Towards Understanding Learning in Neural Networks with Linear Teachers

Roei Sarussi [1] **Alon Brutzkus** [1] **Amir Globerson** [1]

Abstract

Can a neural network minimizing cross-entropy learn linearly separable data? Despite progress in the theory of deep learning, this question remains unsolved. Here we prove that SGD globally optimizes this learning problem for a two-layer network with Leaky ReLU activations. The learned network can in principle be very complex. However, empirical evidence suggests that it often turns out to be approximately linear. We provide theoretical support for this phenomenon by proving that if network weights converge to two weight clusters, this will imply an approximately linear decision boundary. Finally, we show a condition on the optimization that leads to weight clustering. We provide empirical results that validate our theoretical analysis.

1. Introduction

Neural networks have achieved remarkable performance in many machine learning tasks (Krizhevsky et al., 2012; Silver et al., 2016; Devlin et al., 2019). Although their success has already transformed technology, a theoretical understanding of how this performance is achieved is not complete. Here we focus on one of the simplest learning settings that is still not understood. We consider linearly separable data (i.e., generated by a "linear teacher") that is being learned by a two layer neural net with leaky ReLU activations and minimization of cross entropy loss using gradient descent or its variants. Two key questions immediately come up in this context:

- **The Optimization Question**: Will the optimization succeed in finding a classifier with zero training error, and arbitrarily low training loss?
- **The Inductive Bias Question**: With a large number of hidden units, the network can find many solutions that will separate the data. Which of these will be found by gradient descent?

Our work addresses these questions as follows.

The Optimization Question: We prove that stochastic gradient descent (SGD) will converge to arbitrary low training loss. Concretely, we show that for any $\epsilon > 0$, SGD will converge to ϵ cross-entropy loss in $O\left(\frac{1}{\epsilon^2}\right)$ iterations. We consider SGD which performs multiple passes over the data and we devise a novel variant of the perceptron proof to analyze this setting. Our analysis bounds the number of epochs that have high loss examples, and uses this to show convergence to a low loss solution. Importantly, our result holds for any network size and scale of initialization. Therefore, our analysis goes beyond the Neural Tangent Kernel (NTK) analyses which require large network sizes and relatively large initialization scales.

The Inductive Bias Question: We empirically observe that when a small initialization scale is used, the learned network converges to a decision boundary that is very close to linear. See Figure 1b for a 2D example.[1] We also observe that all neurons cluster nicely into two sets of vectors (i.e., they form two groups of well-aligned neurons) as in Figure 1d. To support these empirical findings, we provide the following theoretical results:

(1) We prove that an approximate clustering of the neurons implies that the decision boundary of the network is approximately linear. This is a result of a nice property of leaky ReLU networks which we prove in Section 5.

(2) We provide a novel sufficient condition on the optimization path of gradient flow which implies convergence to clustered solutions. With the result above, it implies convergence to a linear decision boundary. The condition states that from a certain iteration on, all neurons with the same output sign "agree" on the classification of the data. We observe that this condition holds empirically for several synthetic and real datasets. Finally, we use the latter result to prove that under certain assumptions, the learned network is a solution to an SVM problem with a specific kernel.

Our results above make significant headway in understanding why optimization is tractable with linear teachers, and why convergence is to approximately linear boundaries. We

[1]The Blavatnik School of Computer Science, Tel Aviv University. Correspondence to: Alon Brutzkus <alonbrutzkus@mail.tau.ac.il>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

[1]See Section 5.1 and Section 6.3 for more empirical examples.

also provide empirical evaluation that confirms that weight clustering indeed explains why approximate linear decision boundaries are learned.

2. Related Work

Since training neural networks is NP-Hard for worst-case datasets (Blum & Rivest, 1992), recent works have analyzed neural networks under certain data assumptions to better understand their performance in practice. One common assumption is to analyze neural networks when the data is linearly separable. Even in this case, the theoretical analysis of optimization and generalization is far from resolved. In a work closely related to ours, Brutzkus et al. (2018) consider this setting and show that SGD converges to zero loss for linearly separable data (which was later extended to ReLU activations using noisy SGD in Wang et al. (2019)). The key difference from our work is that they use the hinge loss instead of the cross entropy loss. The cross entropy loss creates unique challenges for proving convergence as we show in Section 4. Thus, their results cannot be directly applied for the cross entropy loss and we use novel techniques to guarantee convergence of SGD in this case. The second key difference from their work is that we present novel insights on the inductive bias of SGD using results of Lyu & Li (2020) and Ji & Telgarsky (2020) which hold for the cross entropy loss and not for the hinge loss. Finally, our result which shows that a network with clustered neurons has an approximate linear decision boundary is new and holds irrespective of the loss used.

Recently, Phuong & Lampert (2021) analyzed a subclass of linear teachers where data is "orthogonally separable". In this case they show that training a ReLU network with the cross entropy loss results in a solution where weights are aligned. In terms of our results, this can be viewed as a case of convergence to a particular PAR (see Section 6). Several other works assume that the data is linearly separable but also that the networks are *linear* (Ji & Telgarsky, 2019a; Moroshko et al., 2020; Gunasekar et al., 2018). We study the more challenging and realistic setting of two layer nonlinear networks with Leaky ReLU activations.

Several works (Lyu & Li, 2020; Ji & Telgarsky, 2020; Nacson et al., 2019) studied the inductive bias of two-layer homogeneous networks and showed connections between gradient methods and margin maximization. Their results hold under the assumption that gradient methods achieve a certain loss value. However, we provide a convergence proof for SGD that shows that it can obtain arbitrary low loss values. Furthermore, we use the results of Lyu & Li (2020) and Ji & Telgarsky (2020) to obtain a more fine grained analysis of the inductive bias of gradient flow for linear teachers. Other works considered the inductive bias of infinite two-layer networks (Chizat & Bach, 2020; 2018; Wei et al., 2019; Mei et al., 2018). Our results hold for networks of any size. An inductive bias towards clustered solutions has been observed in Brutzkus & Globerson (2019) and proved for a simple setup with nonlinear data.

Fully connected networks were also analyzed via the NTK approximation (Du et al., 2019; 2018; Arora et al., 2019; Ji & Telgarsky, 2019b; Cao & Gu, 2019; Jacot et al., 2018; Fiat et al., 2019; Allen-Zhu et al., 2019; Li & Liang, 2018; Daniely et al., 2016). However other works (Yehudai & Shamir, 2019; Daniely & Malach, 2020) have highlighted limitations of the NTK framework, suggesting that it does not accurately model neural networks as they are used in practice. Our convergence analysis in Section 4 holds for any initialization scale and network size and therefore goes beyond the NTK analysis.

Recently, Li et al. (2020) analyzed two-layer networks beyond NTK in the case of Gaussian inputs and squared loss. We assume linearly separable inputs and the cross entropy loss. Allen-Zhu & Li (2019) analyze a three layer ResNet and provide generalization guarantees for sufficiently wide networks in a regression setting. Woodworth et al. (2020) study the inductive bias of gradient methods for a simplified nonlinear model.

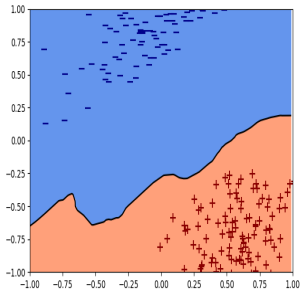
3. Preliminaries

Notations: We use $||\cdot||$ to denote the L^2 norm on vectors and Frobenius norm on matrices. For a vector $\boldsymbol{v}$ we denote $\hat{\boldsymbol{v}} = \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|}$.

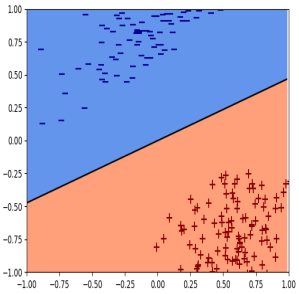
Data Generating Distribution: Define $\mathbb{X} = \{\boldsymbol{x} \in \mathbb{R}^d : ||\boldsymbol{x}|| \leq R_x\}$ and $\mathbb{Y} = \{\pm 1\}$. We consider a distribution of linearly separable points. Formally, let $D(\mathrm{x}, \mathrm{y})$ be a distribution over $\mathbb{X} \times \mathbb{Y}$ such that there exists $\boldsymbol{w}^* \in \mathbb{R}^d$ for which $\mathbb{P}_{(\mathrm{x},\mathrm{y})\sim D}[\mathrm{y}\boldsymbol{w}^* \cdot \boldsymbol{x} \geq 1] = 1$. Let $\mathbb{S} := \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_n, y_n)\} \subseteq \mathbb{X} \times \mathbb{Y}$ be a training set sampled IID from $D(\mathrm{x}, \mathrm{y})$. Let $\mathbb{S}_+ \subset \mathbb{S}$ denote the training points with positive labels and $\mathbb{S}_-$ with negative labels. We denote $\boldsymbol{x} \in \mathbb{S}$ if there exists $y \in \{\pm 1\}$ such that $(\boldsymbol{x}, y) \in \mathbb{S}$.

Network Architecture: We consider a two-layer neural network with $2k > 0$ hidden units, where the second layer is fixed and the first layer is learned. Formally, we denote the parameters of the network that are learned by $\boldsymbol{W} \in \mathbb{R}^{2k \times d}$ and for the second layer we define the *fixed* vector $\boldsymbol{v} \in \mathbb{R}^{2k}$, where $\boldsymbol{v} = (\overbrace{v, v, ..., v}^{k}, \overbrace{-v, ..., -v}^{k})$ and $v > 0$. The network output is given by the function $N_{\boldsymbol{W}} : \mathbb{R}^d \to \mathbb{R}$ defined as $N_{\boldsymbol{W}}(\boldsymbol{x}) = \boldsymbol{v} \cdot \sigma(\boldsymbol{W}\boldsymbol{x})$, where $\sigma(\boldsymbol{x}) = \max\{\boldsymbol{x}, \alpha\boldsymbol{x}\}$ is the Leaky-ReLU activation function applied element-wise, parameterized by $0 < \alpha < 1$ and $\cdot$ denotes dot product.[2]

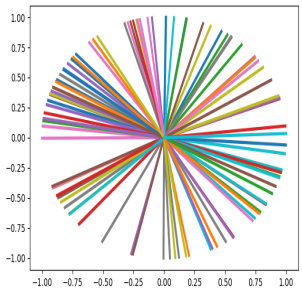
[2]We do not introduce a bias term, but all our results extend to using bias (see Supplementary for a formal justification).



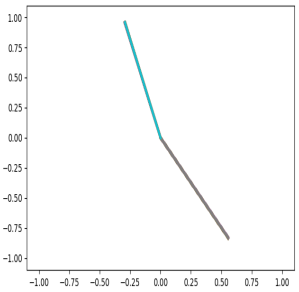
(a) Decision boundary (large initialization)



(b) Decision boundary (small initialization)



(c) Learned Neurons (large initialization)



(d) Learned Neurons (small initialization)

Figure 1: Results for training a Leaky-ReLU network on linearly separable data, with different initialization scales. Figures (a)+(b) show the resulting decision boundary. It can be seen that large initialization leads to a non-linear boundary, whereas small initialization leads to a linear boundary. Figures (c)+(d) show the learned weight vectors (normalized to unit norm). It can be seen that small initialization leads to two tight clusters of neurons whereas large initialization does not lead to clustering. The network has 100 neurons, initialized from a Gaussian with standard deviation 0.001 for small initialization and 30 for large initialization.

It is easy to see that such a network is as expressive as a standard two-layer network where the second layer vector is not fixed (Brutzkus et al., 2018). Furthermore, the assumption that the second layer is fixed is common in previous works (e.g., see Du et al., 2018; Brutzkus et al., 2018; Ji & Telgarsky, 2019b). We denote row i of $\boldsymbol{W}$ by $\boldsymbol{w}^{(i)}$ and row $k+i$ by $\boldsymbol{u}^{(i)}$ for $1 \leq i \leq k$. We say that $\boldsymbol{w}^{(i)}$ are the $\boldsymbol{w}$ neurons and $\boldsymbol{u}^{(i)}$ are the $\boldsymbol{u}$ neurons. Then the network is given by:

$$N_{\boldsymbol{W}}(\boldsymbol{x}) = v \sum_{i=1}^{k} \sigma\left(\boldsymbol{w}^{(i)} \cdot \boldsymbol{x}\right) - v \sum_{i=1}^{k} \sigma\left(\boldsymbol{u}^{(i)} \cdot \boldsymbol{x}\right) \quad (1)$$

Training Loss: Define the empirical loss over $\mathbb{S}$ to be the cross-entropy loss:

$$L_{\mathbb{S}}(\boldsymbol{W}) \coloneqq \frac{1}{n} \sum_{i=1}^{n} \ell(y_i N_{\boldsymbol{W}}(\boldsymbol{x}_i)) \quad (2)$$

where $\ell(z) = \log(1 + e^{-z})$ is the binary cross entropy loss.

Optimization Algorithm: The training-loss mimimization optimization problem is to find:

$$\arg \min_{\boldsymbol{W} \in \mathbb{R}^{2k \times d}} L_{\mathbb{S}}(\boldsymbol{W}) \quad (3)$$

We focus on two different gradient-based methods in different parts of the paper. First, we consider the case where $L_{\mathbb{S}}(\boldsymbol{W})$ is minimized using SGD in epochs with a batch size of one and a learning rate η. Data points are sampled without replacement at each epoch. Denote by $\boldsymbol{W}_t$ the parameters after t updates.

Our main optimization result, described in Section 4 is shown for SGD. When studying convergence to clustered solutions, we consider gradient flow, because there we can use recent strong results from Lyu & Li (2020) and Ji & Telgarsky (2020). Recall that gradient flow is the infinitesimal step limit of gradient descent where $\boldsymbol{W}_t$ changes continuously in time and satisfies the differential inclusion $\frac{d\boldsymbol{W}_t}{dt} \in -\partial^\circ L_{\mathbb{S}}(\boldsymbol{W}_t)$. Here $\partial^\circ L_{\mathbb{S}}(\boldsymbol{W}_t)$ stands for Clarke's sub-differential which is a generalization of the differential for non-differentiable functions.

The importance of gradient flow is that it can be shown to maximize margin in the following sense. Define the network margin for a single data point $(\boldsymbol{x}_i, y_i)$ by $q_i(\boldsymbol{W}) \coloneqq y_i N_{\boldsymbol{W}}(\boldsymbol{x}_i)$, and the normalized network margin as:

$$\overline{\gamma}(\boldsymbol{W}) \coloneqq \frac{1}{||\boldsymbol{W}||} \min_{(\boldsymbol{x},y) \in \mathbb{S}} y_i N_{\boldsymbol{W}}(\boldsymbol{x}) \quad (4)$$

where $||\boldsymbol{W}||$ is the Frobenius norm of $\boldsymbol{W}$.

The smoothed margin is defined as:[3]

$$\tilde{\gamma}(\boldsymbol{W}) \coloneqq \frac{1}{||\boldsymbol{W}||}\log\left(\frac{1}{\exp(nL_{\mathbb{S}}(\boldsymbol{W}))-1}\right) \quad (5)$$

From Lyu & Li (2020) and Ji & Telgarsky (2020) it follows that gradient flow converges to KKT points of the network margin maximization problem (see Supplementary for details). Here we will use this result in Section 6 to characterize the linear decision boundaries of learned networks.

4. Risk Convergence

We next prove that for any $\varepsilon > 0$, SGD converges to ε empirical-loss (see Eq. (2)) within $O\left(\frac{n^4}{\varepsilon^2}\right)$ updates.

Let $\overrightarrow{\boldsymbol{W}}_t = \left(\boldsymbol{w}_t^{(1)},\ldots,\boldsymbol{w}_t^{(k)},\boldsymbol{u}_t^{(1)},\ldots,\boldsymbol{u}_t^{(k)}\right) \in \mathbb{R}^{2kd}$ be the vectorized version of $\boldsymbol{W}_t$. We assume that the network is initialized such that the norms of all rows of $\boldsymbol{W}_0$ are upper bounded by some constant $R_0 > 0$. Namely for all $1 \le i \le k$ it holds that $||\boldsymbol{w}_0^{(i)}||, ||\boldsymbol{u}_0^{(i)}|| \le R_0$.

Define $M(n,\epsilon) = \frac{Cn^4}{\varepsilon^2}$, where C is a constant that depends polynomially on $R_x, \alpha, R_0, k, \eta, v$ and $||\boldsymbol{w}^*||$.[4] See the supplementary for the exact definition of $M(n,\epsilon)$.

The following theorem states that SGD will converge to ϵ loss within $M(n,\varepsilon)$ updates.

Theorem 4.1. *For any $\varepsilon > 0$, there exists an iteration $t \le M(n,\varepsilon)$ such that $L_{\mathbb{S}}(\boldsymbol{W}_t) < \varepsilon$.*

We note that the convergence analysis holds for any $\eta > 0$. This is in line with other analyses of learning linearly separable data, which show that convergence holds for any $\eta > 0$ (Brutzkus et al., 2018). We next briefly sketch the proof of Theorem 4.1. The full proof is deferred to the supplementary.

Our proof is based on the proof for the hinge loss in Brutzkus et al. (2018) with several novel ideas that enable us to show convergence for the cross entropy loss.

For the hinge loss proof, Brutzkus et al. (2018) consider the vector $\overrightarrow{\boldsymbol{W}}^* = (\overbrace{\boldsymbol{w}^*\ldots\boldsymbol{w}^*}^{k}, \overbrace{-\boldsymbol{w}^*\cdots-\boldsymbol{w}^*}^{k}) \in \mathbb{R}^{2kd}$ and define $F(\boldsymbol{W}_t) = \overrightarrow{\boldsymbol{W}}_t \cdot \overrightarrow{\boldsymbol{W}}^*$ and $G(\boldsymbol{W}_t) = \left\|\overrightarrow{\boldsymbol{W}}_t\right\|$. Using an online perceptron proof and the fact that $\frac{|F(\boldsymbol{W}_t)|}{G(\boldsymbol{W}_t)\left\|\overrightarrow{\boldsymbol{W}}^*\right\|} \le 1$, they obtain a bound on the number of points with non-zero loss that SGD samples, which provides the convergence guarantee. This proof is unique to the hinge loss setting, where points can have exactly zero loss. However, in the case of the cross entropy loss, every update has a non-zero loss. Therefore, the online proof for the hinge loss cannot be applied in this case. To overcome this we (1) use an "epoch-based" analysis that is tailored to the SGD variant we use here, that samples data without replacement in each epoch. (2) bound the number of epochs where there exists a point with loss at least ϵ. By applying these key ideas with further technical analyses that are unique to the cross entropy loss, we prove Theorem 4.1.

[3] See Remark A.4. in Lyu & Li (2020).

[4] In some cases the polynomial dependence is on the inverse of the parameter, e.g., $\frac{1}{\eta}$.

5. Weight Clustering and Linear Separation

As shown in Figure 1, learning with SGD can result in a linear decision boundary, despite the existence of zero-loss solutions that are highly non-linear. In what follows, we provide theoeretical and empirical insights into why an approximately linear boundary is learned.

We next show a nice property of Leaky-ReLU networks that can explain why they converge to linear decision boundaries. Assume that a learned network in Eq. (1) is such that all of its $\boldsymbol{w}$ neurons form a ball of "small" radius (i.e., they are well clustered) and likewise all the $\boldsymbol{u}$ neurons (see Figure 1 and Figure 2 for simulations that show such a case). Then, as we show in Theorem 5.1, this implies that the resulting decision boundary will be approximately linear. Later, we give further empirical and theoretical support that learned networks indeed have this clustering structure, and together with Theorem 5.1 this explains the approximate linearity.

Consider the network in Eq. (1). Denote $\overline{\boldsymbol{w}} = \frac{1}{k}\sum_{i=1}^{k}\boldsymbol{w}^{(i)}$ and $\overline{\boldsymbol{u}} = \frac{1}{k}\sum_{i=1}^{k}\boldsymbol{u}^{(i)}$. Also, let r denote the maximum radius of the positive and negative weights around their averages. Namely:

$$\begin{aligned}\|\boldsymbol{w}^{(i)} - \overline{\boldsymbol{w}}\|_2 &\le r \quad i=1,\ldots,k\\ \|\boldsymbol{u}^{(i)} - \overline{\boldsymbol{u}}\|_2 &\le r \quad i=1,\ldots,k\end{aligned}$$

The following result says that the decision boundary will be linear except for a region whose size is determined by r.

Theorem 5.1. *Consider the linear classifier $f(\boldsymbol{x}) = sign\left((\overline{\boldsymbol{w}}-\overline{\boldsymbol{u}})\cdot\boldsymbol{x}\right)$. Then $sign\left(N_{\boldsymbol{W}}(\boldsymbol{x})\right) = f(\boldsymbol{x})$ for all $\boldsymbol{x}$ such that $|(\overline{\boldsymbol{w}}-\overline{\boldsymbol{u}})\cdot\boldsymbol{x}| \ge 2r||\boldsymbol{x}||$.*

The theorem has a simple intuitive implication. The smaller r is, the closer the classifier is to linear. In particular when $r = 0$ the classifier is exactly linear.

An alternative interpretation of the theorem comes from rewriting the condition as:

$$\frac{|(\overline{\boldsymbol{w}}-\overline{\boldsymbol{u}})\cdot\boldsymbol{x}|}{\|\boldsymbol{x}\|\|\overline{\boldsymbol{w}}-\overline{\boldsymbol{u}}\|} \ge \frac{r}{\|\overline{\boldsymbol{w}}-\overline{\boldsymbol{u}}\|} \quad (6)$$

Namely that linearity holds whenever the absolute value of

the cosine of the angle between $\boldsymbol{x}$ and $\overline{\boldsymbol{w}} - \overline{\boldsymbol{u}}$ is greater than $\frac{r}{\|\overline{\boldsymbol{w}} - \overline{\boldsymbol{u}}\|}$.

The proof is in the supplementary and is somewhat technical, but a brief outline is as follows. First we show that $\forall \boldsymbol{x} \in \mathbb{R}^d$ such that $|\overline{\boldsymbol{w}} \cdot \boldsymbol{x}| \geq r||\boldsymbol{x}||$ either we have $[\forall 1 \leq j \leq k \ \boldsymbol{w}^{(j)} \cdot \boldsymbol{x} > 0]$ or it holds that $[\forall 1 \leq j \leq k \ \boldsymbol{w}^{(j)} \cdot \boldsymbol{x} < 0]$ and similarly for the $\boldsymbol{u}$ neurons. Using this, we show that $\forall \boldsymbol{x} \in \mathbb{R}^d$ such that $|\overline{\boldsymbol{w}} \cdot \boldsymbol{x}| \geq r||\boldsymbol{x}|| \wedge |\overline{\boldsymbol{u}} \cdot \boldsymbol{x}| \geq r||\boldsymbol{x}||$ it holds that $\text{sign}\left(N_{\boldsymbol{W}}(\boldsymbol{x})\right) = \text{sign}\left((\overline{\boldsymbol{w}} - \overline{\boldsymbol{u}}) \cdot \boldsymbol{x}\right)$. We show this by dividing the input space to four regions based on the classification of the $\boldsymbol{w}$ and $\boldsymbol{u}$ neurons and using properties of Leaky ReLU. Then via an involved analysis, we proceed to prove that $\text{sign}\left(N_{\boldsymbol{W}}(\boldsymbol{x})\right) = \text{sign}\left((\overline{\boldsymbol{w}} - \overline{\boldsymbol{u}}) \cdot \boldsymbol{x}\right)$ in other regions of the set $\left\{\boldsymbol{x} \in \mathbb{R}^d \mid |(\overline{\boldsymbol{w}} - \overline{\boldsymbol{u}}) \cdot \boldsymbol{x}| \geq 2r||\boldsymbol{x}||\right\}$, which concludes the proof.

We note that the proof strongly relies on two assumptions. The first is that the activation function is Leaky ReLU. The result is not true for ReLU networks (see supplementary for an example). The second is that the clusters correspond to the $\boldsymbol{w}$ and $\boldsymbol{u}$ sets of neurons.

5.1. Experiments

Theorem 5.1 states that if neurons cluster, the resulting decision boundary will be approximately linear. But do neurons actually cluster in practice, and what is the resulting r? In Figure 2 we show the value of r during training. We use this r to calculate the linear regime in Theorem 5.1 and the fraction of train and test points that fall outside this regime. It can be seen that for small initialization, this fraction converges to zero, implying that the learned classifiers are effectively linear over the data. Additional experiments in the supplementary provide support for the neurons being tightly clustered and r being very small.

Theorem 5.1 shows that a well clustered network leads to a linear decision boundary. However, it does *not* imply that the network output itself is a linear function of the input. Figure 3 provides a nice illustration of this fact.

6. On Conditions for Convergence to Clustered Solutions

Figure 1 suggests that gradient methods converge to a network with a linear decision boundary when trained on linearly separable data. Understanding when this occurs is important, because a model with a linear decision boundary has good generalization guarantees.[5]

In the previous section we saw that clustering of neurons to two directions implies that the network has an approximate linear decision boundary. Therefore, this reduces the problem of proving that the network has a linear decision boundary to proving that the network neurons are well clustered. It remains to show under which conditions gradient methods converge to clustered solutions.

Providing an end-to-end analysis which shows that gradient methods converge to clustered solutions is a major challenge. In this section we provide initial results for tackling this problem. In Section 6.1 we derive a novel condition on the optimization trajectory which implies that the network converges to a clustered solution and therefore to a linear decision boundary. In Section 6.2 we study a special case where a more fine-grained characterization of the linear decision boundary can be derived using a convex optimization program. Finally, we empirically validate our findings in Section 6.3.

To obtain the results in this section, we apply recent results of Lyu & Li (2020) and (Ji & Telgarsky, 2020) and therefore make the same assumptions presented in these papers. Specifically, we assume that we run **gradient flow** (GF) as defined in Section 3. We further assume that we are in the late phase of training:

Assumption 6.1. *There exists t_0 such that $L_{\mathbb{S}}(\boldsymbol{W}_{t_0}) < \frac{1}{n}$.*

We note that by the results in Section 4, SGD can attain the loss value in Assumption 6.1. However, in this section we need this assumption because we consider gradient flow and not SGD.

6.1. A Sufficient Condition

We first observe that using Theorem 5.1 we can conclude that when the neurons are perfectly clustered around two directions (i.e., $r = 0$), the decision boundary is linear. We formally define this below.

Definition 6.1. *A network $N_{\boldsymbol{W}}(\boldsymbol{x})$ is perfectly clustered if for all $1 \leq i, j \leq k$ it holds that: $\boldsymbol{w}^{(i)} = \boldsymbol{w}^{(j)}$ and $\boldsymbol{u}^{(i)} = \boldsymbol{u}^{(j)}$.*

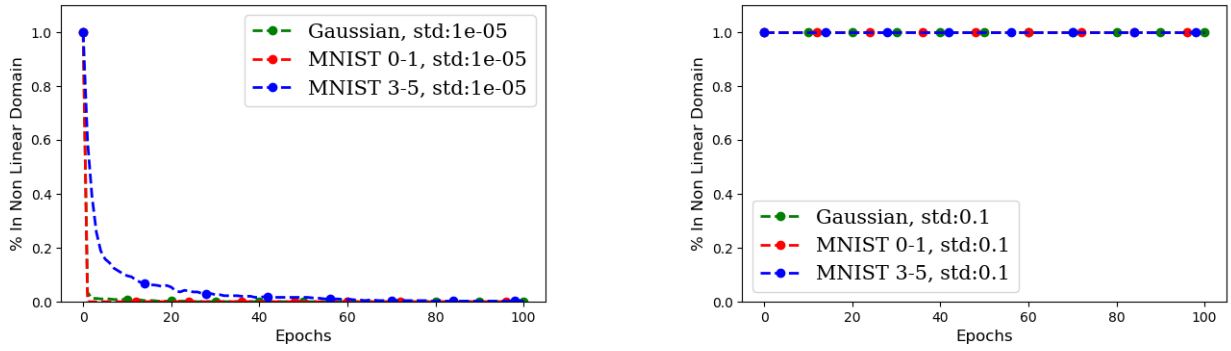
By applying Theorem 5.1 with $r = 0$, we have:

Corollary 6.1. *If a network $N_{\boldsymbol{W}}$ is perfectly clustered, then its decision boundary is linear for all $\boldsymbol{x} \in \mathbb{R}^d$.*

For completeness we provide a proof in the supplementary (this result is easier to prove directly than Theorem 5.1).

The key question that remains is under which conditions is the learned network perfectly clustered? To address this, we define a novel condition on the optimization trajectory that implies clustering. We define the Neural Agreement Regime (NAR) of weights of a network as follows. Informally, a network is in the NAR regime if all the $\boldsymbol{w}$ neurons "agree" on the classification of the training data and likewise for the $\boldsymbol{u}$ neurons. Classification in both

[5] For example, standard VC bounds imply $O(\sqrt{d/n})$ sample complexity in this case.



(a) (b)

Figure 2: Empirical evaluation of the linear decision boundary prediction of Theorem 5.1. A network is trained on Gaussian data and binary MNIST problems. At each epoch, the clustering level r is calculated, and the corresponding linear decision region in Theorem 5.1. Finally, all data points (train and test) are checked to see if they are in the linear region or not. The figure reports the fraction of points in the non-linear region. It can be seen that (a) for small initialization the fraction quickly decreases to zero whereas (b) for large initialization scale it does not.

cases is within a specified margin of β. Define $\overrightarrow{\boldsymbol{W}} = (\boldsymbol{w}^{(1)}, ..., \boldsymbol{w}^{(k)}, \boldsymbol{u}^{(1)}, ..., \boldsymbol{u}^{(k)}) \in \mathbb{R}^{2kd}$, $\hat{\boldsymbol{w}}^{(l)} = \frac{\boldsymbol{w}^{(l)}}{||\boldsymbol{w}^{(l)}||}$ and $\hat{\boldsymbol{u}}^{(l)} = \frac{\boldsymbol{u}^{(l)}}{||\boldsymbol{u}^{(l)}||}$. Let $N_{\hat{\boldsymbol{W}}}$ be the network with normalized parameters $\hat{\boldsymbol{W}} = \frac{\boldsymbol{W}}{||\boldsymbol{W}||}$. Then, NAR is defined as follows:

Definition 6.2. *Let $\beta > 0$, $\boldsymbol{c}^{\boldsymbol{w}} \in \{-1, 1\}^n$ and $\boldsymbol{c}^{\boldsymbol{u}} \in \{-1, 1\}^n$. We define a Neural Agreement Regime (NAR) $\mathcal{N}$ with parameters $(\beta, \boldsymbol{c}^{\boldsymbol{w}}, \boldsymbol{c}^{\boldsymbol{u}})$, to be the set of all parameters $\overrightarrow{\boldsymbol{W}}$ such that for all $\boldsymbol{x}_i \in \mathbb{S}$ and $1 \leq l \leq k$ it holds that (1) $c_i^{\boldsymbol{w}} \hat{\boldsymbol{w}}^{(l)} \cdot \boldsymbol{x}_i \geq \beta$ and (2) $c_i^{\boldsymbol{u}} \hat{\boldsymbol{u}}^{(l)} \cdot \boldsymbol{x}_i \geq \beta$.*

Note that the value $c_i^{\boldsymbol{w}}$ determines the *agreement* of the $\boldsymbol{w}$ neurons on the point $\boldsymbol{x}_i$. Indeed, if $c_i^{\boldsymbol{w}} = 1$, then for $1 \leq l \leq k$ it holds that $\hat{\boldsymbol{w}}^{(l)} \cdot \boldsymbol{x}_i \geq \beta$. Similarly, $c_i^{\boldsymbol{u}}$ determines the agreement of the $\boldsymbol{u}$ neurons on $\boldsymbol{x}$.

Importantly, if a network is in an NAR then its neurons can be "far" from being perfectly clustered. Namely, the angles between the normalized weights of different neurons can be relatively large. Next, we show a non-trivial fact: if gradient flow enters an NAR at some time T_{NAR} and stays in it, then it will converge to a perfectly clustered network.

Theorem 6.1. *Assume that Assumption 6.1 holds and consider the NAR regime $\mathcal{N}$ with parameters $(\beta, \boldsymbol{c}^{\boldsymbol{w}}, \boldsymbol{c}^{\boldsymbol{u}})$. Assume that there exists a time $T_{NAR} \geq t_0$ such that for all $t \geq T_{NAR}$ it holds that $\overrightarrow{\boldsymbol{W}} \in \mathcal{N}$. Then, gradient flow converges to a solution in $\mathcal{N}$ and at convergence the network with normalized parameters $N_{\hat{\boldsymbol{W}}}(\boldsymbol{x})$ is perfectly clustered.*

Theorem 6.1 says that if training is such that the trajectory enters an NAR and never leaves it, then the network will become perfectly clustered. The proof uses results from Lyu & Li (2020) and Ji & Telgarsky (2020) that together guarantee convergence of gradient flow to a KKT point of a minimum norm optimization problem. The theorem then follows from a simple observation that in an NAR, the KKT conditions imply that the network is perfectly clustered. The proof is in the supplementary.

Using Corollary 6.1 we immediately obtain the following.

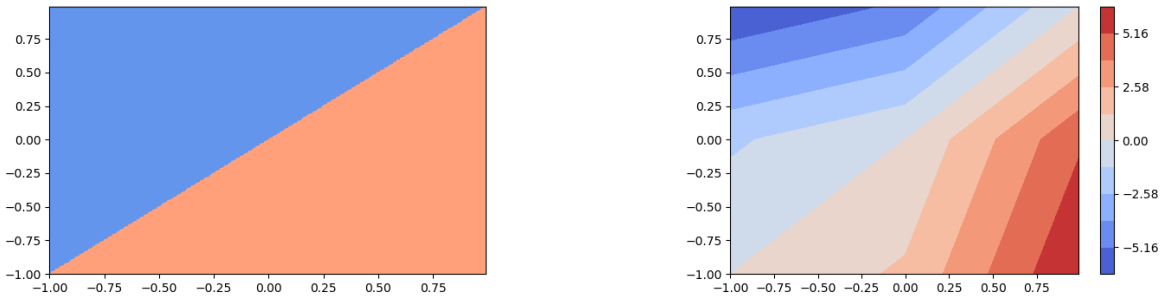
Corollary 6.2. *Under the assumptions in Theorem 6.1, GF converges to a network with a linear decision boundary.*

Therefore, we see that if a network is at an NAR from some time T_{NAR}, then it will converge to a solution with a linear decision boundary. The question that remains is whether networks indeed converge to an NAR and remain there.

6.2. The Perfect Agreement Regime

To better understand convergence to NARs, in this section we study a specific NAR for which we provide a more fine-grained analysis. We identify conditions on the training data and optimization trajectory that imply that gradient flow converges to an NAR which we call the Perfect Agreement Regime (PAR). Using Theorem 6.1 and results from Lyu & Li (2020); Ji & Telgarsky (2020), we provide a complete characterization of the weights that gradient flow converges to in this case. Admittedly, the conditions on the data and optimization trajectory are fairly strong. Nonetheless, we show that our theoretical results accurately predict the dynamics that we observe in experiments. Indeed, in Section 6.3 we show empirically that for certain linearly separable datasets, gradient flow converges to a solution in the PAR which is in agreement with our results.

In the PAR, each neuron classifies the data perfectly. Namely, all $\boldsymbol{w}$ neurons classify like the ground truth $\boldsymbol{w}^*$, and all $\boldsymbol{u}$ neurons classify like $-\boldsymbol{w}^*$. Formally, let $\boldsymbol{y} =$



(a) (b)

Figure 3: The decision boundary and network output values for a two neuron leaky ReLU network with two-dimensional inputs. Figure (a) shows the decision boundary sign $(N_{\boldsymbol{W}}(\boldsymbol{x}))$ and (b) shows the network output $N_{\boldsymbol{W}}(\boldsymbol{x})$. It can be seen that the decision boundary is linear but the network output itself is not linear.

$(y_1, ..., y_n)$. Then PAR is defined as follows.

Definition 6.3 (Perfect Agreement Regime)**.** *Given training data $\mathbb{S}$ with labels $\boldsymbol{y}$, the PAR(β) is the NAR with parameters $(\beta, \boldsymbol{y}, -\boldsymbol{y})$.*

Note that the fact that a network is in PAR does not mean that $\boldsymbol{w}_i = -\boldsymbol{u}_j$. Indeed, PAR only requires that $\boldsymbol{w}_i$ and $-\boldsymbol{u}_j$ both correctly classify the training set.

Next, we provide conditions under which a network will converge to a PAR. The conditions require a lower bound on the network smoothed margin (Eq. (4)), as well as a separability condition on the data. To define the separability condition we consider the following:

$$\mathbb{V}_{\beta}^{+}(\mathbb{S}) \coloneqq \{\boldsymbol{v} \in \mathbb{R}^d \mid \forall \boldsymbol{x} \in \mathbb{S}_+ \quad \hat{\boldsymbol{v}} \cdot \boldsymbol{x} \geq \beta, \\ \exists \boldsymbol{x} \in \mathbb{S}_- \quad \text{s.t. } \hat{\boldsymbol{v}} \cdot \boldsymbol{x} \geq \beta\}$$

Namely, $\mathbb{V}_{\beta}^{+}(\mathbb{S})$ is the set of vectors that classifies the positive points correctly and incorrectly classifies at least one of the negative points as a positive one, where all classifications are with margin β. Similarly we define:

$$\mathbb{V}_{\beta}^{-}(\mathbb{S}) \coloneqq \{\boldsymbol{v} \in \mathbb{R}^d \mid \forall \boldsymbol{x} \in \mathbb{S}_- \quad \hat{\boldsymbol{v}} \cdot \boldsymbol{x} \geq \beta, \\ \exists \boldsymbol{x} \in \mathbb{S}_+ \quad \text{s.t. } \hat{\boldsymbol{v}} \cdot \boldsymbol{x} \geq \beta\}$$

Thus, $\mathbb{V}_{\beta}^{-}(\mathbb{S})$ is the same as $\mathbb{V}_{\beta}^{+}(\mathbb{S})$ but with the roles of $\mathbb{S}_+$ and $\mathbb{S}_-$ reversed. With these definitions we can provide a sufficient condition for convergence to PAR.

Theorem 6.2. *Assume that:*

1. *Assumption 6.1 holds.*
2. *There exists an NAR $\mathcal{N}$ and $T_{NAR} \geq t_0$ such that for all $t \geq T_{NAR}$ it holds that $\overrightarrow{\boldsymbol{W}}_t \in \mathcal{N}$.*
3. *There exists $T_{Margin} \geq T_{NAR}$ such that $\tilde{\gamma}_{T_{Margin}} > \sqrt{k}\alpha v \cdot \max_{\boldsymbol{x} \in \mathbb{S}} ||x_i||$*
4. *The training data $\mathbb{S}$ satisfies $\mathbb{V}_{\beta}^{+}(\mathbb{S}) = \mathbb{V}_{\beta}^{-}(\mathbb{S}) = \emptyset$.*

Then $\mathcal{N}$ is a PAR(β) for all $t > T_{Margin}$, and there exists $\delta_w, \delta_u > 0$ such that gradient flow converges to a network whose normalized version is perfectly clustered with neuron directions $\hat{\boldsymbol{w}}, \hat{\boldsymbol{u}}$, where $(\delta_w \hat{\boldsymbol{w}}, \delta_u \hat{\boldsymbol{u}})$ is the solution to the following convex optimization problem:

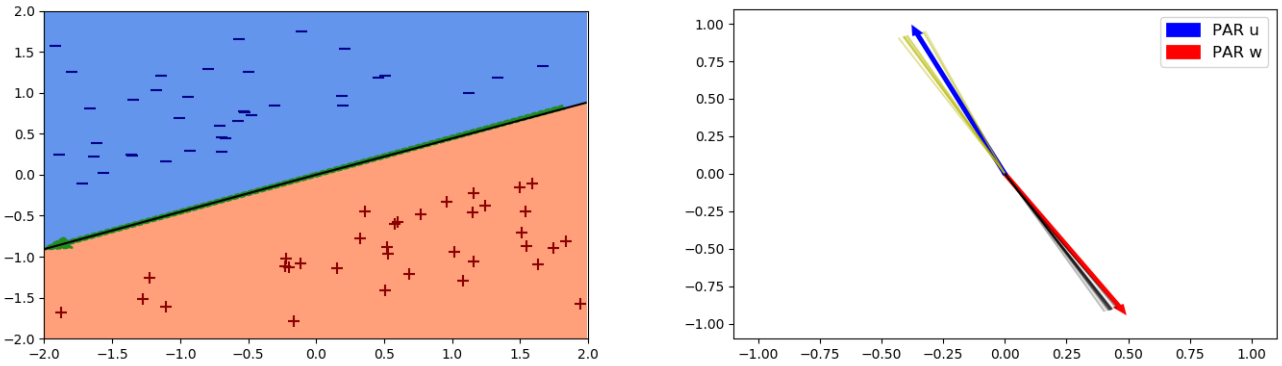
$$\arg\min_{\boldsymbol{w} \in \mathbb{R}^d, \boldsymbol{u} \in \mathbb{R}^d} \quad ||\boldsymbol{w}||^2 + ||\boldsymbol{u}||^2 \tag{7}$$
$$\forall \boldsymbol{x}_+ \in \mathbb{S}_+ : \boldsymbol{w} \cdot \boldsymbol{x}_+ - \alpha \boldsymbol{u} \cdot \boldsymbol{x}_+ \geq 1$$
$$\forall \boldsymbol{x}_- \in \mathbb{S}_- : \boldsymbol{u} \cdot \boldsymbol{x}_- - \alpha \boldsymbol{w} \cdot \boldsymbol{x}_- \geq 1$$

We first comment on the assumptions. The first two assumptions are the same assumptions on the optimization trajectory as in Theorem 6.1. Assumption 3 is another assumption on the trajectory that says that sufficiently large smoothed margin is achieved at some stage of the optimization. We note that the lower bound on the smoothed margin can be made small by considering a small α.

Assumption 4 refers to the training set. Informally, it corresponds to requiring that the two classes are approximately symmetric with respect to the origin. The next lemma shows that a certain symmetric training set satisfies Assumption 4:

Lemma 6.1. *Assume that for any $\boldsymbol{x} \in \mathbb{S}$ it holds that $-\boldsymbol{x} \in \mathbb{S}$. Then, for any $\beta > 0$, $\mathbb{V}_{\beta}^{+}(\mathbb{S}) = \mathbb{V}_{\beta}^{-}(\mathbb{S}) = \emptyset$.*

The proof is given in the supplementary. This example suggests that we should observe PAR in symmetric distributions, which produce approximately symmetric training sets. Indeed, we empirically show in Section 6.3 that gradient flow converges to a solution in PAR for a distribution with



(a) Learned Decision Boundary and PAR Solution

(b) Learned Neurons and PAR Solutions

Figure 4: Illustration of the Perfect Agreement Regime (PAR): A network with 100 neurons is trained on linearly separable data sampled from two Gaussians, and points inside a linear margin are excluded. Figure (a) shows that the network learns a linear decision boundary. Furthermore, the green arrow shows the decision boundary predicted by the PAR result (optimization problem Eq. (7)), and it agrees with the learned boundary. Figure (b) shows the learned neurons (yellow lines for $\boldsymbol{u}$ neurons and grey lines for $\boldsymbol{w}$ neurons), as well as the theoretical PAR solutions. It can be seen that neurons indeed converge to the PAR solution.

two symmetric Gaussians. We note that this example shows that Assumption 4 is independent of the maximum margin attainable on the training set. Indeed, by scaling the points, we can obtain any margin and still satisfy the assumption.

The theorem not only implies that convergence will be to a PAR, but it provides the solution that GF will converge to. The optimization problem in Eq. (7) is an SVM optimization problem with the kernel: $K(\boldsymbol{x}, \boldsymbol{x}') = \sum_{y\in\{-1,1\}} \sigma'(y\boldsymbol{w}^* \cdot \boldsymbol{x})\sigma'(y\boldsymbol{w}^* \cdot \boldsymbol{x}')\boldsymbol{x} \cdot \boldsymbol{x}'$. The corresponding feature map is: $\phi(\boldsymbol{x}) = [\sigma'(\boldsymbol{w}^* \cdot \boldsymbol{x})\boldsymbol{x}, -\sigma'(-\boldsymbol{w}^* \cdot \boldsymbol{x})\boldsymbol{x}] \in \mathbb{R}^{2d}$.

We prove Theorem 6.2 in the supplementary, and provide a sketch next. First, we use Theorem 6.1 to show that gradient flow converges to an NAR and the neurons are clustered. Then we show that under Assumption 3 and using the monotonicity of the smoothed margin (Eq. (5)), by Lyu & Li (2020), all $\boldsymbol{w}$ neurons classify the positive points correctly and all $\boldsymbol{u}$ neurons classify the negative points correctly for all $t > T_{Margin}$. Then, using Assumption 4 we show that the solution is in PAR. Finally, we use results of Lyu & Li (2020) to show that the network directions solve the convex optimization problem in the theorem.

6.3. Experiments

In Theorem 6.2 we show that when learning enters the PAR regime the solution will be given by Eq. (7). We performed experiments in several settings that show the above behavior is observed in practice when classes are sampled from Gaussians. Figure 4 shows the decision boundary (Figure 4a) and learned weights (Figure 4b), for learning from points sampled from two classes corresponding to Gaussians. The figure also shows the PAR predictions for the decision boundary and learned weights, and these show excellent agreement with the empirical results. We have also verified that in this case convergence is indeed to a PAR solution. We performed such experiments also for higher dimensional settings, and the results are in the supplementary. Finally, note that we do not expect learning to always converge to a PAR. In the supplementary we show an example where this does not happen.

7. Conclusions

Optimization and generalization are closely coupled in deep-learning. Yet both are little understood even for simple models. Here we consider perhaps the simplest "teacher" model where the ground truth is linear. We prove that cross-entropy can be globally minimized by SGD, despite the non-convexity of the loss, and for any initialization scale. We are not aware of any such result for non-linear networks (for example NTK optimization results require large initialization scale, and sufficiently wide networks (Ji & Telgarsky, 2019b)). Our novel proof technique analyzes SGD in an offline setting and uses the notion of loss-violation per epoch, which we believe could be useful elsewhere.

In our setting, small initialization scale leads empirically to approximately linear decision boundaries. We prove that such boundaries are obtained when neurons with same output-weight sign are clustered. Empirically we show that such clustering indeed occurs. Moreover, we provide sufficient conditions for converging to such clustered solutions.

Several open questions remain. The first is reducing the assumptions when proving convergence to a clustered solution. Another interesting direction is extending our results to simple non-linear teachers.

8. Acknowledgements

This research is supported by the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (grant ERC HOLI 819080) and by the Yandex Initiative in Machine Learning at Tel Aviv University. AB is supported by the Google Doctoral Fellowship in Machine Learning.

References

Allen-Zhu, Z. and Li, Y. What can resnet learn efficiently, going beyond kernels? *arXiv preprint arXiv:1905.10337*, 2019.

Allen-Zhu, Z., Li, Y., and Song, Z. A convergence theory for deep learning via over-parameterization. In *International Conference on Machine Learning*, pp. 242–252. PMLR, 2019.

Arora, S., Du, S., Hu, W., Li, Z., and Wang, R. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pp. 322–332, 2019.

Blum, A. L. and Rivest, R. L. Training a 3-node neural network is np-complete. *Neural Networks*, 5(1):117–127, 1992.

Brutzkus, A. and Globerson, A. Why do larger models generalize better? a theoretical perspective via the xor problem. In *International Conference on Machine Learning*, pp. 822–830. PMLR, 2019.

Brutzkus, A., Globerson, A., Malach, E., and Shalev-Shwartz, S. Sgd learns over-parameterized networks that provably generalize on linearly separable data. *International Conference on Learning Representations*, 2018.

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

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

Chizat, L. and Bach, F. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In Abernethy, J. D. and Agarwal, S. (eds.), *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*, volume 125 of *Proceedings of Machine Learning Research*, pp. 1305–1338. PMLR, 2020. URL http://proceedings.mlr.press/v125/chizat20a.html.

Daniely, A. and Malach, E. Learning parities with neural networks. *arXiv preprint arXiv:2002.07400*, 2020.

Daniely, A., Frostig, R., and Singer, Y. Toward deeper understanding of neural networks: The power of initialization and a dual view on expressivity. In *Advances In Neural Information Processing Systems*, pp. 2253–2261, 2016.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT (1)*, 2019.

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

Du, S. S., Zhai, X., Poczos, B., and Singh, A. Gradient descent provably optimizes over-parameterized neural networks. *International Conference on Learning Representations*, 2018.

Fiat, J., Malach, E., and Shalev-Shwartz, S. Decoupling gating from linearity. *arXiv preprint arXiv:1906.05032*, 2019.

Gunasekar, S., Lee, J. D., Soudry, D., and Srebro, N. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, pp. 9461–9471, 2018.

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

Ji, Z. and Telgarsky, M. Gradient descent aligns the layers of deep linear networks. *ICLR*, 2019a.

Ji, Z. and Telgarsky, M. Polylogarithmic width suffices for gradient descent to achieve arbitrarily small test error with shallow relu networks. In *International Conference on Learning Representations*, 2019b.

Ji, Z. and Telgarsky, M. Directional convergence and alignment in deep learning, 2020.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

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

Li, Y., Ma, T., and Zhang, H. R. Learning over-parametrized two-layer neural networks beyond NTK. In *Conference on Learning Theory*, pp. 2613–2682, 2020.

Lyu, K. and Li, J. Gradient descent maximizes the margin of homogeneous neural networks. *ICLR*, 2020.

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

Moroshko, E., Gunasekar, S., Woodworth, B., Lee, J. D., Srebro, N., and Soudry, D. Implicit bias in deep linear classification: Initialization scale vs training accuracy. *arXiv preprint arXiv:2007.06738*, 2020.

Nacson, M. S., Gunasekar, S., Lee, J., Srebro, N., and Soudry, D. Lexicographic and depth-sensitive margins in homogeneous and non-homogeneous deep models. In *International Conference on Machine Learning*, pp. 4683–4692, 2019.

Phuong, M. and Lampert, C. The inductive bias of relu networks on orthogonally separable data. *ICLR*, 2021.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

Wang, G., Giannakis, G. B., and Chen, J. Learning relu networks on linearly separable data: Algorithm, optimality, and generalization. *IEEE Transactions on Signal Processing*, 67(9):2357–2370, May 2019. ISSN 1941-0476. doi: 10.1109/tsp.2019.2904921. URL http://dx.doi.org/10.1109/TSP.2019.2904921.

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

Woodworth, B. E., Gunasekar, S., Lee, J. D., Moroshko, E., Savarese, P., Golan, I., Soudry, D., and Srebro, N. Kernel and rich regimes in overparametrized models. In Abernethy, J. D. and Agarwal, S. (eds.), *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*, volume 125 of *Proceedings of Machine Learning Research*, pp. 3635–3673. PMLR, 2020. URL http://proceedings.mlr.press/v125/woodworth20a.html.

Yehudai, G. and Shamir, O. On the power and limitations of random features for understanding neural networks. In *Advances in Neural Information Processing Systems*, pp. 6594–6604, 2019.