1. Then, through solving the following two optimization problem:

$$\widehat{\boldsymbol{\alpha}}_1 = \arg\min_{\boldsymbol{\alpha}_1 \in \mathbb{R}^K} : \left|\widehat{\boldsymbol{M}}_{i,1} - \sum_{j=1}^K \psi_1\alpha_{1,j}\widehat{\overline{\boldsymbol{w}}}_j^*\right|$$

$$\widehat{\boldsymbol{\alpha}}_2 = \arg\min_{\boldsymbol{\alpha}_2 \in \mathbb{R}^K} : \left|\widehat{\boldsymbol{M}}_{i,2} - \sum_{j=1}^K \psi_2\alpha_{2,j}\widehat{\overline{\boldsymbol{w}}}_j^*\widehat{\overline{\boldsymbol{w}}}_j^{*T}\right|. \quad (11)$$

The estimation of s_j can be given as $\hat{s}_j = \operatorname{sign}(\widehat{\alpha}_{1,j}/\widehat{\alpha}_{2,j})$. In addition, we know that $|\widehat{\alpha}_{1,j}|$ is the estimation of $\|\boldsymbol{w}_j^*\|$ and $\hat{v}_j = \operatorname{sign}(\widehat{\alpha}_{1,j}/\boldsymbol{s}_j)$. Thus, $\boldsymbol{W}^{(0)}$ is given as $\big[\operatorname{sign}(\widehat{\alpha}_{2,1})\widehat{\alpha}_{1,1}\widehat{\overline{\boldsymbol{w}}}_1^*, \ldots, \operatorname{sign}(\widehat{\alpha}_{2,K})\widehat{\alpha}_{1,K}\widehat{\overline{\boldsymbol{w}}}_K^*\big]$.

To reduce the computational complexity of tensor decomposition, one can project $\widehat{\boldsymbol{M}}_{i,3}$ to a lower dimensional tensor [46]. The idea is to first estimate the subspace spanned by $\{\boldsymbol{w}_j^*\}_{j=1}^K$, and let $\widehat{\boldsymbol{V}}$ denote the estimated subspace. Then, from (7) and (10), we know that $\boldsymbol{M}_{i,3}(\widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}}) \in \mathbb{R}^{K \times K \times K}$ is represented by

$$\begin{aligned}\boldsymbol{M}_{i,3}\big(\widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}}\big) &= \mathbb{E}_{\boldsymbol{x}}\left\{y\left[\left(\widehat{\boldsymbol{V}}^T\boldsymbol{P}_i\boldsymbol{x}\right)^{\otimes 3} - \left(\widehat{\boldsymbol{V}}^T\boldsymbol{P}_i\boldsymbol{x}\right)\widetilde{\otimes}\boldsymbol{I}\right]\right\} \\ &= \sum_{j=1}^K \psi_3\left(\widehat{\boldsymbol{V}}^T\boldsymbol{w}_j^*\right) \cdot \left(\widehat{\boldsymbol{V}}^T\overline{\boldsymbol{w}}_j^*\right)^{\otimes 3} \quad (12)\end{aligned}$$

and can be estimated by training samples as well. Next, one can decompose the estimate $\widehat{\boldsymbol{M}}_{i,3}(\widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}})$ to obtain unit vectors $\{\widehat{\boldsymbol{u}}_j\}_{j=1}^K \in \mathbb{R}^K$. Since $\overline{\boldsymbol{w}}^*$ lies in the subspace $\boldsymbol{V}$, we have $\boldsymbol{V}\boldsymbol{V}^T\overline{\boldsymbol{w}}_j^* = \overline{\boldsymbol{w}}_j^*$. Then, $\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j$ is an estimate of $s_j\overline{\boldsymbol{w}}_j^*$. The initialization process is summarized in Subroutine 1.

Subroutine 1 Tensor Initialization Method

```
Input: training data $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$;
Partition $\mathcal{D}$ into three disjoint subsets $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$;
Calculate $\widehat{\boldsymbol{M}}_{i,1}$, $\widehat{\boldsymbol{M}}_{i,2}$ following (5), (6) using $\mathcal{D}_1$, $\mathcal{D}_2$, respectively;
Obtain the estimate subspace $\widehat{\boldsymbol{V}}$ of $\widehat{M}_{i,2}$;
Calculate $\widehat{\boldsymbol{M}}_{i,3}(\widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}})$ using (12) through $\mathcal{D}_3$;
Obtain $\{\widehat{\boldsymbol{u}}_j\}_{j=1}^K$ via tensor decomposition method [17];
Obtain $\widehat{\boldsymbol{\alpha}}_1$, $\widehat{\boldsymbol{\alpha}}_2$ by solving optimization problem (11);
Return: $\boldsymbol{w}_j^{(0)} = \operatorname{sign}(\widehat{\alpha}_{2,j})\widehat{\alpha}_{1,j}\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j$ and $\hat{\boldsymbol{v}} = \operatorname{sign}(\widehat{\boldsymbol{\alpha}}_2)$, $j = 1, \ldots, K$.
```

B. Parameter Estimation Through Accelerated Gradient Descent

In this part, we provide the major theoretical results. Lemma 1 provides the first error bound of the initialization using the tensor initialization method in the presence of noise. Based on the tensor initialization method, Theorem 1 summarizes the recovery accuracy of $\boldsymbol{W}^*$ using Algorithm 1.

Lemma 1: Assume the noise level $|\xi| \leq KM\sigma_1$ and the number of samples $N \geq C_1\kappa^8M^2Kd\log^4 d$ for some large positive constant C_1, the tensor initialization method in Subroutine 1 outputs $\hat{\boldsymbol{v}}$, $\boldsymbol{W}^{(0)}$ such that

$$\hat{\boldsymbol{v}} = \boldsymbol{v}^* \quad (13)$$

and

$$\|\boldsymbol{W}^{(0)} - \boldsymbol{W}^*\|_2 \leq C_2\kappa^6\sqrt{\frac{K^4d\log d}{N}}(KM\sigma_1 + |\xi|) \quad (14)$$

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

Theorem 1: Let $\{\boldsymbol{W}^{(t)}\}_{t=1}^{T}$ be the sequence generated in Algorithm 1 with $\eta = (1/(12M^2K))$. Suppose the noise level $|\xi| \leq KM\sigma_1$ and the number of samples satisfies

$$N \geq C_3\varepsilon_0^{-2}\kappa^9\gamma^3 M^3 K^8 d\log^4 d\log(1/\varepsilon) \tag{15}$$

for some constants $C_3 > 0$ and $\varepsilon_0 \in (0, (1/2))$. Then, $\{\boldsymbol{W}^{(t)}\}_{t=1}^{T}$ converges linearly to $\boldsymbol{W}^*$ with probability at least $1 - K^2M^2T \cdot d^{-10}$ as

$$\|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\|_2 \leq \nu(\beta)^t \|\boldsymbol{W}^{(0)} - \boldsymbol{W}^*\|_2 + C_4\sqrt{\frac{\kappa^2\gamma MK^2 d\log d}{N}} \cdot |\xi| \tag{16}$$

and

$$\|\boldsymbol{W}^{(T)} - \boldsymbol{W}^*\|_2 \leq \varepsilon\|\boldsymbol{W}^*\|_2 + C_4\sqrt{\frac{\kappa^2\gamma MK^2 d\log d}{N}} \cdot |\xi| \tag{17}$$

where $\nu(\beta)$ is the convergence rate that depends on β, and C_4 is some positive constant. Moreover, we have

$$\nu(\beta) < \nu(0) \quad \text{for some small nonzero } \beta. \tag{18}$$

Specifically, let $\beta^* = (1 - (((1-\varepsilon_0)/(132\kappa^2\gamma KM)))^{1/2})^2$, we have

$$\begin{aligned} 1 - \frac{1-\varepsilon_0}{132\kappa^2\gamma KM} \leq \nu(0) &\leq 1 - \frac{1-2\varepsilon_0}{132\kappa^2\gamma KM} \\ \nu(\beta^*) &\leq 1 - \frac{1-\varepsilon_0}{\sqrt{132\kappa^2\gamma KM}}. \end{aligned} \tag{19}$$

Remark 1 (Zero Generalization Error of Learned Model): Lemma 1 shows that the weight vector $\boldsymbol{v}^*$ of the second layer can be exactly recovered when the noise is bounded, and there exist enough samples. Theorem 1 shows that the iterates returned by Algorithm 1 converge to $\boldsymbol{W}^*$ exactly in the noiseless case or approximately in noisy case. For the convenience of presentation, we refer to the second term on the right-hand side of (16) and (17) as the noise error term. Specifically, when the relation of input $\boldsymbol{x}$ and the output y can be exactly described by the CNN model, i.e., the noise $\xi = 0$, then the noise error term vanishes, and the ground-truth $\boldsymbol{W}^*$ can be estimated exactly with a finite number of samples. When the noise is not zero, the noise error term decreases as the number of samples N increases in the order of $(1/N)^{1/2}$. With a sufficiently large sample size, iterates can approach $\boldsymbol{W}^*$ for an arbitrarily small error. With the number of samples satisfies (15), the second error term on the right-hand side of (16) is proportional to the noise magnitude $|\xi|$. From the definition of $g(\cdot)$, one can check that $\kappa KM\sigma_1 \leq \mathbb{E}_{\boldsymbol{x}}|g(\boldsymbol{x})| \leq KM\sigma_1$ when $\boldsymbol{x}$ follows $\mathcal{N}(0, 1)$. Then, the condition in Lemma 1 and Theorem 1 that $|\xi| \leq KM\sigma_1$ means that the noise can be as high as the order of the average energy of the noiseless output $g(\boldsymbol{x})$.

Remark 2 (Faster Linear Convergence Rate Than GD in Learning Neural Networks): Theorem 1 indicates that the heavy ball step can accelerate the rate of convergence, as shown in (18). Without the second momentum term, i.e., $\beta = 0$, the rate of convergence is $1 - \Theta(1/(KM))$ for the vanilla GD. If β is selected appropriately, the rate of convergence is improved and upper bounded by $1 - \Theta(1/(KM)^{1/2})$. This is the first article to provide theoretical guarantees for the convergence of AGD methods in learning neural networks.

Remark 3 (Sample Complexity Analysis): Theorem 1 requires $O(M^3K^8 d\log^4 d\log(1/\varepsilon))$ number of samples for the successful estimation. K is the number of nodes in the hidden layer and, usually, a fixed constant for a given neural network. d is the dimension of patches and scales with the size of input data. ε is the estimation error of $\boldsymbol{W}^*$. Note that the degree of freedom of $\boldsymbol{W}^*$ is Kd. The required number of samples in Theorem 1 depends on $d\log^4 d$ and thus is nearly optimal with respect to d.

C. Comparisons With Related Works

We compare our results with all the exiting works to the best of our knowledge that provide generalizability guarantees. We focus on the following three aspects.

1) Tensor Initialization Method and AGD Algorithm: Tensor initialization method is first introduced and analyzed in [46] for fully connected neural networks with homogeneous activation functions. Reference [9] extends the analysis to the nonhomogeneous sigmoid activation. However, both works only consider noiseless settings. When reduced to the case of fully connected neural networks without noise, i.e., $\xi = 0$ and $M = 1$, the bound in (14) is as tight as that in [46].

Existing works only consider the convergence of GD instead of AGD in neural networks. Due to the additional momentum term, the analysis of GD does not directly generalize to AGD. Specifically, the convergence of GD is based on establishing $\|\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^*\|_2 \leq \nu\|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\|_2$ for some $|\nu| < 1$, and therefore, this analysis does not directly apply to AGD. Instead, our analysis of AGD is based on the augmented iteration as $\begin{bmatrix}\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^* \\ \boldsymbol{W}^{(t)} - \boldsymbol{W}^*\end{bmatrix}$, and the convergence rate is calculated as a function of β. Note our analysis also applies to the special case that $\beta = 0$, i.e., the GD algorithm.

2) Noisy Outputs: References [10] and [44] consider noisy outputs are fully connected neural networks. Ge *et al.* [10] analyze stochastic GD through expectation, and the noise is assumed to be zero mean. Thus, the noise level does not appear in the theoretical bounds. Zhang *et al.* [44] assume the existence of a proper initialization, but there is no theoretical guarantee in [44] about whether their proposed initialization method in the noisy setting can return a desirable initialization. Moreover, our error bound (16) is tighter than that in [44]. Specifically, the second term on the right-hand side of (16) only depends on the noise factor ξ. In contrast, [44, eq. (4.1)] shows that the GD algorithm converges to $\boldsymbol{W}^*$ up to an estimation error that depends on both $\|\boldsymbol{W}^*\|_F$ and the noise level. Even when there is no noise, the additional error term in [44, eq. (4.1)] is nonzero.

3) Theoretical Guarantees: As most existing works only focus on the GD algorithm with noiseless outputs, we compare with these works by reducing to $\beta = 0$ and $\xi = 0$ in Theorem 1. References [3], [8], [9], and [45] consider one-hidden-layer nonoverlapping convolutional neural networks. References [3] and [8] show that the GD algorithm converges to the ground truth with a constant probability from one random initialization, but the result only applies to the case of one node in the hidden layer, i.e., $K = 1$. Moreover, the analyses assume an infinite number of input samples and do not consider the sample complexity. Based on the tensor initialization method [46], [9], and [45] show that the GD algorithm converges to the ground truth with a linear convergence rate, but the result only applies to smooth activation functions, such as sigmoid functions and excludes

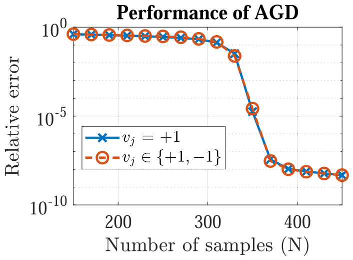


Fig. 2. Recovery error of AGD under different $\boldsymbol{v}^*$.

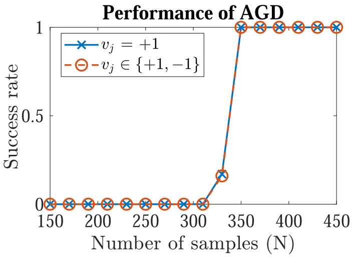


Fig. 3. Success rate of AGD under different $\boldsymbol{v}^*$.

ReLU functions. References [10] and [44] provide the sample complexity analysis with the ReLU activation function but focus on one-hidden-layer fully connected neural networks, which can be viewed as a special case of the convolutional neural network studied in this article by selecting $M = 1$. The sample complexity in [10] with respect to d is poly(d), but the power of d is not provided explicitly. Moreover, the convergence rate in [10] is sublinear, while our theorem shows that both GD and AGD enjoy linear convergence rates.

IV. Simulation

The input data $\{\boldsymbol{x}_n\}_{n=1}^N$ are randomly selected from the Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$. The number of patches M is selected as a factor of the signal dimension p, and all the patches have the same size d with $d = p/M$. Entries of the weight matrix $\boldsymbol{W}^*$ are i.i.d generated from $\mathcal{N}(0, 1^2)$. The noise $\{\xi_n\}_{n=1}^N$ are i.i.d from $\mathcal{N}(0, \sigma^2)$, and the noise level is measured by σ/E_y, where E_y is the average energy of the noiseless outputs $\{g(\boldsymbol{x}_n)\}_{n=1}^N$ calculated as $E_y = ((1/N)\sum_{n=1}^N |g(\boldsymbol{x}_n)|^2)^{1/2}$. The output data $\{y_n\}_{n=1}^N$ are generated by (1). In the following numerical experiments, the whole data set $\{\boldsymbol{x}_n, y_n\}_{n=1}^N$ instead of a fresh subset is used to calculate the gradient in each iteration. The initialization is randomly selected from $\{\boldsymbol{W}_0 \big| \|\boldsymbol{W}_0 - \boldsymbol{W}^*\|_F / \|\boldsymbol{W}^*\|_F < 0.5\}$ and $\boldsymbol{v}^{(0)} = \boldsymbol{v}^*$ to reduce the computation. As shown in [9] and [44], random initialization and the tensor method have very similar numerical performance.

If not otherwise specified, we use the following parameter setup. p is chosen as 50, and M is selected as 5. Hence, $d = p/M$ is 10. The number of nodes in hidden layer K is chosen as 5. The number of samples N is chosen as 200. The step size of the gradient η is $((2K)/M^2)$, and β is selected as $(1 - (1/(KM)^{1/2}))^2$. All the simulations are implemented in MATLAB 2015a on a desktop with 3.4-GHz Intel Core i7.

A. Performance of AGD With Different $\boldsymbol{v}^*$

Figs. 2 and 3 show the performance of AGD with different v_j^*, and the results are averaged over 100 independent trials. In Fig. 2, the relative error is defined as $\|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\|_F / \|\boldsymbol{W}^*\|_F$, where $\boldsymbol{W}^{(t)}$ is the estimate in the tth iteration. In Fig. 3, each trial is called a success if the relative error is less than 10^{-6}. We generate two cases of $\boldsymbol{v}^*$. In Case 1, all the entries of $\boldsymbol{v}^*$ are 1, while each entry is i.i.d. selected from $\{+1, -1\}$ with equal probability in Case 2. k is set as 5, and d is set as 60 with $p = 300$. In Figs. 2 and 3, the results of Case 1 is shown by the lines marked as "$v_j = +1$," and the second group is marked as "$v_j \in \{+1, -1\}$." We can see that the performances of these two cases are almost the same. In the following experiments, we fix $\boldsymbol{v}_j^*$ as 1 for all j.

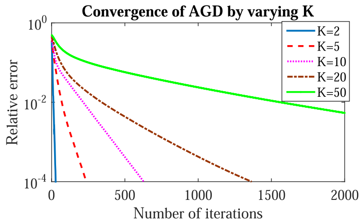


Fig. 4. Convergence of AGD with different K.

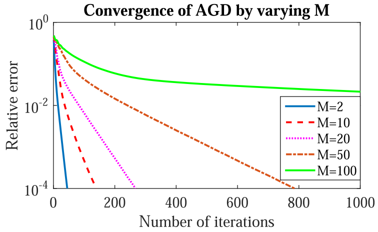


Fig. 5. Convergence of AGD with different M.

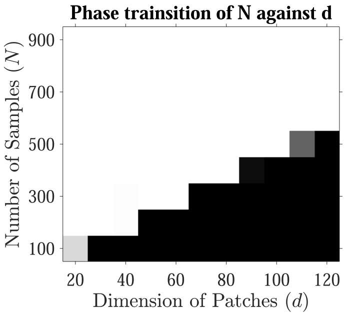


Fig. 6. Phrase transition of N against d.

B. Performance of AGD With Noiseless Output

Figs. 4 and 5 show the convergence of AGD by varying K and M. In Fig. 4, η and β are calculated based the value of K, and other parameters are fixed. For each K, we conducted independent trials with random selected $\boldsymbol{x}_n$, $\boldsymbol{W}^*$ and the corresponding y_n. Given K, the convergence rates of different trials vary slightly. Fig. 4 shows one example of these trials for each K. We can see that the convergence rate decreases as K increases. Similarly, Fig. 5 shows that the convergence rate decreases as M increases.

Figs. 6 and 7 show the phase transition, where the number of samples N, the dimension of patches d, and the number of nodes in the hidden layer K change. All the other parameters except N and d (or k) remain the same as the default values. For each (N, d) or (N, K) pair, we conduct 100 independent trials. Each trial is called a success if the relative error is less than 10^{-6}. A white block means all the trails are successful, while a black one means all the trials fail.

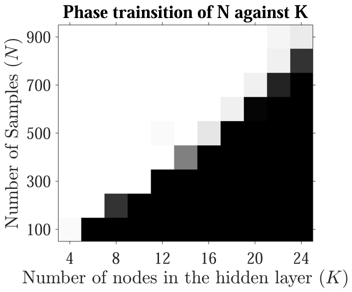


Fig. 7. Phrase transition of N against K.

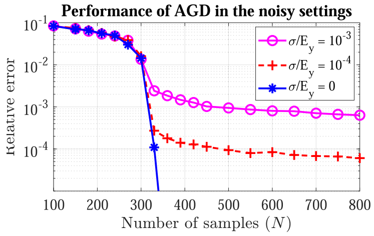


Fig. 8. Performance of Algorithm 1 with noisy measurements.

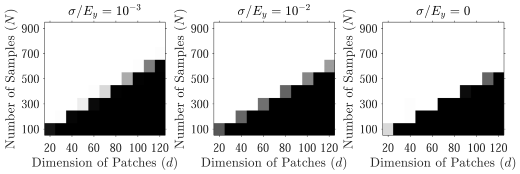


Fig. 9. Phrase transition of AGD in noisy settings.

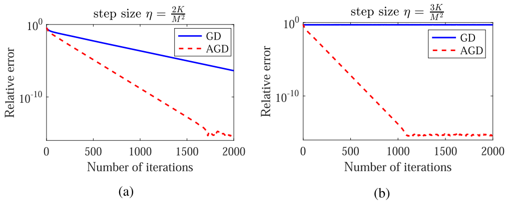


Fig. 10. Performance of AGD and GD under different η.

C. Performance of AGD With Noisy Output

Fig. 8 shows the relative error of the AGD algorithm by varying the number of samples N in the noisy case. K is set as 5, and d is set as 60 with $p = 300$. Hence, the degree of freedom of $\boldsymbol{W}^*$ is 300. The y-axis stands for the relative error, and the results are averaging over 100 independent trials. We can see that the relative errors are high when N is less than the degree of freedom at 300. Once the number of samples exceeds the degree of freedom, the relative error decreases dramatically in both noisy and noiseless settings. As N increases, the relative error in the noisy setting converges fast to the noise level.

Fig. 9 shows the phrase transition of N against d with different noise levels. A trial is considered successful if the returned $\boldsymbol{W}$ satisfies $\|\boldsymbol{W} - \boldsymbol{W}^*\|_2/\|\boldsymbol{W}^*\|_2 \leq \sigma/E_y$ (or 10^{-6} in noiseless settings). As d increases, the required number of samples for all successful estimations increases as well. In addition, with a higher noise level, the success region becomes smaller.

D. Comparison of GD and AGD

Fig. 10 shows the progress of both the GD and AGD methods across iterations. We fix the same initialization for GD and AGD in Fig. 10(a) and (b), respectively. In both cases, β and other parameters except for η are fixed as the default values. The only difference is that the step size η is $((2K)/(M^2))$ in Fig. 10(a) and $((3K)/(M^2))$ in Fig. 10(b). One can see that starting from the same initialization, GD sometimes diverges in (b) with a large step size. By adding the heavy-ball term, the AGD method can converge to the global minimum. Moreover, when both GD and AGD converge, AGD converges faster than GD.

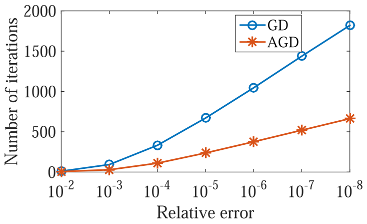


Fig. 11. Comparison of AGD and GD in number of iterations.

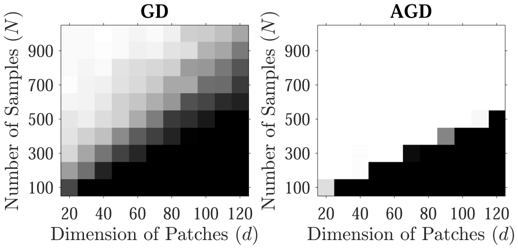


Fig. 12. Phase transition of GD and AGD.

Fig. 11 compares the convergence rates of AGD and GD. The number of samples N is set as 500, and other parameters are the default values. Each point means the smallest number of iterations needed to reach the corresponding estimation error, and the results are averaged over 100 independent trials. AGD requires a smaller number of the iterations than GD to achieve the same relative error.

Fig. 12 shows the phrase transition of GD and AGD by varying N and d when the output is noiseless. AGD has a larger successful region than GD, so that AGD requires a smaller number of samples to guarantee successful recovery for a given d.

V. Conclusion

We have analyzed the performance of (accelerated) GD methods in learning one-hidden-layer nonoverlapping convolutional neural networks with multiple nodes and the ReLU activation function. We have shown that if the number of samples exceeds our provided sample complexity, GD methods with the tensor initialization find the ground-truth parameters with a linear convergence rate. The parameters can be estimated exactly when the data are noiseless. Moreover, accelerated GD is proved to converge faster than vanilla GD. One future direction is to extend the analysis framework to multilayer overlapping convolutional neural networks.

Appendix

A. Proof of Theorem 1

We first summarize the high-level ideas in proving Theorem 1 before presenting the technical proof. Following the recent line of research, such as [45] and [46], the idea is to initialize the weights $\boldsymbol{W}$ near the ground-truth $\boldsymbol{W}^*$ and then gradually converge to it. Our initialization is similar to [46], as discussed in Section III-A. However, our proof is more involved than that of [46] to handle the additional noise item, the nonsmooth ReLU functions, the additional momentum term in accelerated gradient descent, and different neural network structures.

As for the convergence analysis, [45] and [46] apply the intermediate value theorem over $\nabla \hat{f}_{\mathcal{D}_t}$ at each iterate $\boldsymbol{W}_t$ as

$$\nabla \hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)}) \simeq \langle \nabla^2 \hat{f}_{\mathcal{D}_t}(\widehat{\boldsymbol{W}}^{(t)}), \boldsymbol{W}^{(t)} - \boldsymbol{W}^* \rangle$$

for some $\widehat{\boldsymbol{W}}^{(t)}$ between $\boldsymbol{W}^{(t)}$ and $\boldsymbol{W}^*$ and analyze $\nabla^2 \hat{f}_{\mathcal{D}_t}$ to obtain a recursive inequality of $\boldsymbol{W}^{(t)} - \boldsymbol{W}^*$ over t. The intermediate value theorem only applies to the continuous functions, and their analyses do not extend to our setup because with the ReLU activation function, the resulting $\nabla \hat{f}_{\mathcal{D}}$ is noncontinuous. Instead, we will first prove that the population loss function f, which is defined as

$$\begin{aligned} f(\boldsymbol{W}) &:= \mathbb{E}_{\mathcal{D}_t} \hat{f}_{\mathcal{D}_t}(\boldsymbol{W}) \\ &= \mathbb{E}_{\boldsymbol{x}} \left(\frac{1}{K} \sum_{j=1}^{K} \sum_{i=1}^{M} \phi(\boldsymbol{w}_j^T \boldsymbol{P}_i \boldsymbol{x}) - y \right)^2 \end{aligned} \tag{20}$$

is locally convex near $\boldsymbol{W}^*$, and the gradient of $\hat{f}_{\mathcal{D}_t}$ is close enough to ∇f. We will then show that the iterates based on $\nabla \hat{f}_{\mathcal{D}_t}$ converge to $\boldsymbol{W}^*$.

The following two lemmas are important for our proof. We leave their proofs to Appendices C and D.

Lemma 2: For any $\boldsymbol{W}$ that satisfies

$$\|\boldsymbol{W} - \boldsymbol{W}^*\|_2 \leq \frac{\varepsilon_0 \sigma_K}{44\kappa^2 \gamma M}, \tag{21}$$

we have

$$\frac{(1-\varepsilon_0)M}{11\kappa^2\gamma} \boldsymbol{I} \leq \nabla^2 f(\boldsymbol{W}) \leq 6M^2 K \boldsymbol{I}. \tag{22}$$

Lemma 3: Suppose a fixed point $\boldsymbol{W}$ satisfies (21). Then, for a training set $\mathcal{D}$ with $N > d \log d$ samples, we have

$$\begin{aligned} &\|\nabla f(\boldsymbol{W}) - \nabla \hat{f}_{\mathcal{D}}(\boldsymbol{W})\|_2 \\ &\quad \lesssim MK \sqrt{\frac{d \log d}{N}} (MK \|\boldsymbol{W} - \boldsymbol{W}^*\|_2 + |\xi|) \end{aligned} \tag{23}$$

with probability at least $1 - K^2 M^2 \cdot d^{-10}$.

Lemma 2 shows that the population loss function $f(\boldsymbol{W})$ is locally convex near $\boldsymbol{W}^*$. Then, the analysis of the AGD algorithm over the empirical loss function $\hat{f}_{\mathcal{D}}(\boldsymbol{W})$ is based on the analysis over $f(\boldsymbol{W})$ and the error bound between $\nabla \hat{f}_{\mathcal{D}}(\boldsymbol{W})$ and $\nabla f(\boldsymbol{W})$ as shown in (26).

Lemma 3 describes the error bound between $\nabla f(\boldsymbol{W})$ and $\nabla \hat{f}_{\mathcal{D}}(\boldsymbol{W})$, and (23) shows that $\nabla \hat{f}_{\mathcal{D}}(\boldsymbol{W})$ converges to $\nabla f(\boldsymbol{W})$ in a small neighborhood of $\boldsymbol{W}^*$ when N is large enough. A similar result is stated in [44, Lemma 5.3] for fully connected neural networks with the ReLU activation function. Fully connected neural networks can be viewed as a special kind of convolutional neural networks with $M = 1$. Moreover, even when reducing our model to the case $M = 1$, the error bound presented in (23) is much tighter than that in [44, Lemma 5.3].

Combining Lemmas 1–3, we will show the convergence of GD in solving (3) by mathematical induction. Conditioned on the assumption that $\boldsymbol{W}^{(t)}$ satisfies (21), we show that $\|\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^*\|_2$ is related to $\|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\|_2$ by (38). The acceleration of heavy-ball steps is analyzed through (32), and the result is summarized in (33). The next step is to show (38) holds for all $0 \leq t \leq T-1$. By Lemma 1, we can choose N to be large enough so that $\boldsymbol{W}^{(0)}$ satisfies (21). Then, in the induction step, with a large enough N and a bounded ξ, we will show that $\|\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^*\|_2 < \|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\|_2$. Then $\boldsymbol{W}^{(t)}$ satisfies (21) naturally. The details are as follows.

Proof of Theorem 1: The update rule of $\boldsymbol{W}^{(t)}$ is

$$\begin{aligned} \boldsymbol{W}^{(t+1)} &= \boldsymbol{W}^{(t)} - \eta \nabla \hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)}) + \beta(\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t-1)}) \\ &= \boldsymbol{W}^{(t)} - \eta \nabla f(\boldsymbol{W}^{(t)}) + \beta(\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t-1)}) \\ &\quad + \eta(\nabla f(\boldsymbol{W}^{(t)}) - \nabla \hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)})). \end{aligned} \tag{24}$$

Since $\nabla^2 f$ is a smooth function, by the intermediate value theorem, we have

$$\begin{aligned} \boldsymbol{W}^{(t+1)} &= \boldsymbol{W}^{(t)} - \eta \nabla^2 f(\widehat{\boldsymbol{W}}^{(t)})(\boldsymbol{W}^{(t)} - \boldsymbol{W}^*) \\ &\quad + \beta(\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t-1)}) \\ &\quad + \eta(\nabla f(\boldsymbol{W}^{(t)}) - \nabla \hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)})) \end{aligned} \tag{25}$$

where $\widehat{\boldsymbol{W}}^{(t)}$ lies in the convex hull of $\boldsymbol{W}^{(t)}$ and $\boldsymbol{W}^*$.

Next, we have

$$\begin{aligned} &\begin{bmatrix} \boldsymbol{W}^{(t+1)} - \boldsymbol{W}^* \\ \boldsymbol{W}^{(t)} - \boldsymbol{W}^* \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{I} - \eta \nabla^2 f(\widehat{\boldsymbol{W}}^{(t)}) + \beta \boldsymbol{I} & \beta \boldsymbol{I} \\ \boldsymbol{I} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{W}^{(t)} - \boldsymbol{W}^* \\ \boldsymbol{W}^{(t-1)} - \boldsymbol{W}^* \end{bmatrix} \\ &\quad + \eta \begin{bmatrix} \nabla f(\boldsymbol{W}^{(t)}) - \nabla \hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)}) \\ 0 \end{bmatrix}. \end{aligned} \tag{26}$$

Let $\boldsymbol{A}(\beta) = \begin{bmatrix} \boldsymbol{I} - \eta \nabla^2 f(\widehat{\boldsymbol{W}}^{(t)}) + \beta \boldsymbol{I} & \beta \boldsymbol{I} \\ \boldsymbol{I} & 0 \end{bmatrix}$, and therefore, we have

$$\begin{aligned} \left\| \begin{bmatrix} \boldsymbol{W}^{(t+1)} - \boldsymbol{W}^* \\ \boldsymbol{W}^{(t)} - \boldsymbol{W}^* \end{bmatrix} \right\|_2 &= \|\boldsymbol{A}(\beta)\|_2 \left\| \begin{bmatrix} \boldsymbol{W}^{(t)} - \boldsymbol{W}^* \\ \boldsymbol{W}^{(t-1)} - \boldsymbol{W}^* \end{bmatrix} \right\|_2 \\ &\quad + \eta \left\| \begin{bmatrix} \nabla f(\boldsymbol{W}^{(t)}) - \nabla \hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)}) \\ 0 \end{bmatrix} \right\|_2. \end{aligned}$$

From Lemma 3, we know that

$$\begin{aligned} &\eta \|\nabla f(\boldsymbol{W}^{(t)}) - \nabla \hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)})\|_2 \\ &\quad \leq C_5 \eta M^2 \sqrt{\frac{d \log d}{N_t}} \left(\|\boldsymbol{W} - \boldsymbol{W}^*\|_2 + \frac{|\xi|}{M} \right) \end{aligned} \tag{27}$$

for some constant $C_5 > 0$. Then, we have

$$\begin{aligned} &\|\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^*\|_2 \\ &\leq \left(\|\boldsymbol{A}(\beta)\|_2 + C_5 \eta M^2 \sqrt{\frac{d \log d}{N_t}} \right) \|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\|_2 \\ &\quad + C_5 \eta M \sqrt{\frac{d \log d}{N_t}} |\xi| \\ &:= \nu(\beta) \|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\|_2 + C_5 \eta M \sqrt{\frac{d \log d}{N_t}} |\xi|. \end{aligned} \tag{28}$$

Let $\nabla^2 f(\widehat{\boldsymbol{W}}^{(t)}) = \boldsymbol{S} \Lambda \boldsymbol{S}^T$ be the eigendecomposition of $\nabla^2 f(\widehat{\boldsymbol{W}}^{(t)})$. Then, we define

$$\begin{aligned} \widetilde{\boldsymbol{A}}(\beta) &:= \begin{bmatrix} \boldsymbol{S}^T & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S}^T \end{bmatrix} \boldsymbol{A}(\beta) \begin{bmatrix} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S} \end{bmatrix} \\ &= \begin{bmatrix} \boldsymbol{I} - \eta \Lambda + \beta \boldsymbol{I} & \beta \boldsymbol{I} \\ \boldsymbol{I} & 0 \end{bmatrix}. \end{aligned} \tag{29}$$

Since $\begin{bmatrix} \boldsymbol{S} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S} \end{bmatrix}\begin{bmatrix} \boldsymbol{S}^T & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{S}^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I} \end{bmatrix}$, we know $\boldsymbol{A}(\beta)$ and $\widetilde{\boldsymbol{A}}(\beta)$ share the same eigenvalues. Let λ_i be the ith eigenvalue of $\nabla^2 f(\widehat{\boldsymbol{W}}^{(t)})$, then the corresponding ith eigenvalue of $\boldsymbol{A}(\beta)$, denoted by $\delta_i(\beta)$, satisfies

$$\delta_i^2 - (1 - \eta\lambda_i + \beta)\delta_i + \beta = 0. \tag{30}$$

Then, we have

$$\delta_i(\beta) = \frac{(1 - \eta\lambda_i + \beta) + \sqrt{(1 - \eta\lambda_i + \beta)^2 - 4\beta}}{2} \tag{31}$$

and

$$|\delta_i(\beta)| = \begin{cases} \sqrt{\beta}, & \text{if } \beta \geq \left(1 - \sqrt{\eta\lambda_i}\right)^2 \\ \frac{1}{2}\left|(1-\eta\lambda_i+\beta)+\sqrt{(1-\eta\lambda_i + \beta)^2 - 4\beta}\right|, & \text{otherwise.} \end{cases} \tag{32}$$

Note that the other root of (30) is abandoned because the root in (31) is always no less than the other root with $|1-\eta\lambda_i| < 1$. By simple calculations, we have

$$\delta_i(0) > \delta_i(\beta), \quad \text{for} \quad \forall \beta \in \left(0, (1 - \eta\lambda_i)^2\right). \tag{33}$$

Moreover, δ_i achieves the minimum $\delta_i^* = |1 - (\eta\lambda_i)^{1/2}|$ when $\beta = (1 - (\eta\lambda_i)^{1/2})^2$.

Let us first assume $\boldsymbol{W}^{(t)}$ satisfies (21), then from Lemma 2, we know that

$$0 < \frac{(1 - \varepsilon_0)M}{11\kappa^2\gamma} \leq \lambda_i \leq 6M^2K.$$

Let $\gamma_1 = (((1 - \varepsilon_0)M)/(11\kappa^2\gamma))$ and $\gamma_2 = 6KM^2$. If we choose β such that

$$\beta^* = \max\left\{\left(1 - \sqrt{\eta\gamma_1}\right)^2, \left(1 - \sqrt{\eta\gamma_2}\right)^2\right\} \tag{34}$$

then we have $\beta \geq (1 - (\eta\lambda_i)^{1/2})^2$ and $\delta_i = \max\{|1 - (\eta\gamma_1)^{1/2}|, |1 - (\eta\gamma_2)^{1/2}|\}$ for any i.

Let $\eta = (1/(2\gamma_2))$, then β^* equals to $(1 - (\gamma_1/(2\gamma_2))^{1/2})^2$. Then, for any $\varepsilon_0 \in (0, 1/2)$, we have

$$\begin{aligned} \left\|\boldsymbol{A}\left(\beta^*\right)\right\|_2 &= \max_i \delta_i\left(\beta^*\right) = 1 - \sqrt{\frac{\gamma_1}{2\gamma_2}} \\ &= 1 - \sqrt{\frac{1 - \varepsilon_0}{132\kappa^2\gamma KM}} \\ &\leq 1 - \frac{1 - (3/4) \cdot \varepsilon_0}{\sqrt{132\kappa^2\gamma KM}}. \end{aligned} \tag{35}$$

Then, let

$$C_5\eta M^2\sqrt{\frac{d \log d}{N_t}} \leq \frac{\varepsilon_0}{4\sqrt{132\kappa^2\gamma KM}} \tag{36}$$

we need $N_t \gtrsim \varepsilon_0^{-2}\kappa^2\gamma MK^3 d \log d$. Combining (35) and (36), we have

$$\nu\left(\beta^*\right) \leq 1 - \frac{1 - \varepsilon_0}{\sqrt{132\kappa^2\gamma KM}}. \tag{37}$$

Let $\beta = 0$, we have

$$\nu(0) \geq \|\boldsymbol{A}(0)\|_2 = 1 - \frac{1 - \varepsilon_0}{132\kappa^2\gamma KM}$$

$$\nu(0) \leq \|\boldsymbol{A}(0)\|_2 + C_5\eta M^2\sqrt{\frac{d \log d}{N_t}} \leq 1 - \frac{1 - 2\varepsilon_0}{132\kappa^2\gamma KM}$$

if $N_t \gtrsim \varepsilon_0^{-2}\kappa^2\gamma M^2K^4 d \log d$.

Hence, with $\eta = (1/(2\gamma_2))$ and $\beta = (1 - (\gamma_1/(2\gamma_2)))^2$, we have

$$\begin{aligned} \left\|\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^*\right\|_2 \leq &\left(1 - \frac{1 - \varepsilon_0}{\sqrt{132\kappa^2\gamma KM}}\right)\left\|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\right\|_2 \\ &+ 2C\eta M\sqrt{\frac{d \log d}{N_t}}|\xi| \end{aligned} \tag{38}$$

provided that $\boldsymbol{W}^{(t)}$ satisfies (21), and

$$N_t \gtrsim \varepsilon_0^{-2}\kappa^2\gamma MK^3 d \log d. \tag{39}$$

Then, we can start mathematical induction of (38) over t.

Base Case: According to Lemma 1, we know that (21) holds for $\boldsymbol{W}^{(0)}$ if

$$N \gtrsim \varepsilon_0^{-2}\kappa^9\gamma^2K^8M^2 d \log^4 d. \tag{40}$$

According to (15) in Theorem 1, it is clear that the number of samples N satisfies (40), then (21) indeed holds for $t = 0$. Since (21) holds for $t = 0$ and N in (15) satisfies (39) as well, we have (38) holds for $t = 0$.

Induction Step: Assuming (38) holds for $\boldsymbol{W}^{(t)}$, we need to show that (38) holds for $\boldsymbol{W}^{(t+1)}$. In other words, we need: 1) N satisfies (39) and 2) (21) holds for $\boldsymbol{W}^{(t+1)}$. The requirement 1) holds naturally from (15). To guarantee 2) holds, we need

$$\eta M\sqrt{\frac{d \log d}{N_t}} \lesssim \frac{1 - \varepsilon_0}{\sqrt{132\kappa^2\gamma KM}} \cdot \frac{\varepsilon_0\sigma_K}{44\kappa^2\gamma K^2M}. \tag{41}$$

That requires

$$N_t \gtrsim \varepsilon_0^{-2}\kappa^8\gamma^3M^3K^6 d \log d. \tag{42}$$

Therefore, when $N_t \gtrsim \varepsilon_0^{-2}\kappa^9\gamma^3M^3K^8 d \log^4 d$, we know that (38) holds for all $0 \leq t \leq T - 1$ with probability at least $1 - K^2M^2T \cdot d^{-10}$. By simple calculations, we can obtain

$$\begin{aligned} \left\|\boldsymbol{W}^{(T)} - \boldsymbol{W}^*\right\|_2 \leq &\left(1 - \frac{1 - \varepsilon_0}{\sqrt{132\kappa^2\gamma KM}}\right)^T \left\|\boldsymbol{W}^{(0)} - \boldsymbol{W}^*\right\|_2 \\ &+ C_4\sqrt{\frac{\kappa^2\gamma MK^2 d \log d}{N_t}} \cdot |\xi| \end{aligned} \tag{43}$$

for some constant $C_4 > 0$. □

B. Proof of Lemma 1

The proof of Lemma 1 is divided into three major parts to bound I_1, I_2, and I_3 in (50). Lemmas 4, 5, and 6 provide the error bounds for I_1, I_2, and I_3, respectively. Compared with the proof of [46, Th. 5.6], which considers noiseless measurements, we need to handle additional items corresponding with noise, and the error bounds for these items are obtained by applying matrix concentration inequalities shown in Lemma 7. The detailed proofs of Lemmas 4–6 can be found in the Supplementary Materials.

Lemma 4: Suppose $\boldsymbol{M}_{i,2}$ is defined as in (6) and $\widehat{\boldsymbol{M}}_{i,2}$ is the estimation of $\boldsymbol{M}_{i,2}$ by samples $\mathcal{D} = \{\boldsymbol{x}_n, y_n\}_{n=1}^N$. Then, with probability $1 - d^{-10}$, we have

$$\left\|\widehat{\boldsymbol{M}}_{i,2} - \boldsymbol{M}_{i,2}\right\| \lesssim \sqrt{\frac{d \log d}{N}}(KM\sigma_1 + |\xi|) \tag{44}$$

provided that $N \gtrsim d \log^4 d$.

Lemma 5: Let $\widehat{\boldsymbol{V}}$ be generated by step 4 in Subroutine 1. Suppose $\boldsymbol{M}_{i,3}(\widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}}, \widehat{\boldsymbol{V}})$ is defined as in (12) and

$\widehat{\boldsymbol{M}}_{i,3}(\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}})$ is the estimation of $\boldsymbol{M}_{i,3}(\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}})$ by samples $\mathcal{D}=\{\boldsymbol{x}_n,y_n\}_{n=1}^N$. Furthermore, we assume $\boldsymbol{V}\in\mathbb{R}^{d\times K}$ is an orthogonal basis of $\boldsymbol{W}^*$ and satisfies $\|\boldsymbol{V}\boldsymbol{V}^T-\widehat{\boldsymbol{V}}\widehat{\boldsymbol{V}}^T\|\le 1/4$. Then, provided that $N\gtrsim K^5\log^6 d$, with probability at least $1-d^{-10}$, we have

$$\big\|\widehat{\boldsymbol{M}}_{i,3}\big(\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}}\big)-\boldsymbol{M}_{i,3}\big(\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}}\big)\big\| \lesssim (KM\sigma_1+|\xi|)\sqrt{\frac{K^3\log d}{N}}. \tag{45}$$

Lemma 6: Suppose $\boldsymbol{M}_{i,1}$ is defined as in (5) and $\widehat{\boldsymbol{M}}_{i,1}$ is the estimation of $\boldsymbol{M}_{i,1}$ by samples $\mathcal{D}=\{\boldsymbol{x}_n,y_n\}_{n=1}^N$. Then, with probability $1-d^{-10}$, we have

$$\big\|\widehat{\boldsymbol{M}}_{i,1}-\boldsymbol{M}_{i,1}\big\|\lesssim (KM\sigma_1+|\xi|)\sqrt{\frac{d\log d}{N}} \tag{46}$$

provided that $N\gtrsim d\log^4 d$.

Lemma 7 [37, Th. 1.6]: Consider a finite sequence $\{\boldsymbol{Z}_k\}$ of independent, random matrices with dimensions $d_1\times d_2$. Assume that such random matrix satisfies

$$\mathbb{E}(\boldsymbol{Z}_k)=0 \quad\text{and}\quad \|\boldsymbol{Z}_k\|\le R \quad\text{almost surely.}$$

Define

$$\delta^2:=\max\left\{\left\|\sum_k\mathbb{E}\big(\boldsymbol{Z}_k\boldsymbol{Z}_k^*\big)\right\|,\left\|\sum_k\mathbb{E}\big(\boldsymbol{Z}_k^*\boldsymbol{Z}_k\big)\right\|\right\}.$$

Then, for all $t\ge 0$, we have

$$\operatorname{Prob}\left\{\left\|\sum_k\boldsymbol{Z}_k\right\|\ge t\right\}\le(d_1+d_2)\exp\left(\frac{-t^2/2}{\delta^2+Rt/3}\right).$$

Lemma 8 [46, Lemma E.6]: Let $\boldsymbol{V}\in\mathbb{R}^{d\times K}$ be an orthogonal basis of $\boldsymbol{W}^*$ and $\widehat{\boldsymbol{V}}$ be generated by step 4 in Subroutine 1. Assume $\|\widehat{\boldsymbol{M}}_{i,2}-\boldsymbol{M}_{i,2}\|_2\le\sigma_K(\boldsymbol{M}_{i,2})/10$. Then, for some small ε_0, we have

$$\big\|\boldsymbol{V}\boldsymbol{V}^T-\widehat{\boldsymbol{V}}\widehat{\boldsymbol{V}}^T\big\|_2\le\frac{\big\|\boldsymbol{M}_{i,2}-\widehat{\boldsymbol{M}}_{i,2}\big\|}{\sigma_K\big(\boldsymbol{M}_{i,2}\big)}. \tag{47}$$

Lemma 9 [46, Lemmas E.13 and E.14]: Let $\boldsymbol{V}\in\mathbb{R}^{d\times K}$ be an orthogonal basis of $\boldsymbol{W}^*$ and $\widehat{\boldsymbol{V}}$ be generated by step 4 in Subroutine 1. Assume $\boldsymbol{M}_{i,1}$ can be written in the form of (8) with some constant ϕ_1, and let $\widehat{\boldsymbol{M}}_{i,1}$ be the estimation of $\boldsymbol{M}_{i,1}$ by samples $\mathcal{D}=\{\boldsymbol{x}_n,y_n\}_{n=1}^N$. Let $\widehat{\boldsymbol{\alpha}}_1$ and $\widehat{\boldsymbol{\alpha}}_2$ be the optimal solutions of (11) with $\widehat{\overline{\boldsymbol{w}}}_j=\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j$. Then, for each $j\in\{1,2,\ldots,K\}$, if

$$\begin{aligned}T_1&:=\big\|\boldsymbol{V}\boldsymbol{V}^T-\widehat{\boldsymbol{V}}\widehat{\boldsymbol{V}}^T\big\|_2\le\frac{1}{\kappa^2\sqrt{K}}\\ T_2&:=\big\|\widehat{\boldsymbol{u}}_j-s_j\widehat{\boldsymbol{V}}^T\overline{\boldsymbol{w}}_j\big\|_2\le\frac{1}{\kappa^2\sqrt{K}}\\ T_3&:=\big\|\widehat{\boldsymbol{M}}_{i,1}-\boldsymbol{M}_{i,1}\big\|_2\le\frac{1}{4}\|\boldsymbol{M}_{i,1}\|_2\end{aligned} \tag{48}$$

then we have

$$\begin{aligned}\big|\alpha_{1,j}^*-\widehat{\alpha}_{1,j}\big|&\le\Big(\kappa^4K^{\frac{3}{2}}(T_1+T_2)+\kappa^2K^{\frac{1}{2}}T_3\Big)\big|\alpha_{1,j}^*\big|\\ \big|\alpha_{2,j}^*-\widehat{\alpha}_{2,j}\big|&\le\big(\kappa^8K^3T_2+\kappa^2K^2T_3\big)\big|\alpha_{2,j}^*\big|\end{aligned} \tag{49}$$

where $\alpha_{1,j}^*=s_jv_j^*\|\boldsymbol{w}_j^*\|_2$ and $\alpha_{2,j}^*=v_j^*\|\boldsymbol{w}_j^*\|_2$.

Proof of Lemma 1: we have

$$\begin{aligned}&\big\|\boldsymbol{w}_j^*-s_j\big|\widehat{\alpha}_{1,j}\big|\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j\big\|_2\\ &\le\big\|\boldsymbol{w}_j^*-s_j\big\|\boldsymbol{w}_j\big\|_2\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j+s_j\big\|\boldsymbol{w}_j\big\|_2\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j-s_j\big|\widehat{\alpha}_{1,j}\big|\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j\big\|_2\\ &\le\big\|\boldsymbol{w}_j^*-s_j\big\|\boldsymbol{w}_j\big\|_2\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j\big\|_2+\big\|\big\|\boldsymbol{w}_j\big\|_2\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j-\big|\widehat{\alpha}_{1,j}\big|\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j\big\|_2\\ &\le\big\|\boldsymbol{w}_j^*\big\|_2\big\|\overline{\boldsymbol{w}}_j^*-s_j\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j\big\|_2+\big|\big\|\boldsymbol{w}_j\big\|_2-\big|\widehat{\alpha}_{1,j}\big|\big|\big\|\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j\big\|_2\\ &\le\sigma_1\Big(\Big\|\overline{\boldsymbol{w}}_j^*-\widehat{\boldsymbol{V}}\widehat{\boldsymbol{V}}^T\overline{\boldsymbol{w}}_j^*\Big\|_2+\Big\|\widehat{\boldsymbol{V}}^T\overline{\boldsymbol{w}}_j^*-s_j\widehat{\boldsymbol{u}}_j\Big\|_2\Big)\\ &\quad+\big|\big\|\boldsymbol{w}_j\big\|_2-\big|\widehat{\alpha}_{1,j}\big|\big|\\ &:=\sigma_1(I_1+I_2)+I_3.\end{aligned} \tag{50}$$

From Lemma 8, we have

$$\begin{aligned}I_1&=\Big\|\overline{\boldsymbol{w}}_j^*-\widehat{\boldsymbol{V}}\widehat{\boldsymbol{V}}^T\overline{\boldsymbol{w}}_j^*\Big\|_2\le\Big\|\boldsymbol{V}\boldsymbol{V}^T-\widehat{\boldsymbol{V}}\widehat{\boldsymbol{V}}^T\Big\|_2\\ &\le\frac{\big\|\widehat{\boldsymbol{M}}_{i,2}-\boldsymbol{M}_{i,2}\big\|_2}{\sigma_K\big(\boldsymbol{M}_{i,2}\big)}\end{aligned} \tag{51}$$

where the last inequality comes from Lemma 4. Then, from (9), we know that

$$\sigma_K\big(\boldsymbol{M}_{i,2}\big)\lesssim\min_{1\le j\le K}\|\boldsymbol{w}_j\|_2\lesssim\sigma_K. \tag{52}$$

From [17, Th. 3], we have

$$\begin{aligned}I_2&=\Big\|\widehat{\boldsymbol{V}}^T\overline{\boldsymbol{w}}_j^*-s_j\widehat{\boldsymbol{u}}_j\Big\|_2\\ &\lesssim\frac{\kappa}{\sigma_K}\big\|\widehat{\boldsymbol{M}}_{i,3}\big(\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}}\big)-\boldsymbol{M}_{i,3}\big(\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}},\widehat{\boldsymbol{V}}\big)\big\|_2.\end{aligned} \tag{53}$$

To guarantee the condition (48) in Lemma 9 hold, according to Lemmas 4 and 5, we need $N\gtrsim\kappa^3M^2Kd\log d$. Then, from Lemma 9, we have

$$I_3=\big(\kappa^4K^{3/2}(I_1+I_2)+\kappa^2K^{1/2}\big\|\widehat{\boldsymbol{M}}_{i,1}-\boldsymbol{M}_{i,1}\big\|\big)\sigma_1. \tag{54}$$

Since $d\gg K$, according to Lemmas 4–6, we have

$$\big\|\boldsymbol{w}_j^*-|\widehat{\alpha}_{1,j}|\widehat{\boldsymbol{V}}\widehat{\boldsymbol{u}}_j\big\|_2\lesssim\varepsilon_0\kappa^6\sqrt{\frac{K^3d\log d}{N}}(M\sigma_1+|\xi|) \tag{55}$$

provided that $N\gtrsim d\log^4 d$.

When $N\gtrsim\varepsilon_0^{-2}\kappa^8K^4Md\log d$ for $\varepsilon_0\in(0,1)$, we have

$$\big|\widehat{\alpha}_{1,j}-\alpha_{1,j}^*\big|<\varepsilon_0\big|\alpha_{1,j}^*\big|,\ \text{and}\ \big|\widehat{\alpha}_{2,j}-\alpha_{2,j}^*\big|<\varepsilon_0\big|\alpha_{2,j}^*\big|. \tag{56}$$

Hence, $\widehat{\alpha}_{1,j}$ and $\widehat{\alpha}_{2,j}$ share the same signs of $\alpha_{1,j}^*$ and $\alpha_{2,j}^*$, and $\hat{v}_j=v_j^*$. □

C. Proof of Lemma 2

In this section, we provide the proof of Lemma 2, which shows the local convexity of f in a small neighborhood of $\boldsymbol{W}^*$. The roadmap is to first bound the smallest eigenvalue of $\nabla^2 f$ in the ground truth as shown in (59), then show that the difference of $\nabla^2 f$ between any fixed point $\boldsymbol{W}$ in this region and the ground truth $\boldsymbol{W}^*$ is bounded in terms of $\|\boldsymbol{W}-\boldsymbol{W}^*\|_2$ by Lemma 10 the proof of which is in the Supplementary Materials.

Lemma 10: Suppose $\boldsymbol{W}$ satisfies (21), with any $1\le j\le K$ and $1\le i\le M$, we have

$$\mathbb{E}_{\boldsymbol{x}}\Big|\phi'\big(\boldsymbol{w}_j^T\boldsymbol{P}_i\boldsymbol{x}\big)-\phi'\Big(\boldsymbol{w}_j^{*T}\boldsymbol{P}_i\boldsymbol{x}\Big)\Big|\le\frac{2}{\pi}\frac{\big\|\boldsymbol{w}_j^*-\boldsymbol{w}_j\big\|}{\big\|\boldsymbol{w}_j^*\big\|} \tag{57}$$

$$\big\|\nabla^2f\big(\boldsymbol{W}^*\big)-\nabla^2f(\boldsymbol{W})\big\|\le 4M^2K^2\frac{\big\|\boldsymbol{W}^*-\boldsymbol{W}\big\|_2}{\sigma_K}. \tag{58}$$

Proof of Lemma 2: By the triangle inequality, we have

$$\left|\left\|\nabla^2 f(\boldsymbol{W})\right\|_2 - \left\|\nabla^2 f\left(\boldsymbol{W}^*\right)\right\|_2\right| \leq \left\|\nabla^2 f\left(\boldsymbol{W}^*\right) - \nabla^2 f(\boldsymbol{W})\right\|_2$$

and

$$\left\|\nabla^2 f(\boldsymbol{W})\right\|_2 \leq \left\|\nabla^2 f\left(\boldsymbol{W}^*\right)\right\|_2 + \left\|\nabla^2 f\left(\boldsymbol{W}^*\right) - \nabla^2 f(\boldsymbol{W})\right\|_2$$
$$\left\|\nabla^2 f(\boldsymbol{W})\right\|_2 \geq \left\|\nabla^2 f\left(\boldsymbol{W}^*\right)\right\|_2 - \left\|\nabla^2 f\left(\boldsymbol{W}^*\right) - \nabla^2 f(\boldsymbol{W})\right\|_2.$$

The error bound of $\|\nabla^2 f(\boldsymbol{W}^*) - \nabla^2 f(\boldsymbol{W})\|_2$ can be derived from Lemma 10, and the remaining part is to bound $\nabla^2 f(\boldsymbol{W}^*)$. The second-order derivative of f at $\boldsymbol{W}$ is written as

$$\begin{aligned}&\frac{\partial^2 f(\boldsymbol{W})}{\partial \boldsymbol{w}_{j_1}\partial \boldsymbol{w}_{j_2}}\\ &= \mathbb{E}_{\boldsymbol{x}}\left[v_i^* v_j^* \left(\sum_{i=1}^{M} \phi'\left(\boldsymbol{w}_{j_1}^T \boldsymbol{P}_i \boldsymbol{x}\right)\boldsymbol{P}_i \boldsymbol{x}\right)\left(\sum_{i=1}^{M} \phi'\left(\boldsymbol{w}_{j_2}^T \boldsymbol{P}_i \boldsymbol{x}\right)\boldsymbol{P}_i \boldsymbol{x}\right)^T\right].\end{aligned}$$

Then, denote $\boldsymbol{P}_i\boldsymbol{x}$ by $\boldsymbol{x}_i$. For any vector $\boldsymbol{a} \in \mathbb{R}^{KM}$, the lower bound of $\nabla^2 f(\boldsymbol{W}^*)$ is derived from

$$\begin{aligned}&\boldsymbol{a}^T \nabla^2 f\left(\boldsymbol{W}^*\right)\boldsymbol{a}\\ &\quad= \mathbb{E}_{\boldsymbol{x}}\left(\sum_{j=1}^{K}\sum_{i=1}^{M} v_j^* \boldsymbol{a}_j^T \boldsymbol{x}_i \phi'\left(\boldsymbol{w}_j^{*T} \boldsymbol{x}_i\right)\right)^2 := \mathbb{E}_{\boldsymbol{x}}\left(\sum_{i=1}^{M} h(\boldsymbol{x}_i)\right)^2\\ &\quad= \sum_{i=1}^{M} \mathbb{E}_{\boldsymbol{x}} h^2(\boldsymbol{x}_i) + \frac{1}{K^2}\sum_{i_1\neq i_2} \mathbb{E}_{\boldsymbol{x}} h\left(\boldsymbol{x}_{i_1}\right)h\left(\boldsymbol{x}_{i_2}\right)\\ &\quad= \sum_{i=1}^{M} \mathbb{E}_{\boldsymbol{x}} h^2(\boldsymbol{x}_i) + \sum_{i_1\neq i_2} \mathbb{E}_{\boldsymbol{x}_{i_1}} h\left(\boldsymbol{x}_{i_1}\right)\mathbb{E}_{\boldsymbol{x}_{i_2}} h\left(\boldsymbol{x}_{i_2}\right)\\ &\quad\overset{(a)}{\geq} \sum_{i=1}^{M} \mathbb{E}_{\boldsymbol{x}} h^2(\boldsymbol{x}_i) \geq \frac{M}{11\kappa^2\gamma}\|\boldsymbol{a}\|_2^2,\end{aligned} \tag{59}$$

where (a) holds since $\boldsymbol{x}_{i_1}$ and $\boldsymbol{x}_{i_2}$ share the same distribution, and the last inequality comes from [46, Lemma D.6].

Next, the upper bound of $\nabla^2 f(\boldsymbol{W}^*)$ is derived from

$$\begin{aligned}&\boldsymbol{a}^T \nabla^2 f\left(\boldsymbol{W}^*\right)\boldsymbol{a}\\ &\quad= \mathbb{E}_{\boldsymbol{x}}\left(\sum_{j=1}^{K}\sum_{i=1}^{M} v_j^* \boldsymbol{a}_j^T \boldsymbol{x}_i \phi'\left(\boldsymbol{w}_j^T \boldsymbol{x}_i\right)\right)^2\\ &\quad\leq \sum_{j_1=1}^{K}\sum_{j_2=1}^{K}\sum_{i_1=1}^{M}\sum_{i_2=1}^{M}\Big(\mathbb{E}_{\boldsymbol{x}}\left|\boldsymbol{a}_{j_1}^T\boldsymbol{x}_{i_1}\right|^4 \cdot \mathbb{E}_{\boldsymbol{x}}\left|\phi'\left(\boldsymbol{w}_{j_1}^T\boldsymbol{x}_{i_1}\right)\right|^4\\ &\qquad\qquad \cdot \mathbb{E}_{\boldsymbol{x}}\left|\boldsymbol{a}_{j_2}^T\boldsymbol{x}_{i_2}\right|^4 \cdot \mathbb{E}_{\boldsymbol{x}}\left|\phi'\left(\boldsymbol{w}_{j_2}^T\boldsymbol{x}_{i_2}\right)\right|^4\Big)^{\frac{1}{4}}\\ &\quad\leq \sum_{j_1=1}^{K}\sum_{j_2=1}^{K}\sum_{i_1=1}^{M}\sum_{i_2=1}^{M}\left(\mathbb{E}_{\boldsymbol{x}}\left|\boldsymbol{a}_{j_1}^T\boldsymbol{x}_{i_1}\right|^4 \cdot \mathbb{E}_{\boldsymbol{x}}\left|\boldsymbol{a}_{j_2}^T\boldsymbol{x}_{i_2}\right|^4\right)^{\frac{1}{4}}\\ &\quad\leq 5M^2K\|\boldsymbol{a}\|^2.\end{aligned} \tag{60}$$

Since both (59) and (60) hold for any $\boldsymbol{a} \in \mathbb{R}^{Kd}$, then

$$\frac{M}{11\kappa^2\gamma}\boldsymbol{I} \leq \nabla^2 f\left(\boldsymbol{W}^*\right) \leq 5M^2K\boldsymbol{I}. \tag{61}$$

From the assumption in (21) and Lemma 10, we have

$$\left\|\nabla^2 f(\boldsymbol{W}) - \nabla^2 f\left(\boldsymbol{W}^*\right)\right\|_2 \leq \frac{\varepsilon_0 M}{11\kappa^2\gamma}. \tag{62}$$

Combining (61) and (62) completes the whole proof. □

D. Proof of Lemma 3

The main steps in this proof is to bound the three items in (67). Lemma 11 provides the bound for case that when $i_1 = i_2$, where $\widetilde{\boldsymbol{X}}_1$ (or $\widehat{\boldsymbol{X}}_1$) and $\widetilde{\boldsymbol{X}}_2$ are correlated with each other. When $i_1 \neq i_2$, $\widetilde{\boldsymbol{X}}_1$ (or $\widehat{\boldsymbol{X}}_1$) and $\widetilde{\boldsymbol{X}}_2$ are independent, and the corresponding results are summarized in Lemma 12. Both Lemmas 11 and 12 use the fact that $\widetilde{\boldsymbol{X}}_1$, $\widetilde{\boldsymbol{X}}_2$, and $\widehat{\boldsymbol{X}}_1$ are sub-Gaussian random variables, and the definition of sub-Gaussian is summarized in Definition 1. In addition, the subexponential random variable is defined in Definition 2. The multiplication of two sub-Gaussian random variables belongs to the subexponential distribution, and this property is used in the proofs of Lemmas 11 and 12. The detailed proofs of Lemmas 11 and 12 can be found in the Supplementary Materials.

Reference [44, Lemma 5.3] provides the error bound between ∇f and $\nabla \hat{f}_{\mathcal{D}}$ for the fully connected neural networks. However, there are two major differences from our proof. First, the error bound provided in [44] is much looser than ours. Second, [44] only needs to consider the case that $i_1 = i_2 = 1$ due to the fully connected neural network structures. The error bound of [44, Lemma 5.3] is $O((d\log N)/N)^{1/2}(\|\boldsymbol{W}^*\|_2 + |\xi|)$, while the error bound in Lemma 3 is $O((d\log d)/N)^{1/2}(M\|\boldsymbol{W}^* - \boldsymbol{W}\|_2 + |\xi|)$, and $M = 1$ for fully connected neural networks. Since all the analyses are based on the fact that the iterates lie in a small neighborhood of $\boldsymbol{W}^*$, that is $\|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\|_2 \ll \|\boldsymbol{W}^*\|_2$ especially for large t. Hence, it is obvious the error bound provided in Lemma 3 is tighter.

Definition 1 [38, Definition 5.7]: A random variable X is called a sub-Gaussian random variable if it satisfies

$$\left(\mathbb{E}|X|^p\right)^{1/p} \leq c_1\sqrt{p} \tag{63}$$

for all $p \geq 1$ and some constant $c_1 > 0$. In addition, we have

$$\mathbb{E}e^{s(X-\mathbb{E}X)} \leq e^{c_2\|X\|_{\psi_2}^2 s^2} \tag{64}$$

for all $s \in \mathbb{R}$ and some constant $c_2 > 0$, where $\|X\|_{\phi_2}$ is the sub-Gaussian norm of X defined as $\|X\|_{\psi_2} = \sup_{p\geq 1} p^{-1/2}(\mathbb{E}|X|^p)^{1/p}$.

Moreover, a random vector $\boldsymbol{X} \in \mathbb{R}^d$ belongs to the sub-Gaussian distribution if 1-D marginal $\boldsymbol{\alpha}^T\boldsymbol{X}$ is sub-Gaussian for any $\boldsymbol{\alpha} \in \mathbb{R}^d$, and the sub-Gaussian norm of $\boldsymbol{X}$ is defined as $\|\boldsymbol{X}\|_{\psi_2} = \sup_{\|\boldsymbol{\alpha}\|_2=1} \|\boldsymbol{\alpha}^T\boldsymbol{X}\|_{\psi_2}$.

Definition 2 [38, Definition 5.13]: A random variable X is called a subexponential random variable if it satisfies

$$\left(\mathbb{E}|X|^p\right)^{1/p} \leq c_3 p \tag{65}$$

for all $p \geq 1$ and some constant $c_3 > 0$. In addition, we have

$$\mathbb{E}e^{s(X-\mathbb{E}X)} \leq e^{c_4\|X\|_{\psi_1}^2 s^2} \tag{66}$$

for $s \leq 1/\|X\|_{\psi_1}$ and some constant $c_4 > 0$, where $\|X\|_{\psi_1}$ is the subexponential norm of X defined as $\|X\|_{\psi_1} = \sup_{p\geq 1} p^{-1}(\mathbb{E}|X|^p)^{1/p}$.

Lemma 11: Assume $\boldsymbol{X}$, $\boldsymbol{X}h_1(\boldsymbol{X})$ and $\boldsymbol{X}h_2(\boldsymbol{X})$ all are sub-Gaussian random vectors in $\mathbb{R}^d$, where h_1 and h_2 are some fixed functions from $\mathbb{R}^d$ to $\mathbb{R}$. Let $\{\boldsymbol{X}_n\}_{n=1}^N$ be N independent samples of $\boldsymbol{X}$. Then, the following holds with probability at least $1-d^{-10}$:

$$\left\|\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{X}_n\boldsymbol{X}_n^T h_1(\boldsymbol{X}_n)h_2(\boldsymbol{X}_n)-\mathbb{E}\boldsymbol{X}\boldsymbol{X}^T h_1(\boldsymbol{X})h_2(\boldsymbol{X})\right\|_2 \lesssim \sqrt{\frac{d\log d}{N}}\|\boldsymbol{X}h_1(\boldsymbol{X})\|_{\psi_2}\|\boldsymbol{X}h_2(\boldsymbol{X})\|_{\psi_2}.$$

Lemma 12: Assume $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ are two independent sub-Gaussian random vectors in $\mathbb{R}^d$. Let $\{\boldsymbol{X}_{1,n}\}_{n=1}^N$ and $\{\boldsymbol{X}_{2,n}\}_{n=1}^N$ be N independent samples of $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, respectively. Then, provided that $N\gtrsim d\log d$, the following holds with probability at least $1-d^{-10}$:

$$\left\|\frac{1}{N}\sum_{i=1}^{N}\boldsymbol{X}_{1,n}\boldsymbol{X}_{2,n}-\mathbb{E}\boldsymbol{X}_1\boldsymbol{X}_2\right\|_2 \lesssim \sqrt{\frac{d\log d}{N}}\|\boldsymbol{X}_1\|_{\psi_2}\|\boldsymbol{X}_2\|_{\psi_2}.$$

Proof of Lemma 3: We have

$$\begin{aligned}
&\left[\nabla\hat{f}_{\mathcal{D}}(\boldsymbol{W})\right]_k\\
&=\sum_{j=1}^{K}\sum_{i_1=1}^{M}\sum_{i_2=1}^{M}\frac{v_j^*v_k^*}{N}\\
&\quad\cdot\sum_{n=1}^{N}\Big(\big(\boldsymbol{P}_{i_1}\boldsymbol{x}_n\big)\big(\boldsymbol{P}_{i_2}\boldsymbol{x}_n\big)^T\phi'\big(\boldsymbol{w}_j{}^T\boldsymbol{P}_{i_1}\boldsymbol{x}_n\big)\phi'\big(\boldsymbol{w}_k^T\boldsymbol{P}_{i_2}\boldsymbol{x}_n\big)\boldsymbol{w}_j\\
&\qquad\quad-\big(\boldsymbol{P}_{i_1}\boldsymbol{x}_n\big)\big(\boldsymbol{P}_{i_2}\boldsymbol{x}_n\big)^T\phi'\Big(\boldsymbol{w}_j^{*T}\boldsymbol{P}_{i_1}\boldsymbol{x}_n\Big)\phi'\big(\boldsymbol{w}_k^T\boldsymbol{P}_{i_2}\boldsymbol{x}_n\big)\boldsymbol{w}_j^*\Big)\\
&\quad+\sum_{i=1}^{M}\frac{v_j^*v_k^*}{N}\sum_{n=1}^{N}\xi_n(\boldsymbol{P}_i\boldsymbol{x}_n)^T\phi'\big(\boldsymbol{w}_k^T\boldsymbol{P}_i\boldsymbol{x}_n\big)\\
&=\sum_{j=1}^{K}\sum_{i_1=1}^{M}\sum_{i_2=1}^{M}\frac{v_j^*v_k^*}{N}\\
&\quad\cdot\sum_{n=1}^{N}\Big[\big(\boldsymbol{P}_{i_1}\boldsymbol{x}_n\big)\big(\boldsymbol{P}_{i_2}\boldsymbol{x}_n\big)^T\\
&\qquad\quad\cdot\phi'\Big(\boldsymbol{w}_j^{*T}\boldsymbol{P}_{i_1}\boldsymbol{x}_n\Big)\phi'\big(\boldsymbol{w}_k^T\boldsymbol{P}_{i_2}\boldsymbol{x}_n\big)\big(\boldsymbol{w}_j-\boldsymbol{w}_j^*\big)\\
&\qquad\quad+\ \big(\boldsymbol{P}_{i_1}\boldsymbol{x}_n\big)\big(\boldsymbol{P}_{i_2}\boldsymbol{x}_n\big)^T\\
&\qquad\quad\cdot\Big(\phi'\big(\boldsymbol{w}_j^T\boldsymbol{P}_{i_1}\boldsymbol{x}_n\big)-\phi'\Big(\boldsymbol{w}_j^{*T}\boldsymbol{P}_{i_1}\boldsymbol{x}_n\Big)\Big)\phi'\big(\boldsymbol{w}_k^T\boldsymbol{P}_{i_2}\boldsymbol{x}_n\big)\boldsymbol{w}_j^*\Big]\\
&\quad+\sum_{i=1}^{M}\frac{v_j^*v_k^*}{N}\sum_{n=1}^{N}\xi_n(\boldsymbol{P}_i\boldsymbol{x}_n)^T\phi'\big(\boldsymbol{w}_k^T\boldsymbol{P}_i\boldsymbol{x}_n\big).
\end{aligned}$$

For simplification, let $\widetilde{\boldsymbol{X}}_{1,n}=v_j^*v_k^*(\boldsymbol{P}_{i_1}\boldsymbol{x}_n)\phi'(\boldsymbol{w}_j^{*T}\boldsymbol{P}_{i_1}\boldsymbol{x}_n)$ and $\widetilde{\boldsymbol{X}}_{2,n}=v_j^*v_k^*(\boldsymbol{P}_{i_2}\boldsymbol{x}_n)\phi'(\boldsymbol{w}_j^{*T}\boldsymbol{P}_{i_2}\boldsymbol{x}_n)$. In addition, let $\widehat{\boldsymbol{X}}_{1,n}=v_j^*v_k^*(\boldsymbol{P}_{i_1}\boldsymbol{x}_n)(\phi'(\boldsymbol{w}_j{}^T\boldsymbol{P}_{i_1}\boldsymbol{x}_n)-\phi'(\boldsymbol{w}_j^{*T}\boldsymbol{P}_{i_1}\boldsymbol{x}_n))$. Then, we have

$$\begin{aligned}
&\left[\nabla\hat{f}_{\mathcal{D}}(\boldsymbol{W})\right]_k\\
&\quad=\frac{1}{N}\sum_{j,i_1,i_2,n}\left[\widetilde{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T\big(\boldsymbol{w}_j-\boldsymbol{w}_j^*\big)-\widehat{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T\boldsymbol{w}_j^*\right]\\
&\qquad\quad+\sum_{i_1=1}^{M}\frac{1}{N}\sum_{n=1}^{N}\xi_n\widetilde{\boldsymbol{X}}_{1,n}^T.
\end{aligned}$$

Hence, we have

$$\begin{aligned}
&[\nabla f(\boldsymbol{W})]_k-\left[\nabla\hat{f}_{\mathcal{D}}(\boldsymbol{W})\right]_k\\
&\quad=\frac{1}{N}\sum_{j,i_1,i_2,n}\Big[\Big(\widetilde{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T-\mathbb{E}\widetilde{\boldsymbol{X}}_1\widetilde{\boldsymbol{X}}_2^T\Big)\big(\boldsymbol{w}_j-\boldsymbol{w}_j^*\big)\\
&\qquad\qquad-\Big(\widehat{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T-\mathbb{E}\widehat{\boldsymbol{X}}_1\widetilde{\boldsymbol{X}}_2^T\Big)\boldsymbol{w}_j^*\Big]\\
&\qquad+\sum_{i_1=1}^{M}\frac{1}{N}\sum_{n=1}^{N}\xi_n\widetilde{\boldsymbol{X}}_{1,n}^T.
\end{aligned}\tag{67}$$

We claim that $\widetilde{\boldsymbol{X}}_1$ and $\widehat{\boldsymbol{X}}_1$ belong to the sub-Gaussian distribution. According to Definition 1, for any $\boldsymbol{\alpha}\in\mathbb{R}^d$, we have

$$\begin{aligned}
&\big(\mathbb{E}_{\boldsymbol{x}}|\boldsymbol{\alpha}^T\widetilde{\boldsymbol{X}}_1|^p\big)^{1/p}\\
&\quad\le\big(\mathbb{E}_{\boldsymbol{x}}|\boldsymbol{\alpha}^T\boldsymbol{P}_i\boldsymbol{x}|^p\cdot\mathbb{E}_{\boldsymbol{x}}|\phi'\big(\boldsymbol{w}_j^T\boldsymbol{P}_i\boldsymbol{x}\big)|^p\big)^{1/p}\\
&\quad\le\big(\mathbb{E}_{\boldsymbol{x}}|\boldsymbol{\alpha}^T\boldsymbol{P}_i\boldsymbol{x}|^p\big)^{1/p}\le\sqrt{p}
\end{aligned}\tag{68}$$

where the last inequality holds since $\boldsymbol{P}_i\boldsymbol{x}$ is a Gaussian random vector with covariance matrix $\boldsymbol{I}_d$.

For $\widehat{\boldsymbol{X}}_1$, we have

$$\begin{aligned}
&\Big(\mathbb{E}_{\boldsymbol{x}}\big|\boldsymbol{\alpha}^T\widehat{\boldsymbol{X}}_1\big|^p\Big)^{1/p}\\
&\le\Big(\mathbb{E}_{\boldsymbol{x}}\big|\boldsymbol{\alpha}^T\boldsymbol{P}_i\boldsymbol{x}\big|^p\cdot\mathbb{E}_{\boldsymbol{x}}\Big|\phi'\big(\boldsymbol{w}_j{}^T\boldsymbol{P}_{i_1}\boldsymbol{x}_n\big)-\phi'\Big(\boldsymbol{w}_j^{*T}\boldsymbol{P}_{i_1}\boldsymbol{x}_n\Big)\Big|^p\Big)^{1/p}\\
&\le\frac{2}{\pi}\frac{\|\boldsymbol{w}_{j_2}^*-\boldsymbol{w}_{j_2}\|}{\|\boldsymbol{w}_{j_2}^*\|}\sqrt{p}
\end{aligned}$$

where the last inequality comes from Lemma 10.

When $i_1=i_2$, by Lemma 11, we have

$$\begin{aligned}
&\left\|\widetilde{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T-\mathbb{E}\widetilde{\boldsymbol{X}}_1\widetilde{\boldsymbol{X}}_2^T\right\|_2\lesssim\sqrt{\frac{d\log d}{N}}\\
&\left\|\widehat{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T-\mathbb{E}\widehat{\boldsymbol{X}}_1\widetilde{\boldsymbol{X}}_2^T\right\|_2\lesssim\sqrt{\frac{d\log d}{N}}\cdot\frac{\|\boldsymbol{w}_j^*-\boldsymbol{w}_j\|}{\|\boldsymbol{w}_j^*\|}
\end{aligned}\tag{69}$$

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

When $i_1\neq i_2$, from Lemma 12, we also have

$$\begin{aligned}
&\left\|\widetilde{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T-\mathbb{E}\widetilde{\boldsymbol{X}}_1\widetilde{\boldsymbol{X}}_2^T\right\|_2\lesssim\sqrt{\frac{d\log d}{N}}\\
&\left\|\widehat{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T-\mathbb{E}\widehat{\boldsymbol{X}}_1\widetilde{\boldsymbol{X}}_2^T\right\|_2\lesssim\sqrt{\frac{d\log d}{N}}\cdot\frac{\|\boldsymbol{w}_j^*-\boldsymbol{w}_j\|}{\|\boldsymbol{w}_j^*\|}.
\end{aligned}\tag{70}$$

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

For $\sum_{n=1}^{N}\xi_n\widetilde{\boldsymbol{X}}_{1,n}^T$, we have

$$\begin{aligned}
\left\|\frac{1}{N}\sum_{n=1}^{N}\xi_n\widetilde{\boldsymbol{X}}_{1,n}\right\|_2&\le|\xi|\cdot\left\|\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{P}_{i_1}\boldsymbol{x}_n\phi'\big(\boldsymbol{w}_j^T\boldsymbol{P}_{i_1}\boldsymbol{x}_n\big)\right\|_2\\
&\le|\xi|\cdot\left\|\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{P}_{i_1}\boldsymbol{x}_n\right\|_2\\
&\lesssim\sqrt{\frac{d\log d}{N}}|\xi|
\end{aligned}$$

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

In conclusion, with probability at least $1 - K^2M^2d^{-10}$,

$$\begin{aligned}
&\big\|\nabla f(\boldsymbol{W}) - \nabla \hat{f}_{\mathcal{D}}(\boldsymbol{W})\big\|_F \\
&\le \sum_{\substack{k=1,j=1\\ i_1=1,i_2=1}}^{K,K,M,M} \bigg\| \frac{1}{N}\sum_{n=1}^{N} \widetilde{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T - \mathbb{E}\widetilde{\boldsymbol{X}}_1\widetilde{\boldsymbol{X}}_2^T \bigg\|_2 \big\|\boldsymbol{w}_j - \boldsymbol{w}_j^*\big\|_2 \\
&\quad + \sum_{\substack{k=1,j=1\\ i_1=1,i_2=1}}^{K,K,M,M} \bigg\| \frac{1}{N}\sum_{n=1}^{N} \widehat{\boldsymbol{X}}_{1,n}\widetilde{\boldsymbol{X}}_{2,n}^T - \mathbb{E}\widehat{\boldsymbol{X}}_1\widetilde{\boldsymbol{X}}_2^T \bigg\|_2 \big\|\boldsymbol{w}_j^*\big\|_2 \\
&\quad + \sum_{k=1,i_1=1}^{K,M} |\xi| \cdot \bigg\| \frac{1}{N}\sum_{n=1}^{N} \widetilde{X}_{1,n} \bigg\|_2 \\
&\lesssim M^2K^2\sqrt{\frac{d\log d}{N}}\big\|\boldsymbol{W} - \boldsymbol{W}^*\big\|_2 + MK\sqrt{\frac{d\log d}{N}}|\xi|.
\end{aligned}$$

□

REFERENCES

[1] Z. Allen-Zhu, Y. Li, and Y. Liang, "Learning and generalization in overparameterized neural networks, going beyond two layers," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 6155–6166.

[2] O. Bousquet and A. Elisseeff, "Stability and generalization," *J. Mach. Learn. Res.*, vol. 2, pp. 499–526, Mar. 2002.

[3] A. Brutzkus and A. Globerson, "Globally optimal gradient descent for a ConvNet with Gaussian inputs," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 605–614.

[4] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 10, pp. 6232–6251, Oct. 2016.

[5] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, 2008, pp. 160–167.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[7] S. S. Du, J. D. Lee, and Y. Tian, "When is a convolutional filter easy to learn?" 2017, *arXiv:1709.06129*. [Online]. Available: http://arxiv.org/abs/1709.06129

[8] S. S. Du, J. D. Lee, Y. Tian, A. Singh, and B. Poczos, "Gradient descent learns one-hidden-layer CNN: Don't be afraid of spurious local minima," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1338–1347.

[9] H. Fu, Y. Chi, and Y. Liang, "Guaranteed recovery of one-hidden-layer neural networks via cross entropy," 2018, *arXiv:1802.06463*. [Online]. Available: http://arxiv.org/abs/1802.06463

[10] R. Ge, J. D. Lee, and T. Ma, "Learning one-hidden-layer neural networks with landscape design," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–37. [Online]. Available: https://openreview.net/forum?id=BkwHObbRZ

[11] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Statist.*, 2011, pp. 315–323.

[12] S. Goel, A. Klivans, and R. Meka, "Learning one convolutional layer with overlapping patches," in *Proc. ICML*, 2018, pp. 1783–1791.

[13] R. H. Hahnloser and H. S. Seung, "Permitted and forbidden sets in symmetric threshold-linear networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 217–223.

[14] M. Hardt, B. Recht, and Y. Singer, "Train faster, generalize better: Stability of stochastic gradient descent," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1225–1234.

[15] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," 2016, *arXiv:1609.04836*. [Online]. Available: http://arxiv.org/abs/1609.04836

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[17] V. Kuleshov, A. Chaganty, and P. Liang, "Tensor factorization via matrix factorization," in *Proc. Artif. Intell. Statist.*, 2015, pp. 507–516.

[18] T. Laurent and J. Brecht, "The multilinear structure of ReLU networks," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2908–2916.

[19] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, "Face recognition: A convolutional neural-network approach," *IEEE Trans. Neural Netw.*, vol. 8, no. 1, pp. 98–113, Jan. 1997.

[20] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[21] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[22] Y. Li and Y. Yuan, "Convergence analysis of two-layer neural networks with ReLU activation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 597–607.

[23] S. Liang, R. Sun, J. D. Lee, and R. Srikant, "Adding one neuron can eliminate all bad local minima," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 4355–4365.

[24] R. Livni, S. Shalev-Shwartz, and O. Shamir, "On the computational efficiency of training neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 855–863.

[25] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. 30th Int. Conf. Mach. Learn.*, 2013, vol. 30, no. 1, pp. 1–6.

[26] Y. Nesterov, *Introductory Lectures Convex Optimization: A Basic Course*, vol. 87. Boston, MA, USA: Springer, 2013.

[27] B. Neyshabur, S. Bhojanapalli, D. McAllester, and N. Srebro, "Exploring generalization in deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5947–5956.

[28] B. T. Polyak, *Introduction to optimization*. New York, NY, USA: Optimization Software, 1987.

[29] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychol. Rev.*, vol. 65, no. 6, p. 386, 1958.

[30] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognit. Model.*, vol. 5, no. 3, pp. 533–536, 1988.

[31] I. Safran and O. Shamir, "Spurious local minima are common in two-layer ReLU neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4430–4438.

[32] O. Shamir, "Distribution-specific hardness of learning neural networks," *J. Mach. Learn. Res.*, vol. 19, no. 1, pp. 1135–1163, 2018.

[33] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[34] M. Soltanolkotabi, A. Javanmard, and J. D. Lee, "Theoretical insights into the optimization landscape of over-parameterized shallow neural networks," *IEEE Trans. Inf. Theory*, vol. 65, no. 2, pp. 742–769, Feb. 2019.

[35] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1139–1147.

[36] Y. Tian, "An analytical formula of population gradient for two-layered ReLU network and its applications in convergence and critical point analysis," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 3404–3413.

[37] J. A. Tropp, "User-friendly tail bounds for sums of random matrices," *Found. Comput. Math.*, vol. 12, no. 4, pp. 389–434, Aug. 2012.

[38] R. Vershynin, "Introduction to the non-asymptotic analysis of random matrices," 2010, *arXiv:1011.3027*. [Online]. Available: http://arxiv.org/abs/1011.3027

[39] G. Wang, G. B. Giannakis, and J. Chen, "Learning ReLU networks on linearly separable data: Algorithm, optimality, and generalization," *IEEE Trans. Signal Process.*, vol. 67, no. 9, pp. 2357–2370, May 2019.

[40] S. Wu, A. G. Dimakis, and S. Sanghavi, "Learning distributions generated by one-layer ReLU networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 8105–8115.

[41] T. Yang, Q. Lin, and Z. Li, "Unified convergence analysis of stochastic momentum methods for convex and non-convex optimization," 2016, *arXiv:1604.03257*. [Online]. Available: http://arxiv.org/abs/1604.03257

[42] G. Yehudai and O. Shamir, "On the power and limitations of random features for understanding neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 6594–6604.

[43] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," 2016, *arXiv:1611.03530*. [Online]. Available: http://arxiv.org/abs/1611.03530

[44] X. Zhang, Y. Yu, L. Wang, and Q. Gu, "Learning one-hidden-layer ReLU networks via gradient descent," in *Proc. 22nd Int. Conf. Artif. Intell. Statist.*, 2019, pp. 1524–1534.

[45] K. Zhong, Z. Song, and I. S. Dhillon, "Learning non-overlapping convolutional neural networks with multiple kernels," 2017, *arXiv:1711.03440*. [Online]. Available: http://arxiv.org/abs/1711.03440

[46] K. Zhong, Z. Song, P. Jain, P. L. Bartlett, and I. S. Dhillon, "Recovery guarantees for one-hidden-layer neural networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp, 4140–4149. [Online]. Available: https://arxiv.org/abs/1706.03175

Shuai Zhang (Graduate Student Member, IEEE) received the B.E. degree from the University of Science and Technology of China, Hefei, China, in 2016. He is currently pursuing the Ph.D. degree in electrical engineering with the Rensselaer Polytechnic Institute, Troy, NY, USA.

His research interests include signal processing and high dimensional data analysis.

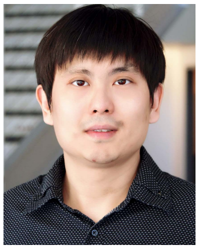
Meng Wang (Member, IEEE) received the Ph.D. degree from Cornell University, Ithaca, NY, USA, in 2012.

She is currently an Associate Professor with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY, USA. Her research interests include high-dimensional data analysis and their applications in power systems monitoring and network inference.

Jinjun Xiong (Senior Member, IEEE) received the Ph.D. degree from the University of California, Los Angeles, CA, USA, in 2006.

He is currently a Research Staff Member and the Program Director for cognitive computing systems research with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA, where he also co-directs the IBM-Illinois Center for Cognitive Computing Systems Research. His research interests include artificial intelligence, machine learning, and systems.

Dr. Xiong received six best paper awards and eight nominations for best paper awards at various international conferences.



Sijia Liu (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Syracuse University, Syracuse, NY, USA, in 2016.

He was a Post-Doctoral Research Fellow with the University of Michigan at Ann Arbor, Ann Arbor, MI, USA. He joined the IBM Research, San Jose, CA, USA, where he is currently a Research Staff Member with the MIT–IBM Watson AI Lab, Cambridge, MA, USA. His recent research interests include optimization for deep learning and adversarial machine learning.

Dr. Liu received the All-University Doctoral Prize for his Ph.D. degree. He received the Best Student Paper Award (Third Place) at ICASSP'17. He was among the seven finalists of the Best Student Paper Award at Asilomar'13.

Pin-Yu Chen (Member, IEEE) received the Ph.D. degree in electrical engineering and computer science from the University of Michigan at Ann Arbor, Ann Arbor, MI, USA, in 2016.

He is currently a Research Staff Member with the IBM Thomas J. Watson Research Center, Yorktown Heights, NY, USA. He is also the Chief Scientist of RPI-IBM AI Research Collaboration and a PI of the ongoing MIT–IBM Watson AI Lab projects. His recent research is on adversarial machine learning and robustness of neural networks. His long-term research vision is building a trustworthy machine learning system.

Dr. Chen received the NeurIPS 2017 Best Reviewer Award and the IEEE GLOBECOM 2010 GOLD Best Paper Award.