Optimization Theory for ReLU Neural Networks Trained with Normalization Layers

Yonatan Dukler [1] **Quanquan Gu** [2] **Guido Montúfar** [1 3 4]

Abstract

The success of deep neural networks is in part due to the use of normalization layers. Normalization layers like Batch Normalization, Layer Normalization and Weight Normalization are ubiquitous in practice, as they improve generalization performance and speed up training significantly. Nonetheless, the vast majority of current deep learning theory and non-convex optimization literature focuses on the un-normalized setting, where the functions under consideration do not exhibit the properties of commonly normalized neural networks. In this paper, we bridge this gap by giving the first global convergence result for two-layer neural networks with ReLU activations trained with a normalization layer, namely Weight Normalization. Our analysis shows how the introduction of normalization layers changes the optimization landscape and can enable faster convergence as compared with un-normalized neural networks.

1. Introduction

Dynamic normalization in the training of neural networks amounts to the application of an intermediate normalization procedure between layers of the network. Such methods have become ubiquitous in the training of neural nets since in practice they significantly improve the convergence speed and stability. This type of approach was popularized with the introduction of Batch Normalization (BN) (Ioffe and Szegedy, 2015) which implements a dynamic re-parametrization normalizing the first two moments of the outputs at each layer over mini-batches. A plethora of additional normalization methods followed BN, notably including Layer Normalization (LN) (Ba et al., 2016) and Weight Normalization (WN) (Salimans and Kingma, 2016). Despite the impressive empirical results and massive popularity of dynamic normalization methods, explaining their utility and proving that they converge when training with non-smooth, non-convex loss functions has remained an unsolved problem. In this paper we provide sufficient conditions on the data, initialization, and over-parametrization for dynamically normalized ReLU networks to converge to a global minimum of the loss function. For the theory we present we focus on WN, which is a widely used normalization layer in training of neural networks. WN was proposed as a method that emulates BN. It normalizes the input weight vector of each unit and separates the scale into an independent parameter. The WN re-parametrization is very similar to BN (see Section 2) and benefits from similar stability and convergence properties. Moreover, WN has the advantage of not requiring a batch setting, therefore considerably reducing the computational overhead that is imposed by BN (Gitman and Ginsburg, 2017).

When introducing normalization methods, the function parametrization defined by the network becomes scale invariant in the sense that re-scaling of the weights does not change the represented function. This re-scaling invariance changes the geometry of the optimization landscape drastically. To better understand this we analyze weight normalization in a given layer.

We consider the class of 2-layer ReLU neural networks which represent functions $f\colon \mathbb{R}^d \to \mathbb{R}$ parameterized by $(\mathbf{W}, \mathbf{c}) \in \mathbb{R}^{m\times d} \times \mathbb{R}^m$ as

$$f(\mathbf{x}; \mathbf{W}, \mathbf{c}) = \frac{1}{\sqrt{m}} \sum_{k=1}^{m} c_k \sigma(\mathbf{w}_k^\top \mathbf{x}). \tag{1.1}$$

Here we use the ReLU activation function $\sigma(s) = \max\{s, 0\}$ (Nair and Hinton, 2010), m denotes the width of the hidden layer, and the output is normalized accordingly by a factor $\sqrt{m}$. We investigate gradient descent training with WN for (1.1), which re-parametrizes the functions in

[1]Department of Mathematics, UCLA, Los Angeles, CA 90095. [2]Department of Computer Science, UCLA, Los Angeles, CA 90095. [3]Department of Statistics, UCLA, Los Angeles, CA 90095. [4]Max Planck Institute for Mathematics in the Sciences, 04103 Leipzig, Germany. Correspondence to: Yonatan Dukler <ydukler@math.ucla.edu>, Quanquan Gu <qgu@cs.ucla.edu>, Guido Montúfar <montufar@math.ucla.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

terms of $(\mathbf{V}, \mathbf{g}, \mathbf{c}) \in \mathbb{R}^{m\times d} \times \mathbb{R}^m \times \mathbb{R}^m$ as

$$f(\mathbf{x}; \mathbf{V}, \mathbf{g}, \mathbf{c}) = \frac{1}{\sqrt{m}} \sum_{k=1}^{m} c_k \sigma\left(g_k \cdot \frac{\mathbf{v}_k^\top \mathbf{x}}{\|\mathbf{v}_k\|_2} \right). \tag{1.2}$$

This gives a similar parametrization to (Du et al., 2018) that study convergence of gradient optimization of convolutional filters on Gaussian data. We consider a regression task, the L^2 loss, a random parameter initialization, and focus on the over-parametrized regime, meaning that $m > n$, where n is the number of training samples. Further, we make little to no assumptions about the data.

The neural network function class (1.1) has been studied in many papers including (Arora et al., 2019a; Du et al., 2019b; Wu et al., 2019; Zhang et al., 2019) along with other similar over-parameterized architectures (Allen-Zhu et al., 2019a; Du et al., 2018; Li and Liang, 2018). An exuberant series of recent works prove that feed-forward ReLU networks converge to zero training error when trained with gradient descent from random initialization. Nonetheless, to the best of our knowledge, there are no proofs that ReLU networks trained with *normalization* on general data converge to a global minimum. This is in part because normalization methods completely change the optimization landscape during training. Here we show that neural networks of the form given above converge at linear rate when trained with gradient descent and WN. The analysis is based on the over-parametrization of the networks, which allows for guaranteed descent while the gradient is non-zero.

For regression training, a group of papers studied the trajectory of the networks' predictions and showed that they evolve via a "neural tangent kernel" (NTK) as introduced by Jacot et al. (2018). The latter paper studies neural network convergence in the continuous limit of infinite width over-parametrization, while the works of (Arora et al., 2019a; Du et al., 2019b; Oymak and Soltanolkotabi, 2019; Wu et al., 2019; Zhang et al., 2019) analyze the finite width setting. For finite-width over-parameterized networks, the training evolution also exhibits a kernel that takes the form of a Gram matrix. In these works, the convergence rate is dictated by the least eigenvalue of the kernel. We build on this fact, and also on the general ideas of the proof of (Du et al., 2019b) and the refined work of (Arora et al., 2019a).

In this work we analyze neural network optimization with weight normalization layers. We rigorously derive the dynamics of weight normalization training and its convergence from the perspective of the neural tangent kernel. Compared with un-normalized training, we prove that normalized networks follow a modified kernel evolution that features a "length-direction" decomposition of the NTK. This leads to two convergence regimes in WN training and explains the utility of WN from the perspective of the NTK. In the settings considered, WN significantly reduces the amount of over-parametrization needed for provable convergence, as compared with un-normalized settings. Further, we present a more careful analysis that leads to improved over-parametrization bounds as compared with (Du et al., 2019b).

The main contributions of this work are:

- We prove the first general convergence result for 2-layer ReLU networks trained with a normalization layer and gradient descent. Our formulation does not assume the existence of a teacher network and has only very mild assumptions on the training data.
- We hypothesize the utility of normalization methods via a decomposition of the neural tangent kernel. In the analysis we highlight two distinct convergence regimes and show how Weight Normalization can be related to natural gradients and enable faster convergence.
- We show that finite-step gradient descent converges for all weight magnitudes at initialization. Further, we significantly reduce the amount of over-parametrization required for provable convergence as compared with un-normalized training.

The paper is organized as follows. In Section 2 we provide background on WN and derive key evolution dynamics of training in Section 3. We present and discuss our main results, alongside with the idea of the proof, in Section 4. We discuss related work in Section 5, and offer a discussion of our results and analysis in Section 6. Proofs are presented in the Appendix.

2. Weight Normalization

Here we give an overview of the WN procedure and review some known properties of normalization methods.

Notation We use lowercase, lowercase boldface, and uppercase boldface letters to denote scalars, vectors and matrices respectively. We denote the Rademacher distribution as $U\{1, -1\}$ and write $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for a Gaussian with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. Training points are denoted by $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ and parameters of the first layer by $\mathbf{v}_k \in \mathbb{R}^d$, $k = 1, \ldots, m$. We use $\sigma(x) \coloneqq \max\{x, 0\}$, and write $\|\cdot\|_2, \|\cdot\|_F$ for the spectral and Frobenius norms for matrices. $\lambda_{\min}(\mathbf{A})$ is used to denote the minimum eigenvalue of a matrix $\mathbf{A}$ and $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product. For a vector $\mathbf{v}$ denote the ℓ_2 vector norm as $\|\mathbf{v}\|_2$ and for a positive definite matrix $\mathbf{S}$ define the induced vector norm $\|\mathbf{v}\|_{\mathbf{S}} \coloneqq \sqrt{\mathbf{v}^\top \mathbf{S} \mathbf{v}}$. The projections of $\mathbf{x}$ onto $\mathbf{u}$ and $\mathbf{u}^\perp$ are defined as $\mathbf{x}^{\mathbf{u}} \coloneqq \frac{\mathbf{u}\mathbf{u}^\top \mathbf{x}}{\|\mathbf{u}\|_2^2}$, $\mathbf{x}^{\mathbf{u}^\perp} \coloneqq \left(\mathbf{I} - \frac{\mathbf{u}\mathbf{u}^\top}{\|\mathbf{u}\|_2^2}\right)\mathbf{x}$. Denote the indicator function of event A as $\mathbb{1}_A$ and for a

weight vector at time t, $\mathbf{v}_k(t)$, and data point $\mathbf{x}_i$ we denote $\mathbb{1}_{ik}(t) \coloneqq \mathbb{1}_{\{\mathbf{v}_k(t)^\top \mathbf{x}_i \geq 0\}}$.

WN procedure For a single neuron $\sigma(\mathbf{w}^\top \mathbf{x})$, WN re-parametrizes the weight $\mathbf{w} \in \mathbb{R}^d$ in terms of $\mathbf{v} \in \mathbb{R}^d$, $g \in \mathbb{R}$ as

$$\mathbf{w}(\mathbf{v}, g) = g \cdot \frac{\mathbf{v}}{\|\mathbf{v}\|_2}, \quad \sigma\left(g \cdot \frac{\mathbf{v}^\top \mathbf{x}}{\|\mathbf{v}\|_2}\right). \tag{2.1}$$

This decouples the magnitude and direction of each weight vector (referred as the "length-direction" decomposition). In comparison, for BN each output $\mathbf{w}^\top \mathbf{x}$ is normalized according to the average statistics in a batch. We can draw the following analogy between WN and BN if the inputs $\mathbf{x}_i$ are centered ($\mathbb{E}\mathbf{x} = \mathbf{0}$) and the covariance matrix is known ($\mathbb{E}\mathbf{x}\mathbf{x}^\top = \mathbf{S}$). In this case, batch training with BN amounts to

$$\begin{aligned} \sigma\left(\gamma \cdot \frac{\mathbf{w}^\top \mathbf{x}}{\sqrt{\mathbb{E}_{\mathbf{x}}(\mathbf{w}^\top \mathbf{x}\mathbf{x}^\top \mathbf{w})}}\right) &= \sigma\left(\gamma \cdot \frac{\mathbf{w}^\top \mathbf{x}}{\sqrt{\mathbf{w}^\top \mathbf{S}\mathbf{w}}}\right) \\ &= \sigma\left(\gamma \cdot \frac{\mathbf{w}^\top \mathbf{x}}{\|\mathbf{w}\|_{\mathbf{S}}}\right). \end{aligned} \tag{2.2}$$

From this prospective, WN is a special case of (2.2) with $\mathbf{S} = \mathbf{I}$ (Kohler et al., 2019; Salimans and Kingma, 2016).

Properties of WN We start by giving an overview of known properties of WN that will be used to derive the gradient flow dynamics of WN training.

For re-parametrization (2.1) of a network function f that is initially parameterized with a weight $\mathbf{w}$, the gradient $\nabla_{\mathbf{w}} f$ relates to the gradients $\nabla_{\mathbf{v}} f$, $\frac{\partial f}{\partial g}$ by the identities

$$\nabla_{\mathbf{v}} f = \frac{g}{\|\mathbf{v}\|_2} (\nabla_{\mathbf{w}} f)^{\mathbf{v}^\perp}, \quad \frac{\partial f}{\partial g} = (\nabla_{\mathbf{w}} f)^{\mathbf{v}}.$$

This implies that $\nabla_{\mathbf{v}} f \cdot \mathbf{v} = 0$ for each input $\mathbf{x}$ and parameter $\mathbf{v}$. For gradient flow, this orthogonality results in $\|\mathbf{v}(0)\|_2 = \|\mathbf{v}(t)\|_2$ for all t. For gradient descent (with step size η) the discretization in conjunction with orthogonality leads to increasing parameter magnitudes during training (Arora et al., 2019b; Hoffer et al., 2018; Salimans and Kingma, 2016), as illustrated in Figure 1,

$$\|\mathbf{v}(s+1)\|_2^2 = \|\mathbf{v}(s)\|_2^2 + \eta^2 \|\nabla_{\mathbf{v}} f\|_2^2 \geq \|\mathbf{v}(s)\|_2^2. \tag{2.3}$$

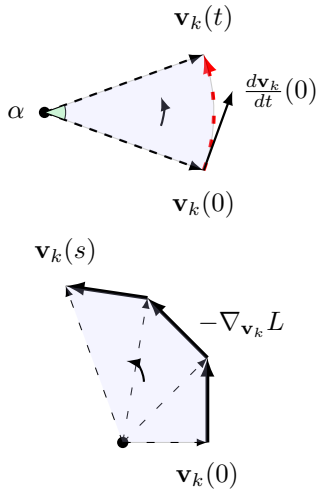

Figure 1. WN updates for gradient flow and gradient descent. For gradient flow, the norm of the weights are preserved, i.e., $\|\mathbf{v}_k(0)\|_2 = \|\mathbf{v}_k(t)\|_2$ for all $t > 0$. For gradient descent, the norm of the weights $\|\mathbf{v}_k(s)\|_2$ is increasing with s.

Problem Setup We analyze (1.1) with WN training (1.2), so that

$$f(\mathbf{x}; \mathbf{V}, \mathbf{c}, \mathbf{g}) = \frac{1}{\sqrt{m}} \sum_{k=1}^{m} c_k \sigma\left(g_k \cdot \frac{\mathbf{v}_k^\top \mathbf{x}}{\|\mathbf{v}_k\|_2}\right).$$

We take an initialization in the spirit of (Salimans and Kingma, 2016):

$$\begin{aligned} \mathbf{v}_k(0) \sim N(0, \alpha^2 \mathbf{I}), \quad c_k \sim U\{-1, 1\}, \\ \text{and} \quad g_k(0) = \|\mathbf{v}_k(0)\|_2 / \alpha. \end{aligned} \tag{2.4}$$

Where α^2 is the variance of $\mathbf{v}_k$ at initialization. The initialization of $g_k(0)$ is therefore taken to be independent of α. We remark that the initialization (2.4) gives the same initial output distribution as in methods that study the un-normalized network class (1.1). The parameters of the network are optimized using the training data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ with respect to the square loss

$$L(f) = \frac{1}{2} \sum_{i=1}^{n} (f(\mathbf{x}_i) - y_i)^2 = \frac{1}{2} \|\mathbf{f} - \mathbf{y}\|_2^2, \tag{2.5}$$

where $\mathbf{f} = (f_1, \ldots, f_n)^\top = (f(\mathbf{x}_1), \ldots, f(\mathbf{x}_n))^\top$ and $\mathbf{y} = (y_1, \ldots, y_n)^\top$.

3. Evolution Dynamics

We present the gradient flow dynamics of training (2.5) to illuminate the modified dynamics of WN as compared with vanilla gradient descent. In Appendix C we tackle gradient descent training with WN where the predictions' evolution vector $\frac{d\mathbf{f}}{dt}$ is replaced by the finite difference $\mathbf{f}(s+1) - \mathbf{f}(s)$.

For gradient flow, each parameter is updated in the negative direction of the partial derivative of the loss with respect to that parameter. The optimization dynamics give

$$\frac{d\mathbf{v}_k}{dt} = -\frac{\partial L}{\partial \mathbf{v}_k}, \quad \frac{dg_k}{dt} = -\frac{\partial L}{\partial g_k}. \tag{3.1}$$

We consider the case where we fix the top layer parameters c_k during training. In the over-parameterized settings we consider, the dynamics of c_k and g_k turn out to be equivalent.

To quantify convergence, we monitor the time derivative of the i-th prediction, which is computed via the chain rule as

$$\frac{\partial f_i}{\partial t} = \sum_{k=1}^{m} \frac{\partial f_i}{\partial \mathbf{v}_k}\frac{d\mathbf{v}_k}{dt} + \frac{\partial f_i}{\partial g_k}\frac{dg_k}{dt}.$$

Substituting (3.1) into the i-th prediction evolution and grouping terms yields

$$\frac{\partial f_i}{\partial t} = -\underbrace{\sum_{k=1}^{m} \frac{\partial f_i}{\partial \mathbf{v}_k}\frac{\partial L}{\partial \mathbf{v}_k}}_{T_{\mathbf{v}}^i} - \underbrace{\sum_{k=1}^{m} \frac{\partial f_i}{\partial g_k}\frac{\partial L}{\partial g_k}}_{T_g^i}. \tag{3.2}$$

The gradients of f_i and L with respect to $\mathbf{v}_k$ are written explicitly as

$$\frac{\partial f_i}{\partial \mathbf{v}_k}(t) = \frac{1}{\sqrt{m}}\frac{c_k \cdot g_k(t)}{\|\mathbf{v}_k(t)\|_2} \cdot \mathbf{x}_i^{\mathbf{v}_k(t)^\perp} \mathbb{1}_{ik}(t),$$
$$\frac{\partial L}{\partial \mathbf{v}_k}(t) = \frac{1}{\sqrt{m}}\sum_{i=1}^{n}(f_i(t)-y_i)\frac{c_k \cdot g_k(t)}{\|\mathbf{v}_k(t)\|_2}\mathbf{x}_i^{\mathbf{v}_k(t)^\perp}\mathbb{1}_{ik}(t).$$

Defining the $\mathbf{v}$-orthogonal Gram matrix $\mathbf{V}(t)$ as

$$\mathbf{V}_{ij}(t) = \frac{1}{m}\sum_{k=1}^{m}\left(\frac{\alpha c_k \cdot g_k(t)}{\|\mathbf{v}_k(t)\|_2}\right)^2 \langle \mathbf{x}_i^{\mathbf{v}_k(t)^\perp}, \mathbf{x}_j^{\mathbf{v}_k(t)^\perp}\rangle \mathbb{1}_{ik}(t)\mathbb{1}_{jk}(t), \tag{3.3}$$

we can compute $T_{\mathbf{v}}^i$ as

$$T_{\mathbf{v}}^i(t) = \sum_{j=1}^{n}\frac{\mathbf{V}_{ij}(t)}{\alpha^2}(f_j(t)-y_j).$$

Note that $\mathbf{V}(t)$ is the induced neural tangent kernel (Jacot et al., 2018) for the parameters $\mathbf{v}$ of WN training. While it resembles the Gram matrix $\mathbf{H}(t)$ studied in (Arora et al., 2019a), here we obtain a matrix that is not piece-wise constant in $\mathbf{v}$ since the data points are projected onto the orthogonal component of $\mathbf{v}$. We compute $T_{\mathbf{g}}^i$ in (3.2) analogously. The associated derivatives with respect to g_k are

$$\frac{\partial f_i}{\partial g_k}(t) = \frac{1}{\sqrt{m}}\frac{c_k}{\|\mathbf{v}_k(t)\|_2}\sigma(\mathbf{v}_k(t)^\top \mathbf{x}_i),$$
$$\frac{\partial L}{\partial g_k}(t) = \frac{1}{\sqrt{m}}\sum_{j=1}^{n}(f_j(t)-y_j)\frac{c_k}{\|\mathbf{v}_k(t)\|_2}\sigma(\mathbf{v}_k(t)^\top \mathbf{x}_j),$$

and we obtain

$$T_{\mathbf{g}}^i(t) = \sum_{k=1}^{m}\frac{1}{m}\sum_{j=1}^{n}\frac{c_k^2(f_j(t)-y_j)}{\|\mathbf{v}_k(t)\|_2^2}\sigma(\mathbf{v}_k(t)^\top \mathbf{x}_j)\sigma(\mathbf{v}_k(t)^\top \mathbf{x}_i).$$

Given that $c_k^2 = 1$, define $\mathbf{G}(t)$ as

$$\mathbf{G}_{ij}(t) = \frac{1}{m}\sum_{k=1}^{m}\frac{\sigma(\mathbf{v}_k(t)^\top \mathbf{x}_i)\sigma(\mathbf{v}_k(t)^\top \mathbf{x}_j)}{\|\mathbf{v}_k(t)\|_2^2} \tag{3.4}$$

hence we can write

$$T_{\mathbf{g}}^i(t) = \sum_{j=1}^{n}\mathbf{G}_{ij}(t)(f_j(t)-y_j).$$

Combining $T_{\mathbf{v}}$ and $T_{\mathbf{g}}$, the full evolution dynamics are given by

$$\frac{d\mathbf{f}}{dt} = -\left(\frac{\mathbf{V}(t)}{\alpha^2} + \mathbf{G}(t)\right)(\mathbf{f}(t)-\mathbf{y}). \tag{3.5}$$

Denote $\mathbf{\Lambda}(t) := \frac{\mathbf{V}(t)}{\alpha^2} + \mathbf{G}(t)$ and write $\frac{d\mathbf{f}}{dt} = -\mathbf{\Lambda}(t)(\mathbf{f}(t)-\mathbf{y})$. We note that $\mathbf{V}(0), \mathbf{G}(0)$, defined in (3.3), (3.4), are independent of α:

Observation 1 (α independence)**.** *For initialization (2.4) and $\alpha > 0$ the Gram matrices $\mathbf{V}(0), \mathbf{G}(0)$ are independent of α.*

This fact is proved in Appendix A. When training the neural network in (1.1) without WN (see Arora et al., 2019a; Du et al., 2019b; Zhang et al., 2019), the corresponding neural tangent kernel $\mathbf{H}(t)$ is defined by $\frac{\partial f_i}{\partial t} = \sum_{k=1}^{m}\frac{\partial f_i}{\partial \mathbf{w}_k}\frac{d\mathbf{w}_k}{dt} = -\sum_{k=1}^{m}\frac{\partial f_i}{\partial \mathbf{w}_k}\frac{\partial L}{\partial \mathbf{w}_k} = -\sum_{j=1}^{n}\mathbf{H}_{ij}(t)(f_j-y_j)$ and takes the form

$$\mathbf{H}_{ij}(t) = \frac{1}{m}\sum_{k=1}^{m}\mathbf{x}_i^\top \mathbf{x}_j \mathbb{1}_{ik}(t)\mathbb{1}_{jk}(t). \tag{3.6}$$

The analysis presented above shows that vanilla and WN gradient descent are related as follows.

Proposition 1. *Define $\mathbf{V}(0)$, $\mathbf{G}(0)$, and $\mathbf{H}(0)$ as in (3.3), (3.4), and (3.6) respectively. then for all $\alpha > 0$,*

$$\mathbf{V}(0) + \mathbf{G}(0) = \mathbf{H}(0).$$

Thus, for $\alpha = 1$,

$$\frac{\partial \mathbf{f}}{\partial t} = -\mathbf{\Lambda}(0)(\mathbf{f}(0)-\mathbf{y}) = -\mathbf{H}(0)(\mathbf{f}(0)-\mathbf{y}).$$

That is, WN decomposes the NTK in each layer into a length and a direction component. We refer to this as the "length-direction decoupling" of the NTK, in analogy to (2.1). From

the proposition, normalized and un-normalized training kernels initially coincide if $\alpha = 1$. We hypothesize that the utility of normalization methods can be attributed to the modified NTK $\mathbf{\Lambda}(t)$ that occurs when the WN coefficient, α, deviates from 1. For $\alpha \gg 1$ the kernel $\mathbf{\Lambda}(t)$ is dominated by $\mathbf{G}(t)$, and for $\alpha \ll 1$ the kernel $\mathbf{\Lambda}(t)$ is dominated by $\mathbf{V}(t)$. We elaborate on the details of this in the next section. In our analysis we will study the two regimes $\alpha > 1$ and $\alpha < 1$ in turn.

4. Main Convergence Theory

In this section we discuss our convergence theory and main results. From the continuous flow (3.5), we observe that the convergence behavior is described by $\mathbf{V}(t)$ and $\mathbf{G}(t)$. The matrices $\mathbf{V}(t)$ and $\mathbf{G}(t)$ are positive semi-definite since they can be shown to be covariance matrices. This implies that the least eigenvalue of the evolution matrix $\mathbf{\Lambda}(t) = \frac{1}{\alpha^2}\mathbf{V}(t) + \mathbf{G}(t)$ is bounded below by the least eigenvalue of each kernel matrix,

$$\lambda_{\min}(\mathbf{\Lambda}(t)) \geq \max\{\lambda_{\min}(\mathbf{V}(t))/\alpha^2, \lambda_{\min}(\mathbf{G}(t))\}.$$

For finite-step gradient descent, a discrete analog of evolution (3.5) holds. However, the discrete case requires additional care in ensuring dominance of the driving gradient terms. For gradient flow, it is relatively easy to see linear convergence is attained by relating the rate of change of the loss to the magnitude of the loss. Suppose that for all $t \geq 0$,

$$\lambda_{\min}\big(\mathbf{\Lambda}(t)\big) \geq \omega/2, \quad \text{with } \omega > 0. \tag{4.1}$$

Then the change in the regression loss is written as

$$\begin{aligned}\frac{d}{dt}\|\mathbf{f}(t) - \mathbf{y}\|_2^2 &= 2(\mathbf{f}(t) - \mathbf{y})^\top \frac{d\mathbf{f}(t)}{dt} \\ &= -2(\mathbf{f}(t) - \mathbf{y})^\top \mathbf{\Lambda}(t)(\mathbf{f}(t) - \mathbf{y}) \\ &\overset{(4.1)}{\leq} -\omega\|\mathbf{f}(t) - \mathbf{y}\|_2^2.\end{aligned}$$

Integrating this time derivative and using the initial conditions yields

$$\|\mathbf{f}(t) - \mathbf{y}\|_2^2 \leq \exp(-\omega t)\|\mathbf{f}(0) - \mathbf{y}\|_2^2,$$

which gives linear convergence. The focus of our proof is therefore showing that (4.1) holds throughout training.

By Observation 1 we have that $\mathbf{V}$ and $\mathbf{G}$ are independent of the WN coefficient α (α only appears in the $1/\alpha^2$ scaling of $\mathbf{\Lambda}$). This suggests that the kernel $\mathbf{\Lambda}(t) = \frac{1}{\alpha^2}\mathbf{V}(t) + \mathbf{G}(t)$ can be split into two regimes: When $\alpha < 1$ the kernel is dominated by the first term $\frac{1}{\alpha^2}\mathbf{V}$, and when $\alpha > 1$ the kernel is dominated by the second term $\mathbf{G}$. We divide our convergence result based on these two regimes.

In each regime, (4.1) holds if the corresponding dominant kernel, $\mathbf{V}(t)$ or $\mathbf{G}(t)$, maintains a positive least eigenvalue. Having a least eigenvalue that is bounded from 0 gives a convex-like property that allows us to prove convergence. To ensure that condition (4.1) is satisfied, for each regime we show that the corresponding dominant kernel is "anchored" (remains close) to an auxiliary Gram matrix which we define in the following for $\mathbf{V}$ and $\mathbf{G}$.

Define the auxiliary $\mathbf{v}$-orthogonal and $\mathbf{v}$-aligned Gram matrices $\mathbf{V}^\infty, \mathbf{G}^\infty$ as

$$\mathbf{V}^\infty_{ij} \coloneqq \mathbb{E}_{\mathbf{v}\sim N(0,\alpha^2\mathbf{I})}\, \langle \mathbf{x}_i^{\mathbf{v}^\perp}, \mathbf{x}_j^{\mathbf{v}^\perp}\rangle \mathbb{1}_{ik}(0)\mathbb{1}_{jk}(0), \tag{4.2}$$

$$\mathbf{G}^\infty_{ij} \coloneqq \mathbb{E}_{\mathbf{v}\sim N(0,\alpha^2\mathbf{I})}\, \langle \mathbf{x}_i^{\mathbf{v}}, \mathbf{x}_j^{\mathbf{v}}\rangle \mathbb{1}_{ik}(0)\mathbb{1}_{jk}(0). \tag{4.3}$$

For now, assume that $\mathbf{V}^\infty$ and $\mathbf{G}^\infty$ are positive definite with a least eigenvalue bounded below by ω (we give a proof sketch below). In the convergence proof we will utilize overparametrization to ensure that $\mathbf{V}(t), \mathbf{G}(t)$ concentrate to their auxiliary versions so that they are also positive definite with a least eigenvalue that is greater than $\omega/2$. The precise formulations are presented in Lemmas B.4 and B.5 that are relegated to Appendix B.

To prove our convergence results we make the assumption that the $\mathbf{x}_i$s have bounded norm and are not parallel.

Assumption 1 (Normalized non-parallel data)**.** *The data points $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$ satisfy $\|\mathbf{x}_i\|_2 \leq 1$ and for each index pair $i \neq j$, $\mathbf{x}_i \neq \beta \cdot \mathbf{x}_j$ for all $\beta \in \mathbb{R} \setminus \{0\}$.*

In order to simplify the presentation of our results, we assume that the input dimension d is not too small, whereby $d \geq 50$ suffices. This is not essential for the proof. Specific details are provided in Appendix A.

Assumption 2. *For data $\mathbf{x}_i \in \mathbb{R}^d$ assume that $d \geq 50$.*

Both assumptions can be easily satisfied by pre-processing, e.g., normalizing and shifting the data, and adding zero coordinates if needed.

Given Assumption 1, $\mathbf{V}^\infty, \mathbf{G}^\infty$ are shown to be positive definite.

Lemma 4.1. *Fix training data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ satisfying Assumption 1. Then the $\mathbf{v}$-orthogonal and $\mathbf{v}$-aligned Gram matrices $\mathbf{V}^\infty$ and $\mathbf{G}^\infty$, defined as in (4.2) and (4.3), are strictly positive definite. We denote the least eigenvalues $\lambda_{\min}(\mathbf{V}^\infty) =: \lambda_0$, $\lambda_{\min}(\mathbf{G}^\infty) =: \mu_0$.*

Proof sketch Here we sketch the proof of Lemma 4.1. The main idea, is the same as (Du et al., 2019b), is to regard the auxiliary matrices $\mathbf{V}^\infty, \mathbf{G}^\infty$ as the covariance matrices of linearly independent operators. For each data point $\mathbf{x}_i$, define $\phi_i(\mathbf{v}) \coloneqq \mathbf{x}_i^{\mathbf{v}^\perp} \mathbb{1}_{\{\mathbf{x}_i^\top \mathbf{v} \geq 0\}}$. The Gram matrix $\mathbf{V}^\infty$ is the covariance matrix of $\{\phi_i\}_{i=1:n}$ taken over $\mathbb{R}^d$ with the measure $N(0, \alpha^2\mathbf{I})$. Hence showing that $\mathbf{V}^\infty$ is strictly positive definite is equivalent to showing that $\{\phi_i\}_{i=1,\ldots n}$ are linearly independent. Unlike (Du et al., 2019b), the

functionals under consideration are not piecewise constant so a different construction is used to prove independence. Analogously, a new set of operators, $\theta_i(\mathbf{v}) := \sigma(\mathbf{x}_i^{\mathbf{v}})$, is constructed for $\mathbf{G}^\infty$. Interestingly, each ϕ_i corresponds to $\frac{d\theta_i}{d\mathbf{v}}$. The full proof is presented in Appendix D. As already observed from evolution (3.5), different magnitudes of α can lead to two distinct regimes that are discussed below. We present the main results for each regime.

V-dominated convergence

For $\alpha < 1$ convergence is dominated by $\mathbf{V}(t)$ and $\lambda_{\min}(\mathbf{\Lambda}(t)) \geq \frac{1}{\alpha^2}\lambda_{\min}(\mathbf{V}(t))$. We present the convergence theorem for the $\mathbf{V}$-dominated regime here.

Theorem 4.1 ($\mathbf{V}$-dominated convergence)**.** *Suppose a neural network of the form (1.2) is initialized as in (2.4) with $\alpha \leq 1$ and that Assumptions 1,2 hold. In addition, suppose the neural network is trained via the regression loss (2.5) with targets $\mathbf{y}$ satisfying $\|\mathbf{y}\|_\infty = O(1)$. If $m = \Omega\big(n^4 \log(n/\delta)/\lambda_0^4\big)$, then with probability $1-\delta$,*

1. *For iterations $s = 0, 1, \ldots$, the evolution matrix $\mathbf{\Lambda}(s)$ satisfies $\lambda_{\min}(\mathbf{\Lambda}(s)) \geq \frac{\lambda_0}{2\alpha^2}$.*
2. *WN training with gradient descent of step-size $\eta = O\Big(\frac{\alpha^2}{\|\mathbf{V}^\infty\|_2}\Big)$ converges linearly as*

$$\|\mathbf{f}(s) - \mathbf{y}\|_2^2 \leq \Big(1 - \frac{\eta\lambda_0}{2\alpha^2}\Big)^s \|\mathbf{f}(0) - \mathbf{y}\|_2^2.$$

The proof of Theorem 4.1 is presented in Appendix C. We will provide a sketch below. We make the following observations about our $\mathbf{V}$-dominated convergence result.

The required over-parametrization m is independent of α. Further, the dependence of m on the failure probability is $\log(1/\delta)$. This improves previous results that require polynomial dependence of order δ^3. Additionally, we reduce the dependence on the sample size from n^6 (as appears in (Arora et al., 2019a)) to $n^4\log(n)$.

In Theorem 4.1, smaller α leads to faster convergence, since the convergence is dictated by λ_0/α^2. Nonetheless, smaller α is also at the cost of smaller allowed step-sizes, since $\eta = O(\alpha^2/\|\mathbf{V}^\infty\|_2)$. The trade-off between step-size and convergence speed is typical. For example, this is implied in Chizat et al. (Chizat et al., 2019), where nonetheless the authors point out that for gradient flow training, the increased convergence rate is not balanced by a limitation on the step-size. The works (Arora et al., 2019b; Hoffer et al., 2018; Wu et al., 2018) define an effective step-size (adaptive step-size) $\eta' = \eta/\alpha^2$ to avoid the dependence of η on α.

G-dominated convergence

For $\alpha > 1$ our convergence result for the class (1.2) is based on monitoring the least eigenvalue of $\mathbf{G}(t)$. Unlike $\mathbf{V}$-dominated convergence, α does not affect the convergence speed in this regime.

Theorem 4.2 ($\mathbf{G}$-dominated convergence)**.** *Suppose a network of the form (1.2) is initialized as in (2.4) with $\alpha \geq 1$ and that Assumptions 1, 2 hold. In addition, suppose the neural network is trained via the regression loss (2.5) with targets $\mathbf{y}$ satisfying $\|\mathbf{y}\|_\infty = O(1)$. If $m = \Omega\big(\max\{n^4\log(n/\delta)/\alpha^4\mu_0^4, n^2\log(n/\delta)/\mu_0^2\}\big)$, then with probability $1-\delta$,*

1. *For iterations $s = 0, 1, \ldots$, the evolution matrix $\mathbf{\Lambda}(s)$ satisfies $\lambda_{\min}(\mathbf{\Lambda}(s)) \geq \frac{\mu_0}{2}$.*
2. *WN training with gradient descent of step-size $\eta = O\Big(\frac{1}{\|\mathbf{\Lambda}(t)\|}\Big)$ converges linearly as*

$$\|\mathbf{f}(s) - \mathbf{y}\|_2^2 \leq \Big(1 - \frac{\eta\mu_0}{2}\Big)^s \|\mathbf{f}(0) - \mathbf{y}\|_2^2.$$

We make the following observations about our $\mathbf{G}$-dominated convergence result, and provide a proof sketch further below.

Theorem 4.2 holds for $\alpha \geq 1$ so long as $m = \Omega\big(\max\{n^4\log(n/\delta)/\mu_0^4\alpha^4, n^2\log(n/\delta)/\mu_0^2\}\big)$. Taking $\alpha = \sqrt{n/\mu_0}$ gives an optimal required over-parametrization of order $m = \Omega\big(n^2\log(n/\delta)/\mu_0^2\big)$. This significantly improves on previous results (Du et al., 2019b) for unnormalized training that have dependencies of order 4 in the least eigenvalue, cubic dependence in $1/\delta$, and n^6 dependence in the number of samples n. In contrast to $\mathbf{V}$-dominated convergence, here the rate of convergence μ_0 is independent of α but the over-parametrization m is α-dependent. We elaborate on this curious behavior in the next sections.

Proof sketch of main results The proof of Theorems 4.1 and 4.2 is inspired by a series of works including (Arora et al., 2019a; Du et al., 2019a;b; Wu et al., 2019; Zhang et al., 2019). The proof has the following steps: (**I**) We show that at initialization $\mathbf{V}(0), \mathbf{G}(0)$ can be viewed as empirical estimates of averaged data-dependent kernels $\mathbf{V}^\infty, \mathbf{G}^\infty$ that are strictly positive definite under Assumption 1. (**II**) For each regime, we prove that the corresponding kernel remains positive definite if $\mathbf{v}_k(t)$ and $g_k(t)$ remain near initialization for each $1 \leq k \leq m$. (**III**) Given a uniformly positive definite evolution matrix $\mathbf{\Lambda}(t)$ and sufficient over-parametrization we show that each neuron, $\mathbf{v}_k(t), g_k(t)$ remains close to its initialization. The full proof is presented in Appendix B for gradient flow and Appendix C for finite-step gradient descent. Next we interpret the main results and discuss how

the modified NTK in WN can be viewed as a form of natural gradient.

Connection with natural gradient Natural gradient methods define the steepest descent direction in the parameter space of a model from the perspective of function space. This amounts to introducing a particular geometry into the parameter space which is reflective of the geometry of the corresponding functions. A re-parametrization of a model, and WN in particular, can also be interpreted as choosing a particular geometry for the parameter space. This gives us a perspective from which to study the effects of WN. The recent work of (Zhang et al., 2019) studies the effects of natural gradient methods from the lens of the NTK and shows that when optimizing with the natural gradient, one is able to get significantly improved training speed. In particular, using the popular natural gradient method K-FAC improves the convergence speed considerably.

Natural gradients transform the NTK from $\mathbf{J}\mathbf{J}^\top$ to $\mathbf{J}\mathbf{G}^\dagger\mathbf{J}^\top$, where $\mathbf{J}$ is the Jacobian with respect to the parameters and $\mathbf{G}$ is the metric. The WN re-parametrization transforms the NTK from $\mathbf{J}\mathbf{J}^\top$ to $\mathbf{J}\mathbf{S}^\top\mathbf{S}\mathbf{J}^\top$. To be more precise, denote the un-normalized NTK as $\mathbf{H} = \mathbf{J}\mathbf{J}^\top$, where $\mathbf{J}$ is the Jacobian matrix for $\mathbf{x}_1, \dots \mathbf{x}_n$ written in a compact tensor as $\mathbf{J} = \big[\mathbf{J}_1, \dots \mathbf{J}_n\big]^\top$ with $\mathbf{J}_i = \left[\frac{\partial f(\mathbf{x}_i)}{\partial \mathbf{w}_1} \dots \frac{\partial f(\mathbf{x}_i)}{\partial \mathbf{w}_m}\right]$, where matrix multiplication is a slight abuse of notation. Namely $\mathbf{J} \in \mathbb{R}^{n\times m\times d}$ and we define multiplication of $\mathbf{A} \in \mathbb{R}^{n\times m\times d} \times \mathbf{B} \in \mathbb{R}^{d\times m\times p} \to \mathbf{A}\mathbf{B} \in \mathbb{R}^{n\times p}$ as

$$(\mathbf{A}\mathbf{B})_{ij} = \sum_{k=1}^{m} \langle \mathbf{A}_{ik:}, \mathbf{B}_{:kj} \rangle.$$

For any re-parametrization $\mathbf{w}(\mathbf{r})$, we have that

$$\mathbf{\Lambda} = \mathbf{K}\mathbf{K}^\top,$$

where $\mathbf{K} = \mathbf{J}\mathbf{S}^\top$ and $\mathbf{S}$ corresponds to the Jacobian of the re-parametrization $\mathbf{w}(\mathbf{r})$. By introducing WN layers the reparameterized NTK is compactly written as

$$\mathbf{\Lambda} = \mathbf{J}\mathbf{S}^\top\mathbf{S}\mathbf{J}^\top.$$

Here $\mathbf{S} = [\mathbf{S}_1, \dots, t\mathbf{S}_m]$ with

$$\mathbf{S}_k = \left[\frac{g_k}{\|\mathbf{v}_k\|_2}\left(\mathbf{I} - \frac{\mathbf{v}_k\mathbf{v}_k^\top}{\|\mathbf{v}_k\|_2}\right), \frac{\mathbf{v}_k}{\|\mathbf{v}_k\|_2}\right].$$

The term $\mathbf{N}(\alpha) := \mathbf{S}\mathbf{S}^\top$ leads to a family of different gradient re-parametrizations depending on α. The above representation of the WN NTK is equivalent to $\mathbf{\Lambda}(\alpha) = \frac{1}{\alpha^2}\mathbf{V} + \mathbf{G} = \mathbf{J}\mathbf{N}(\alpha)\mathbf{J}^\top$. For different initialization magnitudes α, $\mathbf{N}(\alpha)$ leads to different NTKs with modified properties.

For $\alpha = 1$ the term corresponds to training without normalization, yet over $\alpha \in (0, \infty)$, $\mathbf{N}(\alpha)$ leads to a family NTKs with different properties. In addition there exists an α^* that maximizes the convergence rate. Such α^* is either a proper global maximum or is attained at one of $\alpha \to 0, \alpha \to \infty$. For the latter, one may fix α^* with $\alpha^* \ll 1$ or $\alpha^* \gg 1$ respectively so that there exists α^* that outpaces un-normalized convergence ($\alpha = 1$). This leads to equal or faster convergence of WN as compared with un-normalized training:

Proposition 2 (Fast Convergence of WN)**.** *Suppose a neural network of the form* (1.2) *is initialized as in* (2.4) *and that Assumptions 1,2 hold. In addition, suppose the network is trained via the regression loss* (2.5) *with targets* $\mathbf{y}$ *satisfying* $\|\mathbf{y}\|_\infty = O(1)$*. Then, with probability* $1 - \delta$ *over the initialization, there exists* α^* *such that WN training with* α^* *initialization leads to faster convergence: If* $m = \Omega\big(n^4 \log(n/\delta)/\min\{\lambda_0^4, \mu_0^4\}\big)$*,*

1. *WN training with gradient descent of step-size* $\eta_{\alpha^*} = O\Big(\frac{1}{\|\mathbf{V}^\infty/(\alpha^*)^2 + \mathbf{G}^\infty\|_2}\Big)$ *converges linearly as*

$$\|\mathbf{f}(s) - \mathbf{y}\|_2^2 \leq \left(1 - \eta_{\alpha^*}\big(\lambda_0/2(\alpha^*)^2 + \mu_0/2\big)\right)^s \|\mathbf{f}(0) - \mathbf{y}\|_2^2.$$

2. *The convergence rate of WN is faster than un-normalized convergence,*

$$\big(1 - \eta_{\alpha^*}\lambda_{\min}(\mathbf{\Lambda}(s))\big) \leq \big(1 - \eta\lambda_{\min}(\mathbf{H}(s))\big).$$

This illustrates the utility of WN from the perspective of the NTK, guaranteeing that there exists an α^* that leads to faster convergence in *finite-step* gradient descent as compared with un-normalized training.

5. Related Work

Normalization methods theory A number of recent works attempt to explain the dynamics and utility of various normalization methods in deep learning. The original works on BN (Ioffe and Szegedy, 2015) and WN (Salimans and Kingma, 2016) suggest that normalization procedures improve training by fixing the intermediate layers' output distributions. The works of Bjorck et al. (2018) and Santurkar et al. (2018) argue that BN may improve optimization by improving smoothness of the Hessian of the loss, therefore allowing for larger step-sizes with reduced instability. Hoffer et al. (2017) showed that the effective step-size in BN is divided by the magnitude of the weights. This followed the work on WNgrad (Wu et al., 2018) that introduces an adaptive step-size algorithm based on this fact. Following the intuition of WNGrad, Arora et al. (2019b) proved

that for smooth loss and network functions, the diminishing "effective step-size" of normalization methods leads to convergence with optimal convergence rate for properly initialized step-sizes. The work of Kohler et al. (2019) explains the accelerated convergence of BN from a "length-direction decoupling" perspective. The authors along with Cai et al. (2019) analyze the linear least squares regime, with Kohler et al. (2019) presenting a bisection method for finding the optimal weights. Robustness and regularization of Batch Normalization is investigated by Luo et al. (2018) and improved generalization is analyzed empirically. Shortly after the original work of WN, (Yoshida et al., 2017) showed that for a single precptron WN may speed-up training and emphasized the importance of the norm of the initial weights. Additional stability properties were studied by Yang et al. (2019) via mean-field analysis. The authors show that gradient instability is inevitable even with BN as the number of layers increases; this is in agreement with Balduzzi et al. (2017) for networks with residual connections. The work of Arpit et al. (2019) suggests initialization strategies for WN and derives lower bounds on the width to guarantee same order gradients across the layers.

Over-parametrized neural networks There has been a significant amount of recent literature studying the convergence of un-normalized over-parametrized neural networks. In the majority of these works the analysis relies on the width of the layers. These include 2-layer networks trained with Gaussian inputs and outputs from a teacher network (Li and Yuan, 2017; Tian, 2017) and (Du et al., 2018) (with WN). Assumptions on the data distribution are relaxed in (Du et al., 2019b) and the works that followed (Arora et al., 2019a; Wu et al., 2019; Zhang et al., 2019). Our work is inspired by the mechanism presented in this chain of works. Wu et al. (2019) extend convergence results to adaptive step-size methods and propose AdaLoss. Recently, the global convergence of over-parameterized neural networks was also extended to deep architectures (Allen-Zhu et al., 2019b; Du et al., 2019a; Zou and Gu, 2019; Zou et al., 2020). In the context of the NTK, Zhang et al. (2019) have proved fast convergence of neural networks trained with natural gradient methods and the K-FAC approximation (Martens and Grosse, 2015). In the over-parameterized regimes, Arora et al. (2019a) develop generalization properties for the networks of the form (1.1). In addition, in the context of generalization, Allen-Zhu et al. (2019a) illustrates good generalization for deep neural networks trained with gradient descent. Cao and Gu (2020) and (Cao and Gu, 2019) derive generalization error bounds of gradient descent and stochastic gradient descent for learning over-parametrization deep ReLU neural networks.

6. Discussion

Dynamic normalization is the most common optimization set-up of current deep learning models, yet understanding the convergence of such optimization methods is still an open problem. In this work we present a proof giving sufficient conditions for convergence of dynamically normalized 2-layer ReLU networks trained with gradient descent. To the best of our knowledge this is the first proof showcasing convergence of gradient descent training of neural networks with dynamic normalization and general data, where the objective function is non-smooth and non-convex. To understand the canonical behavior of each normalization layer, we study the shallow neural network case, that enables us to focus on a single layer and illustrate the dynamics of weight normalization. Nonetheless, we believe that using the techniques presented in (Allen-Zhu et al., 2019b; Du et al., 2019a) can extend the proofs to the deep network settings. Through our analysis notion of "length-direction decoupling" is clarified by the neural tangent kernel $\mathbf{\Lambda}(t)$ that naturally separates in our analysis into "length", $\mathbf{G}(t)$, and "direction", $\mathbf{V}(t)/\alpha^2$, components. For $\alpha = 1$ the decomposition initially matches un-normalized training. Yet we discover that in general, normalized training with gradient descent leads to 2 regimes dominated by different pieces of the neural tangent kernel. Our improved analysis is able to reduce the amount of over-parametrization that was needed in previous convergence works in the un-normalized setting and in the $\mathbf{G}$-dominated regime, we prove convergence with a significantly lower amount of over-parametrization as compared with un-normalized training.

Acknowledgement YD has been supported by the National Science Foundation under Graduate Research Fellowship Grant No. DGE-1650604. QG was supported in part by the National Science Foundation CAREER Award IIS-1906169, BIGDATA IIS-1855099, and Salesforce Deep Learning Research Award. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement n^o 757983).

References

Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in Neural Information Processing Systems 32*, pages 6158–6169. 2019a.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 242–252. PMLR, 2019b.

Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ru-

osong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 322–332. PMLR, 2019a.

Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. In *International Conference on Learning Representations*, 2019b. URL https://openreview.net/forum?id=rkxQ-nA9FX.

Devansh Arpit, Víctor Campos, and Yoshua Bengio. How to initialize your network? robust initialization for weightnorm & resnets. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 10902–10911. Curran Associates, Inc., 2019.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *Deep Learning Symposium, NIPS-2016*, 2016.

David Balduzzi, Marcus Frean, Lennox Leary, JP Lewis, Kurt Wan-Duo Ma, and Brian McWilliams. The shattered gradients problem: If resnets are the answer, then what is the question? In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 342–350. JMLR. org, 2017.

Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. In *Advances in Neural Information Processing Systems 31*, pages 7694–7705. 2018.

Yongqiang Cai, Qianxiao Li, and Zuowei Shen. A quantitative analysis of the effect of batch normalization on gradient descent. In *International Conference on Machine Learning*, pages 882–890, 2019.

Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in Neural Information Processing Systems 32*, pages 10836–10846. 2019.

Yuan Cao and Quanquan Gu. Generalization error bounds of gradient descent for learning over-parameterized deep ReLU networks. In *AAAI*, 2020.

Lénaïc Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems 32*, pages 2937–2947. 2019.

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1675–1685, Long Beach, California, USA, 09–15 Jun 2019a. PMLR.

Simon S. Du, Jason D. Lee, and Yuandong Tian. When is a convolutional filter easy to learn? In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SkA-IE06W.

Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019b. URL https://openreview.net/forum?id=S1eK3i09YQ.

Igor Gitman and Boris Ginsburg. Comparison of batch normalization and weight normalization algorithms for the large-scale image classification. *arXiv preprint arXiv:1709.08145*, 2017.

Elad Hoffer, Itay Hubara, and Daniel Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In *Advances in Neural Information Processing Systems 30*, pages 1731–1741. 2017.

Elad Hoffer, Ron Banner, Itay Golan, and Daniel Soudry. Norm matters: efficient and accurate normalization schemes in deep networks. In *Advances in Neural Information Processing Systems 31*, pages 2160–2170. 2018.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456. PMLR, 2015.

Arthur Jacot, Franck Gabriel, and Clement Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems 31*, pages 8571–8580. 2018.

Jonas Kohler, Hadi Daneshmand, Aurelien Lucchi, Thomas Hofmann, Ming Zhou, and Klaus Neymeyr. Exponential convergence rates for batch normalization: The power of length-direction decoupling in non-convex optimization. In *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 806–815. PMLR, 2019.

Yuanzhi Li and Yingyu Liang. Learning overparameterized neural networks via stochastic gradient descent on structured data. In *Advances in Neural Information Processing Systems 31*, pages 8157–8166. 2018.

Yuanzhi Li and Yang Yuan. Convergence analysis of two-layer neural networks with ReLU activation. In *Advances in Neural Information Processing Systems 30*, pages 597–607. 2017.

Ping Luo, Xinjiang Wang, Wenqi Shao, and Zhanglin Peng. Understanding regularization in batch normalization. *arXiv preprint arXiv:1809.00846*, 2018.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2408–2417. PMLR, 2015.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

Samet Oymak and Mahdi Soltanolkotabi. Towards moderate overparameterization: global convergence guarantees for training shallow neural networks. *arXiv preprint arXiv:1902.04674*, 2019.

Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in Neural Information Processing Systems 29*, pages 901–909. 2016.

Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. How does batch normalization help optimization? In *Advances in Neural Information Processing Systems 31*, pages 2483–2493. 2018.

Yuandong Tian. An analytical formula of population gradient for two-layered ReLU network and its applications in convergence and critical point analysis. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3404–3413. JMLR. org, 2017.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge University Press, 2018.

Xiaoxia Wu, Rachel Ward, and Léon Bottou. WNGrad: Learn the learning rate in gradient descent. *arXiv preprint arXiv:1803.02865*, 2018.

Xiaoxia Wu, Simon S Du, and Rachel Ward. Global convergence of adaptive gradient methods for an over-parameterized neural network. *arXiv preprint arXiv:1902.07111*, 2019.

Greg Yang, Jeffrey Pennington, Vinay Rao, Jascha Sohl-Dickstein, and Samuel S. Schoenholz. A mean field theory of batch normalization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=SyMDXnCcF7.

Yuki Yoshida, Ryo Karakida, Masato Okada, and Shun-ichi Amari. Statistical mechanical analysis of online learning with weight normalization in single layer perceptron. *Journal of the Physical Society of Japan*, 86(4):044002, 2017.

Guodong Zhang, James Martens, and Roger B Grosse. Fast convergence of natural gradient descent for over-parameterized neural networks. In *Advances in Neural Information Processing Systems 32*, pages 8082–8093. 2019.

Difan Zou and Quanquan Gu. An improved analysis of training over-parameterized deep neural networks. In *Advances in Neural Information Processing Systems 32*, pages 2055–2064. 2019.

Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Gradient descent optimizes over-parameterized deep ReLU networks. *Machine Learning*, 109(3):467–492, 2020. doi: 10.1007/s10994-019-05839-6. URL https://doi.org/10.1007/s10994-019-05839-6.