A Recipe for Global Convergence Guarantee in Deep Neural Networks

Kenji Kawaguchi,[1*] **Qingyun Sun,**[2*]
[1] Harvard University
[2] Stanford University
kkawaguchi@fas.harvard.edu, qysun@stanford.edu

Abstract

Existing global convergence guarantees of (stochastic) gradient descent do not apply to practical deep networks in the practical regime of deep learning beyond the neural tangent kernel (NTK) regime. This paper proposes an algorithm, which is ensured to have global convergence guarantees in the practical regime beyond the NTK regime, under a verifiable condition called the expressivity condition. The expressivity condition is defined to be both data-dependent and architecture-dependent, which is the key property that makes our results applicable for practical settings beyond the NTK regime. On the one hand, the expressivity condition is theoretically proven to hold data-independently for fully-connected deep neural networks with narrow hidden layers and a single wide layer. On the other hand, the expressivity condition is numerically shown to hold data-dependently for deep (convolutional) ResNet with batch normalization with various standard image datasets. We also show that the the proposed algorithm has generalization performances comparable with those of the heuristic algorithm, with the same hyper-parameters and total number of iterations. Therefore, the proposed algorithm can be viewed as a step towards providing theoretical guarantees for deep learning in the practical regime.

1 Introduction

The pursuit of global convergence guarantee has been one of the important aspects of optimization theory. However, ensuring global convergence is notoriously hard for first-order optimization algorithms used to train deep neural networks (Goodfellow, Bengio, and Courville 2016). Recently, some progress has been made on understanding the optimization aspect of overparametrized neural networks. Overparametrized neural networks can be trained to have zero training errors, interpolating all the training data points, and are recently shown to have global convergence guarantees in theoretical regimes (Li and Liang 2018; Soltanolkotabi, Javanmard, and Lee 2018; Kawaguchi and Huang 2019; Daniely 2019; Bresler and Nagaraj 2020; Montanari and Zhong 2020; Bubeck et al. 2020). These studies open up an insightful direction leading to the understanding of the optimization aspect of deep learning.

However, there is still a significant gap between theory and practice. In applications such as computer vision, speech and natural language, a major reason for the success of deep learning in practice is its ability to learn representations with multiple levels of abstraction during training, as explained by LeCun, Bengio, and Hinton (2015). In contrast, special types of neural networks studied in previous theories with global convergence guarantees are not allowed to learn representation during training, as the neural tangent kernels are approximately unchanged during training. Indeed, such special neural networks without the capability to learn representation are considered to have limitations compared to those with the capability (Wei et al. 2019; Chizat, Oyallon, and Bach 2019; Yehudai and Shamir 2019). Furthermore, the set of neural networks studied by previous theories have not yet practical deep neural networks used in practice with good generalization performances (Kawaguchi, Kaelbling, and Bengio 2017; Poggio et al. 2017).

In this work, we propose a two-phase method to modify a base algorithm such that the modified algorithm enables practical deep neural networks to learn representation while having global convergence guarantees *of all layers* under verifiable conditions. Our global convergence guarantees are applicable to a wide range of practical deep neural networks, including deep convolutional networks with skip connection and batch normalization. For example, the verifiable conditions for global convergence guarantees are shown to be satisfied by both fully connected deep neural networks and deep residual neural networks (ResNets) with convolutional layers. Our main contributions can be summarized as:

- We propose a novel algorithm that turns any given first-order training algorithm into a two-phase training algorithm.
- We prove that the resulting two-phase training algorithms find global minima for all layers of deep neural networks, under the *expressivity condition*.
- The condition for global convergence is verified theoretically for fully connected networks with last hidden layer being wide (as the number of training data points) and all other hidden layers being narrow (as the input dimension).
- The condition for global convergence is verified numerically for the deep (convolutional) ResNet with bath-normalization on various standard datasets.

*Equal contribution

- We compare the standard training algorithm (SGD with momentum) and the two-phase version of it with the same hyperparameters and total iterations. The two-phase version is shown to preserve the practical generalization performances of the standard training while providing global convergence guarantees.

2 Related Work

In this section, we discuss related studies and their relationships with the contributions of this paper.

Over-parameterization Over-parameterization has been shown to help optimization of neural networks. More concretely, over-parameterization can remove suboptimal local minima (Soudry and Carmon 2016) and improve the quality of random initialization (Safran and Shamir 2016). Furthermore, gradual over-parameterization (i.e., gradually increasing the number of parameters) is recently shown to improve steadily the quality of local minima (Kawaguchi, Huang, and Kaelbling 2019). The extreme over-parameterization that requires the number of neurons to approach infinity is used to prove global convergence (Mei, Montanari, and Nguyen 2018; Mei, Misiakiewicz, and Montanari 2019; Chizat and Bach 2018; Dou and Liang 2020; Wei et al. 2019; Fang et al. 2020). Polynomial degrees of over-parameterization are also utilized for global convergence in the lazy training regime.

Neural tangent kernel and lazy training It was shown that neural networks under lazy training regime (with a specific scaling and initialization) is nearly a linear model fitted with random features induced by the neural tangent kernel (NTK) at random initialization. Accordingly, in the lazy training regime, which is also called the NTK regime, neural networks provably achieve globally minimum training errors. The lazy training regime is studied for both shallow (with one hidden layer) and deep neural networks and convolutional networks in previous studies (Zou et al. 2020; Li and Liang 2018; Jacot, Gabriel, and Hongler 2018; Du et al. 2019, 2018; Chizat, Oyallon, and Bach 2019; Arora et al. 2019b; Allen-Zhu, Li, and Liang 2019; Fang et al. 2020; Montanari and Zhong 2020).

Lazy training and degree of overparametrization The global convergence guarantee in the lazy training regime was first proven by using the significant overparametrization that requires the number of neurons per layer to be large polynomials in the number of data points (Li and Liang 2018; Soltanolkotabi, Javanmard, and Lee 2018). Later, the requirement on the degree of over-parametrization has been improved to a small polynomial dependency (Kawaguchi and Huang 2019; Bresler and Nagaraj 2020). Furthermore, for two-layer networks with random i.i.d. weights and i.i.d. input data, the requirement was reduced to the number of training data points divided by the input dimension up to log factors, which is the optimal order in theory (Daniely 2019; Montanari and Zhong 2020; Bubeck et al. 2020).

Beyond lazy training regime However, it has been noted that neural networks in many real-world applications has weight parameters trained beyond the lazy training regime, so that the learned features have better expressive power than random features (Yehudai and Shamir 2019; Ghorbani et al. 2019; Arora et al. 2019b,a). Accordingly, a series of studies have demonstrated that the lazy training perspective of neural networks is not enough for understanding the success of deep learning (Wei et al. 2019; Chizat, Oyallon, and Bach 2019; Yehudai and Shamir 2019). Indeed, there are also previous works for the regime beyond the lazy training (Kawaguchi 2016; Kawaguchi and Bengio 2019; Jagtap, Kawaguchi, and Karniadakis 2020; Jagtap, Kawaguchi, and Em Karniadakis 2020). To overcome the weakness of lazy training, in this work, we present a novel method to use learned representation with a learned neural tangent kernel, instead of standard lazy training that use almost the random initialized neural tangent kernel. Our experiments on multiple ML benchmark datasets show empirically that our two-phase training method achieves comparable generalization performances with standard SGD training.

Relation to this paper Unlike previous work on the lazy training regime that use the NTK at random initialization, we allow the NTK to change significantly during training, to learn features and representation. In terms of the degree of overparametrization, the results in this paper achieve the linear order (in the number of training data points) without the assumptions of the i.i.d. weights and i.i.d random input. Our results are also applicable for deep neural networks in practical settings without degrading the generalization performances. On the other hand, this paper further shows that the study of lazy training regime is also useful to understand the new two-phase training algorithm. Thus, we hope that the proposed two-phase training algorithm becomes a bridge between practice and theory of neural tangent kernel.

3 Model

In this paper, we consider the empirical risk minimization problem. Let $((x_i, y_i))_{i=1}^n$ be a training dataset of n samples where $S_x = \{x_i\}_{i=1}^n$ and $S_y = \{y_i\}_{i=1}^n$ are the set of training inputs and target outputs, with $x_i \in \mathcal{X} \subseteq \mathbb{R}^{m_x}$ and $y_i \in \mathcal{Y} \subseteq \mathbb{R}^{m_y}$. Let $\ell : \mathbb{R}^{m_y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ be the loss of each sample that measures the difference between the prediction $f(x_i, w)$ and the target y_i. The goal of empirical risk minimization is to find a prediction function $f(\cdot\,; w) : \mathbb{R}^{m_x} \to \mathbb{R}^{1 \times m_y}$, by minimizing

$$\mathcal{L}(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i, w)^\top, y_i)$$

where $w \in \mathbb{R}^d$ is the parameter vector that contains all the trainable parameters, including the weights and bias terms of all layers of deep neural networks. We define $w_{(l)} \in \mathbb{R}^{d_l}$ to be the vector of all the trainable parameters at l-th layer. For any pair (r, q) such that $1 \leq r \leq q \leq H+1$, let $w_{(r:q)} = [w_{(r)}^\top, \ldots, w_{(q)}^\top]^\top \in \mathbb{R}^{d_{r:q}}$ where $w_{(r:q)} = w_{(r)}$ when $r = q$. With this notation, we can write $w = w_{(1:H+1)}$.

Here, $f(x, w)$ represents the *pre-activation* output of the last layer of a neural network for a given (x, w). Then the output over all data points is $f_X(w) =$

$[f(x_1, w)^\top, \ldots, f(x_n, w)^\top]^\top \in \mathbb{R}^{n \times m_y}$. The pre-activation of the last layer is an affine map given by

$$f_X(w) = \mathbf{h}_{\mathbf{X}}^{(\mathbf{H})} W^{(H+1)} + b^{(H+1)},$$

where $W^{(H+1)} \in \mathbb{R}^{m_H \times m_y}$ and $b^{(H+1)} \in \mathbb{R}^{1 \times m_y}$ are the weight matrix and the bias term at the last layer. Here,

$$\mathbf{h}_{\mathbf{X}}^{(\mathbf{H})} = h_X^{(H)}(w_{(1:H)}) \in \mathbb{R}^{n \times m_H}$$

is the matrix that contains the outputs at the last hidden layer. In order to consider layers with batch normalization, we allow $h_X^{(H)}(w_{(1:H)})$ and $f(x_i, w)$ for each $i \in \{1, \ldots, n\}$ to depend on all data points $x_1, \ldots, x_n$. Here, $w_{(1:H)}$ represents the vector of the trainable parameters of all hidden layers, including the parameters of batch normalization.

4 Algorithm

We now describe the two-phase method as a modification of any given first-order base algorithm. The modified algorithm is proven to have global convergence guarantee under verifiable conditions as shown in the next two sections. The base algorithm can be any given first-order algorithm, including both batch and stochastic algorithms, such as gradient descent and stochastic gradient descent with momentum and adaptive step-size.

The description of the algorithm is presented in Algorithm 1, where $\eta_t \odot g^t$ represent the Hadamard product of η_t and g^t. Here, g^t represents the rules of the parameter update that correspond to different base algorithms. For example, if we use the (mini-batch) stochastic gradient descent as the base algorithm, g^t represents the (mini-batch) stochastic gradient of the loss function with respect to w at the time t.

The first phase of the training algorithm is the same as the base algorithm. Then a random Gaussian perturbation is added on all but last layer weights. After the random perturbation, the second training phase starts. In the second training phase, the base algorithm is modified to preserve the rank of the NTK at time τ after random perturbation, as:

$$\operatorname{rank}\left(\mathbf{K}(w^k)\right) \geq \operatorname{rank}\left(\mathbf{K}(w^\tau)\right), \quad \forall k = \tau, \tau+1, \ldots, t$$

where the NTK matrix, $\mathbf{K}(w) \in \mathbb{R}^{nm_y \times nm_y}$, is defined by

$$\mathbf{K}(w) = \frac{\partial \operatorname{vec}(f_X(w)^\top)}{\partial w} \left(\frac{\partial \operatorname{vec}(f_X(w)^\top)}{\partial w}\right)^\top.$$

As two examples, the rank can be preserved by lazy training on all layer weights or by only training the parameters in the last layer in the second phase. In the next section, we will develop the global convergence theory for Algorithm 1.

5 Theoretical Analysis

In this section, we prove that the parameter w^t in Algorithm 1 converges to a global minimum w^* *of all layers* under the expressivity condition. As a concrete example, we prove that fully-connected neural networks with softplus nonlinear activations and moderately wide last hidden layer satisfy the expressivity condition for all distinguishable training datasets. All proofs in this paper are deferred to Appendix.

Algorithm 1 Two-phase modification $\mathcal{A}$ of a base algorithm with global convergence guarantees

Inputs: an initial parameter vector w^0, a time τ and a base algorithm with updates sequence $(g_t)_t$ and learning rate sequence $(\eta_t)_t$.
▷ **First training phase**
for $t = 0, 1, \ldots, \tau - 1$ **do**
  Update parameters: $w^{t+1} = w^t - \eta_t \odot g^t$
▷ **Random perturbation**
Add noise at time τ,

$$w^\tau_{(1:H)} \leftarrow w^\tau_{(1:H)} + \delta,$$

where the noise vector $\delta = (\delta_1 \ldots, \delta_H) \in \mathbb{R}^{d_{1:H}}$ is sampled from a non-degenerate Gaussian distribution: $\delta_h \sim N(0, \sigma_h^2 I_{d_h})$ for $h = 1, \ldots, H$.
▷ **Second training phase**
for $t = \tau, \tau+1, \ldots$ **do**
  Update parameters: $w^{t+1} = w^t - \eta_t \odot g^t$, where the learning rate $(\eta_t)_{t>\tau}$ is modified to satisfy the *rank preserving* condition: for $k = \tau, \tau+1, \ldots, t$,

$$\operatorname{rank}\left(\mathbf{K}(w^k)\right) \geq \operatorname{rank}\left(\mathbf{K}(w^\tau)\right).$$

5.1 Expressivity Condition

Making the right assumption is often the most critical step in theoretical analysis of deep learning. The assumption needs to be both weak enough to be useful in practice and strong enough for proving desired conclusions. It is often challenging to find the assumption with the right theory-practice trade-off, as typical assumptions that lead to desired conclusions are not weak enough to hold in practice, which contributes to the gap between theory and practice. We aim to find the right trade-off by proposing a data-architecture-dependent, time-independent, and verifiable condition called the *expressivity condition* as a cornerstone for global convergence results. The expressivity condition guarantees the existence of parameters that can interpolate all the training data.

Assumption 1. (Expressivity condition) *There exists* $w_{(1:H)}$ *such that* $\varphi(w_{(1:H)}) \neq 0$, *where* $\varphi(w_{(1:H)}) := \det([h_X^{(H)}(w_{(1:H)}), \mathbf{1}_n][h_X^{(H)}(w_{(1:H)}), \mathbf{1}_n]^\top)$.

In the expressivity condition, the map $h_X^{(H)}$ depends on both architecture and dataset. Such dependency is essential for the theory-practice trade-off; i.e., we obtain a desired conclusion yet only for a certain class of paris of dataset and architecture. We verify the expressivity condition in our experiments. The expressivity condition is also verifiable as demonstrated below.

5.2 Real Analyticity

To prove the global convergence, we also require the function $h_X^{(H)}$ to be real analytic. Since a composition of real analytic functions is real analytic, we only need to check whether each operation satisfies the real analyticity. The

convolution, affine map, average pooling, shortcut skip connection are all real analytic functions. Therefore, the composition of these layers preserve real analyticity.

We now prove that the batch normalization function is also real analytic. The batch normalization that is applied to an output z of an arbitrary coordinate can be written by

$$\mathrm{BN}_{\gamma,\beta}(z) = \gamma \frac{z-\mu}{\sqrt{\sigma^2+\epsilon}} + \beta.$$

Here, μ and σ^2 depend also on other samples as $\mu = \frac{1}{|S|}\sum_{i\in S} z_i$ and $\sigma^2 = \frac{1}{|S|}\sum_{i\in S}(z_i-\mu)^2$, where S is an arbitrary subset of $\{1,2,\dots,n\}$ such that $z \in \{z_i : i \in S\}$. Then, the following statement holds:

Proposition 1. *Batch normalization function $(z,\beta,\gamma) \mapsto \mathrm{BN}_{\gamma,\beta}(z)$ is real analytic.*

We also require the activation function to be analytic. For example, sigmoid, hyperbolic tangents and softplus activations $\sigma(z) = \ln(1+\exp(\varsigma z))/\varsigma$ are all real analytic functions, with any hyperparameter $\varsigma > 0$. The softplus activation can approximate the ReLU activation for any desired accuracy as

$$\sigma(x) \to \mathrm{relu}(x) \text{ as } \varsigma \to \infty.$$

Therefore, the function $h_X^{(H)}$ is real analytic for a large class of neural networks (with batch normalization) such as the standard deep residual networks (He et al. 2016) with real analytic approximation of ReLU activation via softplus.

5.3 Global Convergence

In the following, we assume that the loss function satisfies assumption 2.

Assumption 2. (Use of common loss criteria) *For any $i \in \{1,\dots,n\}$, the function $\ell_i : q \mapsto \ell(q,y_i) \in \mathbb{R}_{\geq 0}$ is differentiable and convex, and $\nabla\ell_i$ is L_ℓ-Lipschitz: i.e., $\|\nabla\ell_i(q) - \nabla\ell_i(q')\| \leq L_\ell\|q-q'\|$ for all $q,q' \in \mathbb{R}$.*

Assumption 2 is satisfied by standard loss functions such as the squared loss $\ell(q,y) = \|q-y\|_2^2$ and cross entropy loss $\ell(q,y) = -\sum_{k=1}^{d_y} y_k \log \frac{\exp(q_k)}{\sum_{k'}\exp(q_{k'})}$. Although the objective function $\mathcal{L} : w \mapsto \mathcal{L}(w)$ used to train a neural network is non-convex in w, the loss criterion $\ell_i : q \mapsto \ell(q,y_i)$ is often convex in q.

Before we state the main theorem, we define the following notation. Let $w^* \in \mathbb{R}^d$ be a global minimum of all layers; i.e., w^* is a global minimum of $\mathcal{L}$. Define $\nu = [\mathbf{0}_{d_{1:H}}^\top, \mathbf{1}_{d_{H+1}}^\top]^\top$ where $\mathbf{0}_{d_{1:H}} \in \mathbb{R}^{d_{1:H}}$ is the column vector of all entries being zeros and $\mathbf{1}_{d_{H+1}} \in \mathbb{R}^{d_{H+1}}$ is the column vector of all entries being ones. Let $R^2 = \min_{\bar{w}^*_{(H+1)}\in Q} \mathbb{E}[\|\bar{w}^*_{(H+1)} - w^\tau_{(H+1)}\|^2]$ where $Q = \mathrm{argmin}_{w_{(H+1)}} \mathcal{L}([(w^\tau_{(1:H)})^\top, (w_{(H+1)})^\top]^\top)$. Now we are ready to state one of our main theorems.

Theorem 1. *Suppose $H \geq 2$, assumptions 1 and 2 hold. Assume that the function $h_X^{(H)}$ is real analytic. Then, with probability one over a randomly sampled δ, the following two statements hold:*

(i) (Gradient descent) if $g^t = \nabla_{w^t_{(H+1)}}\mathcal{L}(w^t)$ and $\eta_t = \frac{1}{L_H}\nu$ for $t \geq \tau$ with $L_H = \frac{L_\ell}{n}\sum_{i=1}^n \|[h^{(H)}(x_i, w^\tau_{(1:H)}),1]\|_2^2$, then for any $t > \tau$,

$$\mathcal{L}(w^t) - \mathcal{L}(w^*) \leq \frac{R^2 L_H}{2(t-\tau)}.$$

(ii) (SGD) if $\mathbb{E}[g^t|w^t] = \nabla_{w^t_{(H+1)}}\mathcal{L}(w^t)$ (almost surely), $\mathbb{E}[\|g^t\|^2] \leq G^2$ and $\eta_t = \bar{\eta}_t\nu$ for $t \geq \tau$ with $\bar{\eta}_t \in \mathbb{R}$ satisfying that $\bar{\eta}_t \geq 0$, $\sum_{t=\tau}^\infty \bar{\eta}_t^2 < \infty$ and $\sum_{t=\tau}^\infty \bar{\eta}_t = \infty$, then for any $t > \tau$,

$$\mathbb{E}[\mathcal{L}(w^{t^*})] - \mathcal{L}(w^*) \leq \frac{R^2 + G^2\sum_{k=\tau}^t \bar{\eta}_k^2}{2\sum_{k=\tau}^t \bar{\eta}_k},$$

where $t^* \in \mathrm{argmin}_{k\in\{\tau,\tau+1,\dots,t\}} \mathcal{L}(w^k)$.

In particular, Theorem 1 part (ii) shows that if we choose $\bar{\eta}_t \sim O(1/\sqrt{t})$, we have $\lim_{t\to\infty} \frac{\sum_{k=\tau}^t \bar{\eta}_k^2}{\sum_{k=\tau}^t \bar{\eta}_k} = 0$ and the optimality gap becomes

$$\mathbb{E}[\mathcal{L}(w^{t^*})] - \mathcal{L}(w^*) = \tilde{O}(1/\sqrt{t}).$$

5.4 Example

As a concrete example that satisfies all the conditions in theorem 1, we consider full-connected deep networks using softplus activation with a wide last hidden layer. In the case of fully-connected networks, the output of the last hidden layer can be simplified to

$$h_X^{(H)}(w_{(1:H)})_{ij} = h^{(H)}(x_i, w_{(1:H)}))_j \in \mathbb{R}, \qquad (1)$$

where $h^{(l)}(x_i, w_{(1:l)}) \in \mathbb{R}^{1\times m_l}$ is defined by

$$h^{(l)}(x_i, w_{(1:l)}) = \sigma(h^{(l-1)}(x_i, w_{(1:l-1)})W^{(l)} + b^{(l)}) \quad (2)$$

for $l = 1,2,\dots,H$ with $h^{(0)}(x_i, w_{(1:0)}) := x_i^\top \in \mathbb{R}^{1\times m_x}$. Here, $W^{(l)} \in \mathbb{R}^{m_{l-1}\times m_l}$ and $b^{(l)} \in \mathbb{R}^{1\times m_l}$ are the weight matrix and the bias term of the l-th hidden layer. Also, m_l represents the number of neurons at the l-th hidden layer. Since $h_X^{(H)}$ is the composition of affine functions and real analytic activation functions (i.e., softplus activation σ), the function $h_X^{(H)}$ is real analytic.

In theorem 2, we show that the expressivity condition is also satisfied for fully-connected networks, for training datasets that satisfy the following input distinguishability assumption.

Assumption 3. (Input distinguishability) $\|x_i\|^2 - x_i^\top x_j > 0$ *for any $x_i, x_j \in S_x$ with $i \neq j$.*

Theorem 2. *Suppose assumption 3 hold. Assume that $h_X^{(H)}$ is defined by equations* (1)-(2) *with softplus activation σ and $H \geq 2$ such that $\min(m_1,\dots,m_{H-1}) \geq \min(m_x, n)$, and $m_H \geq n$. Then, assumption 1 hold true.*

In Theorem 2, the case of $\min(m_1,\dots,m_{H-1}) = m_x$ is allowed. That is, all of the $1,2,\dots,H-1$-th hidden layers are allowed to be narrow (instead of wide) with the number of neurons to be m_x, which is typically smaller than

n. A previous paper recently proved that gradient descent finds a global minimum in a lazy training regime (i.e., the regime where NTK approximately remains unchanged during training) with $d = \tilde{\Omega}(m_y n + m_x H^2 + H^5)$ (Kawaguchi and Huang 2019). In contrast, Theorem 2 only requires $d \geq (m_y + m_x)n + m_x^2 H$ and allows NTK to change significantly during training.

Assumption 3 used in Theorem 2 can be easily satisfied, for example, by normalizing the input features for $x_1, \ldots, x_n$ so that $\|x_i\|^2 = \|x_j\|^2$. With the normalization, the condition is satisfied as long as $\|x_i - x_j\|^2 > 0$ for $i \neq j$ since $\frac{1}{2}\|x_i - x_j\|^2 = \|x_i\|^2 - x_i^\top x_j$. In general, normalization is not necessary, for example, orthogonality on x_i and x_j along with $x_i \neq 0$ satisfies the condition.

5.5 Global Convergence with Lazy Training in the Second Phase

In the previous section, we did not assume the rank preservation condition. Instead, we considered the special learning rate η_t in the second phase to keep submatrices of the kernel matrix $\mathbf{K}(w)$ unchanged during the second phase ($t \geq \tau$). In this section, we show that algorithm 1 can still ensure the global convergence with a standard uniform learning rate $\eta_t = \frac{2\bar{\eta}}{L}\mathbf{1}_d$, as long as the rank preservation condition is satisfied.

Theorem 3. *Let $\eta_t = \frac{2\bar{\eta}}{L}\mathbf{1}_d$ with $\bar{\eta} \in \mathbb{R}$ for $t \geq \tau$. Suppose that $H \geq 2$ and the following three assumptions hold:*

- *Assumption 1 (expressivity condition)*
- *Assumption 2, along with $\|\nabla\mathcal{L}(w) - \nabla\mathcal{L}(w')\| \leq L\|w - w'\|$ for all w, w' in the domain of $\mathcal{L}$.*
- *(rank preserving condition) $\operatorname{rank}\left(\mathbf{K}(w^k)\right) \geq \operatorname{rank}\left(\mathbf{K}(w^\tau)\right)$ for for all $k \in \{\tau+1, \tau+2, \ldots, t\}$.*

Then, the following statement hold for any $t > \tau$:

$$\mathcal{L}(w^{t^*}) - \mathcal{L}(w^*) \leq \frac{1}{\sqrt{(t-\tau)+1}}\sqrt{\frac{L\bar{R}^2(\mathcal{L}(w^\tau) - \mathcal{L}(w^*))}{2\bar{\eta}(1-\bar{\eta})}}.$$

where $t^* \in \operatorname{argmin}_{k\in\{\tau,\tau+1,\ldots,t\}} \mathcal{L}(w^k)$ and $\bar{R} = \max_{\tau\leq k\leq t} \min_{\hat{\omega}^k \in \bar{Q}_k} \|(\nu \odot w^k) - \hat{\omega}^k\|$ with $\bar{Q}_k = \operatorname{argmin}_{\hat{\omega}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^n \ell\left(\sum_{j=1}^d \hat{\omega}_j \frac{\partial f(x_i, w^k)^\top}{\partial w_j}, y_i\right)$.

Theorem 3 shows that using the lazy training that preserves the rank of the NTK matrix during the second phase can ensure the global convergence for Algorithm 1. Therefore, the proposed two-phase training algorithm provides a new perspective for the lazy training regime. That is, NTK in Algorithm 1 is allowed to change significantly during the first phase training $t < \tau$ to learn the features or representation beyond the random features induced by the data-independent NTK at initialization. Our two-phase method allows the lazy training with the learned data-dependent NTK at time τ, which is often a better representation of practical dataset than the NTK at random initialization.

The property of the lazy training now depends on the quantities at time τ instead of time $t = 0$. For example, if we conduct the lazy training with over-parameterization, then the number of neurons required per layer depends on the residual error and the minimum eigenvalue of NTK at time τ, instead of time $t = 0$. This could potential lead to global convergence theory with weaker assumptions. Thus, the two-phase algorithm opens up a new direction of future research for applying the lazy training theory to the data-dependent NTK obtained at the end of first phase training. We can define the domain of $\mathcal{L}$ to be a sublevel set around an initial point to satisfy the Lipschitz continuity.

6 Proof Idea and Key Challenges

For global convergence for optimization of deep neural networks, recent results rely on different assumptions, such as over-parameterization and initial conditions on gradient dynamics. Those different assumptions are essentially used in proofs to enforce the full rankness of the NTK matrix and the corresponding feature matrix during training. If the feature matrix is of full rank, then the global convergence is ensured (and the convergence rate depends on the minimum eigenvalue of the NTK matrix).

In order to apply our theory for practical settings, we want *data-dependency* in two key aspects. First, we want the feature matrix to be data-dependent and to change significantly during training, in order to learn data-dependent features. In contrast, the various assumptions in previous works essentially make the feature matrix to be approximately data-independent. Second, we want the global convergence results to hold data-dependently for a certain class of practical datasets. Instead, previous global convergence results need to hold data-independently, or for linearly-separable datasets or synthetic datasets generated by simple models (e.g., Gaussian mixtures). Because of these differences, there needs to be new proof strategies instead of adopting previous assumptions and their proof ideas.

6.1 Proof for General Networks

The first step in our proof is to show that the feature matrix is of full rank with probability one over random entries of the parameter vector. The global convergence then follows from the full rankness (as shown in the complete proof in Appendix).

A challenge in the proof for the general case is to make the right assumption as discussed above. If we assume significant over-parameterization, then proving the full rankness is relatively easy, but it limits the applicability. Indeed, we want to allow deep networks to have narrow layers.

Although the entries of the parameter vector after random perturbation have independent components, the entries of the feature matrix are dependent on each other. Indeed, the entries of the feature matrix are the outputs of nonlinear and non-convex functions of the entries of the parameter vector. Therefore, we cannot use elementary facts from linear algebra and random matrix theory with i.i.d. entries to prove the full rankness of the feature matrix.

Instead, we take advantage of the fact on the zero set of a real analytic function: if a function is real analytic and not identically zero, then the Lebesgue measure of its zero set is zero (Mityagin 2015). To utilize this fact, we define a func-

tion $\varphi(w_{(1:H)}) = \det(h_X^{(H)}(w_{(1:H)})h_X^{(H)}(w_{(1:H)})^\top)$. We then observe that φ is real analytic since $h_X^{(H)}$ is assumed to be real analytic and a composition of real analytic functions is real analytic. Furthermore, since the rank of $h_X^{(H)}(w_{(1:H)})$ and the rank of the Gram matrix are equal, we have that $\{w_{(1:H)} \in \mathbb{R}^{d_{1:H}} : \mathrm{rank}(h_X^{(H)}(w_{(1:H)})) \neq n\} = \{w_{(1:H)} \in \mathbb{R}^{d_{1:H}} : \varphi(w_{(1:H)}) = 0\}$. Since φ is analytic, if φ is not identically zero ($\varphi \neq 0$), the Lebesgue measure of its zero set $\{w_{(1:H)} \in \mathbb{R}^{d_{1:H}} : \varphi(w_{(1:H)}) = 0\}$ is zero. Therefore, if $\varphi(w_{(1:H)}) \neq 0$ for some $w_{(1:H)} \in \mathbb{R}^{d_{1:H}}$, the Lebesgue measure of the set

$$\{w_{(1:H)} \in \mathbb{R}^{d_{1:H}} : \mathrm{rank}(h_X^{(H)}(w_{(1:H)})) \neq n\}$$

is zero. Then, from the full rankness of the feature (sub)matrix $h_X^{(H)}(w_{(1:H)})$, we can ensure the global convergence, as in the previous papers with over-parameterization and neural tangent kernel.

Based on the above proof idea, as long as there exists a $w_{(1:H)} \in \mathbb{R}^{d_{1:H}}$ such that $\varphi(w_{(1:H)}) \neq 0$, then we can conclude the desired global convergence. The key observation is that this condition is a time-independent and easily verifiable condition in practice. This condition is defined as assumption 1. We verify that assumption 1 holds in experiments numerically for deep ResNets and in theory for fully-connected networks. We complete the proof in Appendix.

6.2 Proof for Fully-Connected Networks

Without our result on the general case, a challenge to prove the global convergence of fully-connected networks lies in the task of dealing with narrow hidden layers; i.e., in the case of $\min(m_1, \dots, m_{H-1}) = m_x < n$. In the case of $\min(m_1, \dots, m_{H-1}) \geq n$, it is easy to see that k-th layer with $m_k \geq n$ can preserve rank n for the corresponding matrices. In the case of $\min(m_1, \dots, m_{H-1}) = m_x < n$, however, it cannot preserve rank n, and deriving the global convergence is non-trivial.

With our result for the general case, however, our only remaining task is to show the existence of a $w_{(1:H)} \in \mathbb{R}^{d_{1:H}}$ such that $\varphi(w_{(1:H)}) \neq 0$. Accordingly, we complete the proof in appendix by constructing such a $w_{(1:H)} \in \mathbb{R}^{d_{1:H}}$ for the fully-connected networks with narrow layers.

7 Experiments

In this section, we study the empirical aspect of our method. The network model we work with is the standard (convolutional) pre-activation ResNet with 18 layers (He et al. 2016).

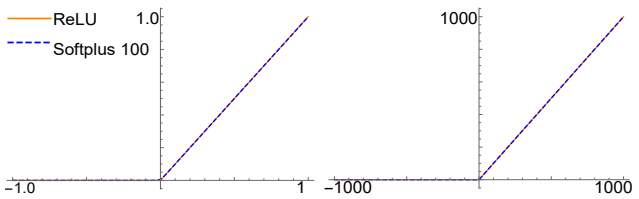


Figure 1: ReLU versus Softplus with $\varsigma = 100$.

To satisfy all the assumptions in our theory, we added a fully-connected last hidden layer of cn neurons with a small constat $c = 1.1$ and set the nonlinear activation functions of all layers to be softplus $\sigma(z) = \ln(1 + \exp(\varsigma z))/\varsigma$ with $\varsigma = 100$, as a real analytic function that approximates the ReLU activation. This approximation is of high accuracy as shown in Figure 1. As discussed above, Proposition 1 implies that the function $h_X^{(H)}$ for the ResNet is real analytic. The loss function we work with is cross-entropy loss, which satisfies assumption 2. Therefore, the only assumption in theorem 1 that is left to verify is assumption 1. In the following subsection, we numerically verify assumption 1.

7.1 Verification of Expressivity Condition

Assumption 1 assumes that the network satisfy expressivity condition, which only requires an existence of a $w_{(1:H)}$ such that $\varphi(w_{(1:H)}) \neq 0$. Here, $\varphi(w_{(1:H)}) \neq 0$ is implied by $\mathrm{rank}([h_X^{(H)}(w_{(1:H)}), \mathbf{1}_n]) = n$. In other words, if we find one $w_{(1:H)}$ with $\mathrm{rank}([h_X^{(H)}(w_{(1:H)}), \mathbf{1}_n]) = n$, then assumption 1 is ensured to hold true. A simple way to find such a $w_{(1:H)}$ is to randomly sample a single $w_{(1:H)}$ and check the condition of $\mathrm{rank}([h_X^{(H)}(w_{(1:H)}), \mathbf{1}_n]) = n$.

Table 1 column 3 summarizes the results of the verification of assumption 1 for various datasets. Here, we used a randomly sampled $w_{(1:H)}$ returned from the default initialization of the ResNet with version 1.4.0. of PyTorch (Paszke et al. 2019) by setting random seed to be 1. This initialization is based on the implementation of (He et al. 2015). The condition of $\mathrm{rank}([h_X^{(H)}(w_{(1:H)}), \mathbf{1}_n]) = n$ was checked by using *numpy.linalg.matrix_rank* in NumPy version 1.18.1 with the default option (i.e., without any arguments except the matrix $[h_X^{(H)}(w_{(1:H)}), \mathbf{1}_n]$), which uses the standard method from (Press et al. 2007).

7.2 Performance

In the following, we compare the generalization performances of the two-phase training algorithm over different hyper-parameters' choices and with the baseline algorithm.

Experimental setting We fixed all hyper-parameters of the base algorithm a priori across all different datasets by using a standard hyper-parameter setting of SGD (following the setting of Kawaguchi and Lu 2020), instead of aiming for state-of-the-art test errors with a possible issue of overfitting to test and validation datasets (Dwork et al. 2015; Rao, Fung, and Rosales 2008). Concretely, we fixed the mini-batch size to be 64, the weight decay rate to be 10^{-5}, the momentum coefficient to be 0.9, the first phase learning rate to be $\eta_t = 0.01$ and the second phase learning rate to be $\eta_t = 0.01 \times [\mathbf{0}_{d_{1:H}}^\top, \mathbf{1}_{d_{H+1}}^\top]^\top$ to only train the last layer. The last epoch T was fixed a priori as $T = 100$ without data augmentation and $T = 400$ with data augmentation.

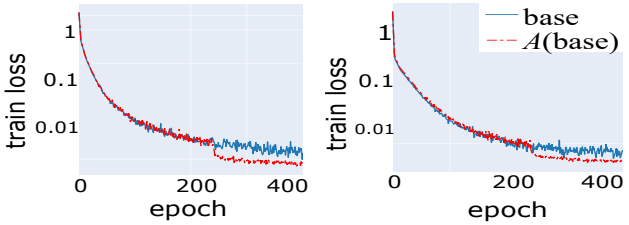
Choice of τ and δ Now we discuss the choice of hyper-parameters for the time of transition τ and for the size of the noise δ. Instead of potentially overfitting hyper-parameters

Dataset	# of training data	Expressivity Condition	Augmentation	Base	$\mathcal{A}$(base) with guarantee
MNIST	60000	Verified	No	0.41 (0.02)	0.38 (0.04)
			Yes	0.34 (0.03)	0.28 (0.04)
CIFAR-10	50000	Verified	No	13.99 (0.17)	13.57 (0.32)
			Yes	7.01 (0.18)	6.84 (0.16)
CIFAR-100	50000	Verified	No	41.43 (0.43)	40.78 (0.22)
			Yes	27.92 (0.36)	27.41 (0.58)
SVHN	73257	Verified	No	4.51 (0.04)	4.50 (0.09)
			Yes	4.32 (0.06)	4.16 (0.16)

Table 1: Test errors (%) of base and $\mathcal{A}$(base) with guarantee where the operator $\mathcal{A}$ maps any given first-order training algorithm to the two-phase version of the given algorithm with theoretical guarantees. The numbers indicate the mean test errors (and standard deviations in parentheses) over five random trials. The column of 'Augmentation' shows 'No' for no data augmentation, and 'Yes' for data augmentation. The expressivity condition (assumption 1) was numerically verified to all datasets.

δ_0 \ τ_0	0.4	0.5	0.6	0.8
0.0001	2.10 (0.14)	2.06 (0.07)	2.02 (0.09)	2.01 (0.05)
0.001	2.06 (0.07)	2.11 (0.12)	1.92 (0.06)	2.01 (0.12)
0.01	2.25 (0.11)	2.25 (0.17)	2.25 (0.11)	2.09 (0.08)

Table 2: Test errors (%) of $\mathcal{A}$(base) with guarantee for Kuzushiji-MNIST with different hyperparameters $\tau = \tau_0 T$ and $\delta = \delta_0 \epsilon$. The numbers indicate the mean test errors (and standard deviations in parentheses) over three random trials. The expressivity condition (Assumption 1) was numerically verified to hold for Kuzushiji-MNIST as well.



(a) CIFAR-10 (b) SVHN

Figure 2: Training loss of base and $\mathcal{A}$(base) with guarantee: the plots show the mean values over five random trials.

to each dataset, we used a different dataset, Kuzushiji-MNIST (Clanuwat et al. 2019), to fix all the hyper-parameters of Algorithm 1 across all other datasets. That is, we used Kuzushiji-MNIST with different hyperparameters' values $\tau_0 = 0.4, 0.5, 0.6, 0.8$ and $\delta_0 = 0.0001, 0.001, 0.01$, where $\tau = \tau_0 T$ and $\delta = \delta_0 \epsilon$. Here, $\epsilon \sim \mathcal{N}(\mathbf{0}, I_d)$ where I_d is the $d \times d$ identity matrix. Based on the results from Kuzushiji-MNIST in Table 2, we fixed $\tau_0 = 0.6$ and $\delta_0 = 0.001$ for all datasets.

Generalization and optimization comparison We now compare the performances between the base algorithm and its two-phase modified version. For generalization aspect, as shown in the last three columns of table 1, the modified algorithm improved the test errors consistently over the four datasets with and without data augmentation. This suggests that the modified version of the base algorithm is competitive with the base algorithm for generalization performance.

For optimization aspect, figure 2 shows that the two-phase training algorithm indeed improves training loss values of the base algorithm in the second phase without changing any hyper-parameters (e.g., learning rate and momentum) of the base algorithm, as expected from our theory. These results suggest that the two-phase algorithm can provide global convergence guarantees to a given base algorithm without hurting the generalization performance.

8 Conclusion

In this paper, we proposed a two-phase method that modifies any given first-order optimization algorithm to have global convergence guarantees without degrading practical performances of the given algorithm. The conditions for global convergence are mathematically proven to hold for fully-connected deep networks with wide last hidden layer (while all other layers are allowed to be narrow). The conditions are also numerically verified for deep ResNet with batch normalization under various standard classification datasets.

The two-phase training method opens up a new future research direction to study the use of the novel NTK regime with *learned representation* from data unlike the standard NTK regime near random initialization. Extending our theoretical analysis on a larger class of NTK with learned representation for global convergence and for generalization performance would be an interesting future direction.

As the global optimal parameters can often achieve near zero training loss, we could expect models trained by our modified algorithm to have the benefits from *terminal phase of training* (Papyan, Han, and Donoho 2020) potentially for better generalization performance, robustness, and interpretability. Verifying these benefits would be sensible directions for future work. An extension of our theory and algorithm to implicit deep learning (Bai, Kolter, and Koltun 2019; El Ghaoui et al. 2019; Kawaguchi 2021) would be another interesting future direction.

Acknowledgments

This work is partially supported by the Center of Mathematical Sciences and Applications at Harvard University. The authors thank Yiqiao Zhong, Mengyuan Yan for discussion.

References

Allen-Zhu, Z.; Li, Y.; and Liang, Y. 2019. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*, 6158–6169.

Arora, S.; Du, S. S.; Hu, W.; Li, Z.; Salakhutdinov, R. R.; and Wang, R. 2019a. On exact computation with an infinitely wide neural net. In *Advances in Neural Information Processing Systems*, 8141–8150.

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

Bai, S.; Kolter, J. Z.; and Koltun, V. 2019. Deep Equilibrium Models. *Advances in Neural Information Processing Systems* 32: 690–701.

Bresler, G.; and Nagaraj, D. 2020. A corrective view of neural networks: Representation, memorization and learning. *arXiv preprint arXiv:2002.00274* .

Bubeck, S.; Eldan, R.; Lee, Y. T.; and Mikulincer, D. 2020. Network size and weights size for memorization with two-layers neural networks. *arXiv preprint arXiv:2006.02855* .

Chizat, L.; and Bach, F. 2018. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in neural information processing systems*, 3036–3046.

Chizat, L.; Oyallon, E.; and Bach, F. 2019. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems*, 2937–2947.

Clanuwat, T.; Bober-Irizar, M.; Kitamoto, A.; Lamb, A.; Yamamoto, K.; and Ha, D. 2019. Deep learning for classical Japanese literature. In *NeurIPS Creativity Workshop 2019*.

Daniely, A. 2019. Neural networks learning and memorization with (almost) no over-parameterization. *arXiv preprint arXiv:1911.09873* .

Dou, X.; and Liang, T. 2020. Training neural networks as learning data-adaptive kernels: Provable representation and approximation benefits. *Journal of the American Statistical Association* 1–14.

Du, S.; Lee, J.; Li, H.; Wang, L.; and Zhai, X. 2019. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, 1675–1685.

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

Dwork, C.; Feldman, V.; Hardt, M.; Pitassi, T.; Reingold, O.; and Roth, A. 2015. The reusable holdout: Preserving validity in adaptive data analysis. *Science* 349(6248): 636–638.

El Ghaoui, L.; Gu, F.; Travacca, B.; Askari, A.; and Tsai, A. Y. 2019. Implicit deep learning. *arXiv preprint arXiv:1908.06315* 2.

Fang, C.; Lee, J. D.; Yang, P.; and Zhang, T. 2020. Modeling from Features: a Mean-field Framework for Over-parameterized Deep Neural Networks. *arXiv preprint arXiv:2007.01452* .

Ghorbani, B.; Mei, S.; Misiakiewicz, T.; and Montanari, A. 2019. Limitations of lazy training of two-layers neural network. In *Advances in Neural Information Processing Systems*, 9111–9121.

Goodfellow, I.; Bengio, Y.; and Courville, A. 2016. *Deep learning*. MIT press.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 1026–1034.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, 630–645. Springer.

Jacot, A.; Gabriel, F.; and Hongler, C. 2018. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, 8571–8580.

Jagtap, A. D.; Kawaguchi, K.; and Em Karniadakis, G. 2020. Locally adaptive activation functions with slope recovery for deep and physics-informed neural networks. *Proceedings of the Royal Society A* 476(2239): 20200334.

Jagtap, A. D.; Kawaguchi, K.; and Karniadakis, G. E. 2020. Adaptive activation functions accelerate convergence in deep and physics-informed neural networks. *Journal of Computational Physics* 404: 109136.

Kawaguchi, K. 2016. Deep Learning without Poor Local Minima. In *Advances in Neural Information Processing Systems*, 586–594.

Kawaguchi, K. 2021. On the Theory of Implicit Deep Learning: Global Convergence with Implicit Layers. In *International Conference on Learning Representations (ICLR)*.

Kawaguchi, K.; and Bengio, Y. 2019. Depth with nonlinearity creates no bad local minima in ResNets. *Neural Networks* 118: 167–174.

Kawaguchi, K.; and Huang, J. 2019. Gradient descent finds global minima for generalizable deep neural networks of practical sizes. In *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 92–99. IEEE.

Kawaguchi, K.; Huang, J.; and Kaelbling, L. P. 2019. Effect of depth and width on local minima in deep learning. *Neural computation* 31(7): 1462–1498.

Kawaguchi, K.; Kaelbling, L. P.; and Bengio, Y. 2017. Generalization in deep learning. *arXiv preprint arXiv:1710.05468* .

Kawaguchi, K.; and Lu, H. 2020. Ordered SGD: A New Stochastic Optimization Framework for Empirical Risk

Minimization. In *International Conference on Artificial Intelligence and Statistics*, 669–679.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature* 521(7553): 436–444.

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

Mei, S.; Misiakiewicz, T.; and Montanari, A. 2019. Mean-field theory of two-layers neural networks: dimension-free bounds and kernel limit. *arXiv preprint arXiv:1902.06015* .

Mei, S.; Montanari, A.; and Nguyen, P.-M. 2018. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences* 115(33): E7665–E7671.

Mityagin, B. 2015. The zero set of a real analytic function. *arXiv preprint arXiv:1512.07276* .

Montanari, A.; and Zhong, Y. 2020. The interpolation phase transition in neural networks: memorization and generalization under lazy training. *preprint arXiv:2007.12826* .

Papyan, V.; Han, X.; and Donoho, D. L. 2020. Prevalence of Neural Collapse during the terminal phase of deep learning training. *arXiv preprint arXiv:2008.08186* .

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, 8024–8035.

Poggio, T.; Kawaguchi, K.; Liao, Q.; Miranda, B.; Rosasco, L.; Boix, X.; Hidary, J.; and Mhaskar, H. 2017. Theory of Deep Learning III: explaining the non-overfitting puzzle. *arXiv preprint arXiv:1801.00173* .

Press, W. H.; Teukolsky, S. A.; Vetterling, W. T.; and Flannery, B. P. 2007. *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.

Rao, R. B.; Fung, G.; and Rosales, R. 2008. On the dangers of cross-validation. An experimental evaluation. In *Proceedings of the 2008 SIAM international conference on data mining*, 588–596. SIAM.

Safran, I.; and Shamir, O. 2016. On the quality of the initial basin in overspecified neural networks. In *International Conference on Machine Learning*, 774–782.

Soltanolkotabi, M.; Javanmard, A.; and Lee, J. D. 2018. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory* 65(2): 742–769.

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

Wei, C.; Lee, J. D.; Liu, Q.; and Ma, T. 2019. Regularization matters: Generalization and optimization of neural nets vs their induced kernel. In *Advances in Neural Information Processing Systems*, 9712–9724.

Yehudai, G.; and Shamir, O. 2019. On the power and limitations of random features for understanding neural networks. In *Advances in Neural Information Processing Systems*, 6598–6608.

Zou, D.; Cao, Y.; Zhou, D.; and Gu, Q. 2020. Gradient descent optimizes over-parameterized deep ReLU networks. *Machine Learning* 109(3): 467–492.