Why do Larger Models Generalize Better? A Theoretical Perspective via the XOR Problem

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

Abstract

Empirical evidence suggests that neural networks with ReLU activations generalize better with over-parameterization. However, there is currently no theoretical analysis that explains this observation. In this work, we provide theoretical and empirical evidence that, in certain cases, overparameterized convolutional networks generalize better than small networks because of an interplay between weight clustering and feature exploration at initialization. We demonstrate this theoretically for a 3-layer convolutional neural network with max-pooling, in a novel setting which extends the XOR problem. We show that this interplay implies that with overparameterization, gradient descent converges to global minima with better generalization performance compared to global minima of small networks. Empirically, we demonstrate these phenomena for a 3-layer convolutional neural network in the MNIST task.

1. Introduction

Most successful deep learning models use more parameters than needed to achieve zero training error. This is typically referred to as *overparameterization*. Indeed, it can be argued that overparameterization is one of the key techniques that has led to the remarkable success of neural networks. However, there is still no theoretical account for its effectiveness.

One very intriguing observation in this context is that over-parameterized networks with ReLU activations, which are trained with gradient based methods, often exhibit better generalization error than smaller networks (Neyshabur et al., 2014; 2019; Novak et al., 2018). In particular, it often happens that two networks, one with N_1 neurons and one with $N_2 > N_1$ neurons achieve zero training error, but the larger network has better test error. This somewhat counter-intuitive observation suggests that first-order methods which are trained on overparameterized networks have an *inductive bias* towards solutions with better generalization performance. Understanding this inductive bias is a necessary step towards a full understanding of neural networks in practice.

Providing theoretical guarantees for overparameterizatied networks is extremely challenging. To show a generalization gap between smaller and larger models, one needs to prove that large networks have better sample complexity than smaller ones. However, current generalization bounds that are based on complexity measures do not offer such guarantees. Furthermore, analyzing the training dynamics of non-linear neural networks is a major challenge even for very simple learning tasks. It is thus natural to try and analyze a simplified scenario, which ideally shares various features with real-world settings.

In this work we follow this approach and show that a possible explanation for the success of overparameterization is a combination of two effects: weight exploration and weight clustering. Weight exploration refers to the fact that larger models explore the set of possible weights more effectively since they have more neurons in each layer. Weight clustering is an effect we demonstrate here, which refers to the fact that weight vectors in the same layer tend to cluster around a small number of prototypes.

To see *informally* how these effects act in the case of over-parameterization, consider a binary classification problem and a training set. The training set typically contains multiple patterns that discriminate between the two classes. The smaller network will find detectors (e.g., convolutional filters) for a subset of these patterns and reach zero training error, but not generalize because it is missing some of the patterns. This is a result of an under-exploration effect for the small net. On the other hand, the larger net has better exploration and will find more relevant detectors for classification. Furthermore, due to the clustering effect its weight vectors will be close to a small set of prototypes. Therefore the effective capacity of the overall model will be restricted, leading to good generalization.

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

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

The contribution of Alon B. is part of Ph.D. thesis research

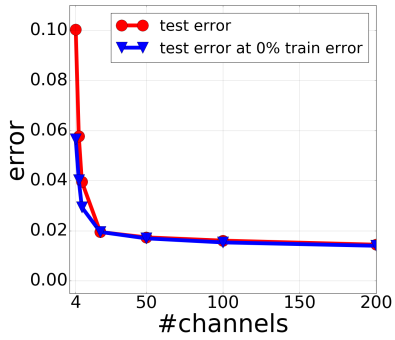


Figure 1: Overparameterization improves generalization in the XORD problem. The network in Eq. 2 is trained on data from the XORD problem (see Sec. 4). The figure shows the test error as a function of the number of channels k. The blue curve shows test error when restricting to cases where training error was zero. It can be seen that increasing the number of channels improves the generalization performance. Experimental details are provided in supplementary material.

.

The network we study here includes some key architectural components used in modern machine learning models. Specifically, it consists of a convolution layer with a ReLU activation function, followed by a max-pooling operation, and a fully-connected layer. This is a key component of most machine-vision models, since it can be used to detect patterns in an input image. We are also not aware of any theoretical guarantees for a network of this structure.

For this architecture, we consider the problem of detecting two dimensional binary patterns in a high dimensional input vector. The patterns we focus on are the XOR combination (i.e., $(1,1)$ or $(-1,-1)$). This problem is a high dimensional extension of the XOR problem. We refer to it as the "XOR Detection problem (XORD)". One advantage of this setting is that it nicely exhibits the phenomenon of overparameterization empirically, and is therefore a good test-bed for understanding overparameterization. Fig. 1 shows the result of learning the XORD problem with the above network, as a function of the number of channels. It can be seen that increasing the number of channels improves test error.[1]

Motivated by these empirical observations, we present a theoretical analysis of optimization and generalization in the XORD problem. Under certain distributional assumptions, we will show that overparameterized networks enjoy a combination of better exploration of features at initialization and clustering of weights, leading to better generalization for overparameterized networks.

conducted at Tel Aviv University.

[1]Note that a similar curve is observed when only considering zero training error, implying that smaller networks are expressive enough to fit the training data.

Importantly, we show empirically that our insights from the XORD problem transfer to other settings. In particular, we see a similar phenomenon when learning on the MNIST data, where we verify that weights are clustered at convergence and observe better exploration of weights for large networks.

Finally, another contribution of our work is the first proof of convergence of gradient descent in the classic XOR problem with inputs in $\{\pm 1\}^2$. The proof is simple and conveys the key insights of the analysis of the general XORD problem. See Section 3 for further details.

2. Related Work

In recent years there have been many works on theoretical aspects of deep learning. We will refer to those that are most relevant to this work. First, we note that we are not aware of any work that shows that generalization performance provably improves with over-parameterization. This distinguishes our work from all previous works.

Several works study convolutional networks with ReLU activations and their properties (Du et al., 2018b;c; Brutzkus & Globerson, 2017). All of these works consider convolutional networks with a single channel. Recently, there have been numerous works that provide guarantees for gradient-based methods in general settings (Daniely, 2017; Li & Liang, 2018; Du et al., 2018c;a; Allen-Zhu et al., 2018). However, their analysis holds for over-parameterized networks with an extremely large number of neurons that is not used in practice (e.g., the number of neurons is a very large polynomial of certain problem parameters). Furthermore, we consider a 3-layer convolutional network with max-pooling which is not studied in these works. Several works (Ji & Telgarsky, 2019; Gunasekar et al., 2018; Arora et al., 2019) provide guarantees for gradient descent on *linear* networks.

Over-parameterization has also been studied for quadratic activation functions (Soltanolkotabi et al., 2018; Du & Lee, 2018; Li et al., 2018). Brutzkus et al. (2018) provide generalization guarantees for over-parameterized networks with Leaky ReLU activations on linearly separable data. Neyshabur et al. (2019) prove generalization bounds for neural networks. However, it is not shown that networks found by gradient based methods give low generalization bounds.

3. Warm up: the XOR Problem

We begin by studying the simplest form of our model: the classic XOR problem in two dimensions.[2] We will show that this problem illustrates the key phenomena that allow overparameterized networks to perform better than smaller

[2]XOR is a specific case of XORD in Sec. 4 where $d = 1$.

ones. Namely, exploration at initialization and clustering during training. For the XOR problem, this will imply that overparameterized networks have better *optimization* performance. In later sections, we will show that the same phenomena occur for higher dimensions in the XORD problem and imply better *generalization* of global minima for overparameterized convolutional networks.

3.1. Problem Formulation

In the XOR problem, we are given a training set $S = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^4 \subseteq \{\pm 1\}^2 \times \{\pm 1\}$ consisting of points $\boldsymbol{x}_1 = (1,1)$, $\boldsymbol{x}_2 = (-1,1)$, $\boldsymbol{x}_3 = (-1,-1)$, $\boldsymbol{x}_4 = (1,-1)$ with labels $y_1 = 1$, $y_2 = -1$, $y_3 = 1$ and $y_4 = -1$, respectively. Our goal is to learn the XOR function $f^* : \{\pm 1\}^2 \to \{\pm 1\}$, such that $f^*(\boldsymbol{x}_i) = y_i$ for $1 \le i \le 4$, with a neural network and gradient descent.

Neural Architecture: For this task we consider the following two-layer fully connected network.

$$N_W(\boldsymbol{x}) = \sum_{i=1}^{k} \left[\sigma\left(\boldsymbol{w}^{(i)} \cdot \boldsymbol{x}\right) - \sigma\left(\boldsymbol{u}^{(i)} \cdot \boldsymbol{x}\right)\right] \quad (1)$$

where $W \in \mathbb{R}^{2k \times 2}$ is the weight matrix whose rows are the $\boldsymbol{w}^{(i)}$ vectors followed by the $\boldsymbol{u}^{(i)}$ vectors, and $\sigma(x) = \max\{0, x\}$ is the ReLU activation applied element-wise. We note that f^* can be implemented with this network for $k = 2$ and this is the minimal k for which this is possible. Thus we refer to $k > 2$ as the overparameterized case.

Training Algorithm: The parameters of the network $N_W(\boldsymbol{x})$ are learned using gradient descent on the hinge loss objective. We use a constant learning rate $\eta = \frac{c_\eta}{k}$, where $c_\eta < \frac{1}{2}$. The parameters N_W are initialized as IID Gaussians with zero mean and standard deviation $\sigma_g \le \frac{c_\eta}{16k^{3/2}}$. We consider the hinge-loss objective:

$$\ell(W) = \sum_{(\boldsymbol{x},y)\in S} \max\{1 - yN_W(\boldsymbol{x}), 0\}$$

where optimization is only over the first layer of the network. We note that for $k \ge 2$ any global minimum W of ℓ satisfies $\ell(W) = 0$ and $\operatorname{sign}(N_W(\boldsymbol{x}_i)) = f^*(\boldsymbol{x}_i)$ for $1 \le i \le 4$.

Notations: We will need the following notations. Let W_t be the weight matrix at iteration t of gradient descent. For $1 \le i \le k$, denote by $\boldsymbol{w}_t^{(i)} \in \mathbb{R}^2$ the i^{th} weight vector at iteration t. Similarly we define $\boldsymbol{u}_t^{(i)} \in \mathbb{R}^2$ to be the $k+i$ weight vector at iteration t. For each point $\boldsymbol{x}_i \in S$ define the following sets of neurons:

$$W_t^+(i) = \left\{j \mid \boldsymbol{w}_t^{(j)} \cdot \boldsymbol{x}_i > 0\right\}$$
$$U_t^+(i) = \left\{j \mid \boldsymbol{u}_t^{(j)} \cdot \boldsymbol{x}_i > 0\right\}$$

and for each iteration t, let $a_i(t)$ be the number of iterations $0 \le t' \le t$ such that $y_i N_{W_{t'}}(\boldsymbol{x}_i) < 1$.

3.2. Over-parameterized Networks Optimize Well

In this section we assume that $k > 16$. The following lemma shows that with high probability, for every training point, overparameterized networks are initialized at directions that have positive correlation with the training point. The proof uses a standard measure concentration argument. We refer to this as "exploration" as it lets the optimization procedure explore these parts of weight space.

Lemma 3.1. ***Exploration at Initialization.*** *With probability at least* $1 - 8e^{-8}$*, for all* $1 \le i \le 4$

$$\frac{k}{2} - 2\sqrt{k} \le \left|W_0^+(i)\right|, \left|U_0^+(i)\right| \le \frac{k}{2} + 2\sqrt{k}$$

Next, we show an example of the weight dynamics which imply that the weights tend to cluster around a few directions. The proof uses the fact that with high probability the initial weights have small norm and proceeds by induction on t to show the dynamics.

Lemma 3.2. ***Clustering Dynamics.*** *Let* $i \in \{1, 3\}$*. With probability* $\ge 1 - \frac{\sqrt{8k}}{\sqrt{\pi}e^{8k}}$*, for all* $t \ge 0$ *and* $j \in W_0^+(i)$ *there exists a vector* $\boldsymbol{v}_t$ *such that* $\boldsymbol{v}_t \cdot \boldsymbol{x}_i > 0$*,* $|\boldsymbol{v}_t \cdot \boldsymbol{x}_2| < 2\eta$ *and* $\boldsymbol{w}_t^{(j)} = a_i(t)\eta\boldsymbol{x}_i + \boldsymbol{v}_t$*.*

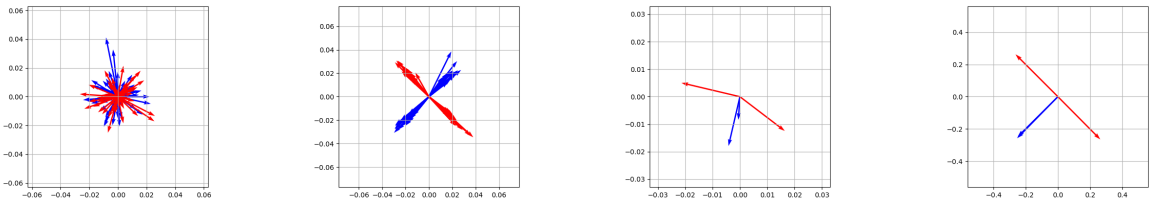
The sequence $\{a_i(t)\}_{t\ge 0}$ is non-decreasing and it can be shown that $a_i(0) = 1$ with high probablity. Therefore, the above lemma shows that for all $j \in W_0^+(i)$, $\boldsymbol{w}_t^{(j)}$ tends to cluster around $\boldsymbol{x}_i$ as t increases (as $\boldsymbol{v}_t$ is bounded in the direction orthogonal to $\boldsymbol{x}_i$ and $\boldsymbol{v}_t \cdot \boldsymbol{x}_i > 0$). Since with probability 1, $W_0^+(1) \cup W_0^+(3) = [k]$, the above lemma characterizes the dynamics of all filters $\boldsymbol{w}_t^{(j)}$. In the supplementary we show a similar result for the filters $\boldsymbol{u}_t^{(j)}$.

By applying both of the above lemmas, it can be shown that for $k > 16$ gradient descent converges to a global minimum with high probability and that the weights are clustered at convergence.

Theorem 3.3. ***Convergence and Clustering.*** *With probability* $\ge 1 - \frac{\sqrt{8k}}{\sqrt{\pi}e^{8k}} - 8e^{-8}$ *after* $T > \frac{1}{c_\eta}\left(\frac{16\sqrt{k}}{\sqrt{k}-4}\right)$ *iterations, gradient descent converges to a global minimum* W_T*. Furthermore, for* $i \in \{1, 3\}$ *and all* $j \in W_0^+(i)$*, the angle between* $\boldsymbol{w}_T^{(j)}$ *and* $\boldsymbol{x}_i$ *is at most* $\arccos\left(\frac{1-2c_\eta}{1+c_\eta}\right)$*. A similar result holds for* $\boldsymbol{u}_T^{(j)}$*.*

3.3. Small Network Fail to Optimize

In contrast to the case of large k, we show that for $k = 2$, the initialization does not explore all directions, leading to convergence to a suboptimal solution.



(a) (b) (c) (d)

Figure 2: Overparameterization and optimization in the XOR problem. The vectors in blue are the vectors $\boldsymbol{w}_t^{(i)}$ and in red are the vectors $\boldsymbol{u}_t^{(i)}$. (a) Exploration at initialization (t=0) for $k = 50$ (Lemma 3.1) (b) Clustering and convergence to global minimum for $k = 50$ (Lemma 3.2 and Theorem 3.3) (c) Non-sufficient exploration at initialization (t=0) for $k = 2$ (Theorem 3.4). (d) Convergence to local minimum (Theorem 3.4).

Theorem 3.4. ***Insufficient Exploration at Initialization.*** *With probability at least* 0.75*, there exists* $i \in \{1, 3\}$ *such that* $W_0^+(i) = \emptyset$ *or* $i \in \{2, 4\}$ *such that* $U_0^+(i) = \emptyset$*. As a result, with probability* ≥ 0.75*, gradient descent converges to a model which errs on at least one input pattern.*

3.4. Experiments

In this section we empirically demonstrate the theoretical results. We implemented the learning setting described in Sec. 3.1 and conducted two experiments: one with $k = 50$ and one with $k = 2$ We note that for $k = 2$ the XOR function f^* can be realized by the network in Eq. 1. Figure 2 shows the results. It can be seen that our theory nicely predicts the behavior of gradient descent. For $k = 50$ we see the effect of exploration at initialization and clustering which imply convergence to a global minimum. In contrast, the small network does not explore all directions at initialization and therefore converges to a local minimum. This is despite the fact that it has sufficient expressive power to implement f^*.

4. The XORD Problem

In the previous section we analyzed the XOR problem, showing that using a large number of channels allows gradient descent to learn the XOR function. This allowed us to understand the effect of overparameterization on optimization. However, it did not let us study generalization because in the learning setting all four examples were given, so that any model with zero training error also had zero test error.

In order to study the effect of overparameterization on generalization we consider a more general setting, which we refer to as the XOR Detection problem (XORD). As can be seen in Fig. 1, in the XORD problem large networks generalize better than smaller ones. This is despite the fact that small networks can reach zero training error. Our goal is to understand this phenomenon from a theoretical persepective.

In this section, we define the XORD problem. We begin with some notations and definitions. We consider a classification problem in the space $\{\pm 1\}^{2d}$, for $d \geq 1$. Given a vector $\boldsymbol{x} \in \{\pm 1\}^{2d}$, we consider its partition into d sets of two coordinates as follows $\boldsymbol{x} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_d)$ where $\boldsymbol{x}_i \in \{\pm 1\}^2$. We refer to each such $\boldsymbol{x}_i$ as a *pattern* in $\boldsymbol{x}$.

Neural Architecture: We consider learning with the following three-layer neural net model. The first layer is a convolutional layer with non-overlapping filters and multiple channels, the second layer is max pooling and the third layer is a fully connected layer with $2k$ hidden neurons and weights fixed to values ± 1. Formally, for an input $\boldsymbol{x} = (\boldsymbol{x}_1, ..., \boldsymbol{x}_d) \in \mathbb{R}^{2d}$ where $\boldsymbol{x}_i \in \mathbb{R}^2$, the output of the network is denoted by $N_W(\boldsymbol{x})$ and is given by:

$$\sum_{i=1}^{k} \left[\max_j \left\{ \sigma\left(\boldsymbol{w}^{(i)} \cdot \boldsymbol{x}_j \right) \right\} - \max_j \left\{ \sigma\left(\boldsymbol{u}^{(i)} \cdot \boldsymbol{x}_j \right) \right\} \right] \tag{2}$$

where notation is as in the XOR problem.

Remark 4.1. *Because there are only* 4 *different patterns, the network is limited in terms of the number of rules it can implement. Specifically, it is easy to show that its VC dimension is at most* 15 *(see supplementary material). Despite this limited expressive power, there is a generalization gap between small and large networks in this setting, as can be seen in Fig. 1, and in our analysis below.*

Data Generating Distribution: Next we define the classification rule we will focus on. Define the four two-dimensional binary patterns $\boldsymbol{p}_1 = (1, 1), \boldsymbol{p}_2 = (1, -1), \boldsymbol{p}_3 = (-1, -1), \boldsymbol{p}_4 = (-1, 1)$. Define $P_{pos} = \{\boldsymbol{p}_1, \boldsymbol{p}_3\}$ to be the set of positive patterns and $P_{neg} = \{\boldsymbol{p}_2, \boldsymbol{p}_4\}$ to be the set of negative patterns. Define the classification rule:

$$f^*(\boldsymbol{x}) = \begin{cases} 1 & \exists i \in \{1, \ldots, d\} : \boldsymbol{x}_i \in P_{pos} \\ -1 & \text{otherwise} \end{cases} \tag{3}$$

Namely, f^* detects whether a positive pattern appears in the input. For $d = 1$, f^* is the XOR classifier in Sec. 3.

Let $\mathcal{D}$ be a distribution over $\mathcal{X} \times \{\pm 1\}$ such that for all $(\boldsymbol{x}, y) \sim \mathcal{D}$ we have $y = f^*(\boldsymbol{x})$. We say that a point $(\boldsymbol{x}, y)$ is positive if $y = 1$ and negative otherwise. Let $\mathcal{D}_+$ be the marginal distribution over $\{\pm 1\}^{2d}$ of positive points and $\mathcal{D}_-$ be the marginal distribution of negative points.

For each point $\boldsymbol{x} \in \{\pm 1\}^{2d}$, define $P_{\boldsymbol{x}}$ to be the set of unique two-dimensional patterns that the point $\boldsymbol{x}$ contains, namely $P_{\boldsymbol{x}} = \{i \mid \exists j, \boldsymbol{x}_j = \boldsymbol{p}_i\}$. In the following definition we introduce the notion of *diverse* points, which will play a key role in our analysis.

Definition 4.2 (Diverse Points). *We say that a positive point $(\boldsymbol{x}, 1)$ is diverse if $P_{\boldsymbol{x}} = \{1, 2, 3, 4\}$.*[3] *We say that a negative point $(\boldsymbol{x}, -1)$ is diverse if $P_{\boldsymbol{x}} = \{2, 4\}$.*

For $\phi \in \{-, +\}$ define p_ϕ to be the probability that $\boldsymbol{x}$ is diverse with respect to $\mathcal{D}_\phi$. For example, if both D_+ and D_- are uniform, then by the inclusion-exclusion principle it follows that $p_+ = 1 - \frac{4 \cdot 3^d - 6 \cdot 2^d + 4}{4^d}$ and $p_- = 1 - \frac{1}{2^{d-1}}$.

Learning Setup: Our analysis will focus on the problem of learning f^* from training data with the three layer neural net model in Eq. 2. The learning algorithm will be gradient descent, randomly initialized. As in any learning task in practice, f^* is unknown to the training algorithm. Our goal is to analyze the performance of gradient descent when given data that is labeled with f^*. We assume that we are given a training set $S = S_+ \cup S_- \subseteq \{\pm 1\}^{2d} \times \{\pm 1\}$ where S_+ consists of m IID points drawn from $\mathcal{D}_+$ and S_- consists of m IID points drawn from $\mathcal{D}_-$.[4]

Importantly, we note that the function f^* can be realized by the above network with $k = 2$. Indeed, the network N_W with $\boldsymbol{w}^{(1)} = 3\boldsymbol{p}_1$, $\boldsymbol{w}^{(2)} = 3\boldsymbol{p}_3$, $\boldsymbol{u}^{(1)} = \boldsymbol{p}_2$, $\boldsymbol{u}^{(2)} = \boldsymbol{p}_4$ satisfies $\operatorname{sign}(N_W(\boldsymbol{x})) = f^*(\boldsymbol{x})$ for all $\boldsymbol{x} \in \{\pm 1\}^{2d}$. It can be seen that for $k = 1$, f^* cannot be realized. Therefore, any $k > 2$ is an overparameterized setting.

Training Algorithm: We will use gradient descent to optimize the following hinge-loss function.

$$\ell(W) = \frac{1}{m} \sum_{(\boldsymbol{x}_i, y_i) \in S_+ : y_i = 1} \max\{\gamma - N_W(\boldsymbol{x}_i), 0\} + \frac{1}{m} \sum_{(\boldsymbol{x}_i, y_i) \in S_- : y_i = -1} \max\{1 + N_W(\boldsymbol{x}_i), 0\} \quad (4)$$

for $\gamma \geq 1$.[5] We assume that gradient descent runs with a constant learning rate η and the weights are randomly initialized with IID Gaussian weights with mean 0 and standard deviation σ_g. Furthermore, only the weights of the first layer, the convolutional filters, are trained.[6] As in Section 3, we will use the notations W_t, $\boldsymbol{w}_t^{(i)}$, $\boldsymbol{u}_t^{(i)}$ for the weights at iteration t of gradient descent. At each iteration (starting from $t = 0$), gradient descent performs the update $W_{t+1} = W_t - \eta \frac{\partial \ell}{\partial W}(W_t)$.

5. XORD on Decoy Sets

In Fig. 1 we showed that the XORD problem exhibits better generalization for overparameterized models. Here we will empirically show how this comes about due to the effects of clustering and exploration. We compare two networks as in Sec. 4. The first has $k = 2$ (i.e., four hidden neurons) and the second has $k = 100$. As mentioned earlier, both these nets can achieve zero test error on the XORD problem.

We consider a *diverse* training set, namely, one which contains only diverse points. The set has 6 positive diverse points and 6 negative diverse points. Each positive point contains all the patterns $\{\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, \boldsymbol{p}_4\}$ and each negative point contains all the patterns $\{\boldsymbol{p}_2, \boldsymbol{p}_4\}$. Note that in order to achieve zero training error on this set, a network needs only to detect *at least* one of the patterns $\boldsymbol{p}_1$ or $\boldsymbol{p}_3$, and *at least* one of the patterns $\boldsymbol{p}_2$ or $\boldsymbol{p}_4$. For example, a network with $k = 2$ and filters $\boldsymbol{w}^{(1)} = \boldsymbol{w}^{(2)} = 3\boldsymbol{p}_1$, $\boldsymbol{u}^{(1)} = \boldsymbol{u}^{(2)} = \boldsymbol{p}_2$, has zero train loss. However, this network will not generalize to non-diverse points, where only a subset of the patterns appear. Thus we refer to it as a "decoy" training set.

Fig. 3 shows the results of training the $k = 2$ and $k = 100$ networks on the decoy training set. Both networks reach zero training error. However, the larger network learns the XORD function exactly, whereas the smaller network does not, and will therefore misclassify certain data points. As Fig. 3 clearly shows, the reason for the failure of the smaller network is that at initialization there is insufficient exploration of weight space. On the other hand, the larger network both explores well at initialization, and converges to clustered weights corresponding to all relevant patterns.

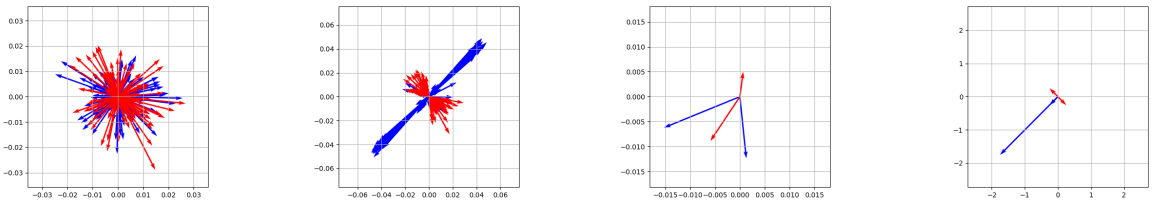
The above observations are for a training set that contains only diverse points. However, there are other decoy training sets which also contain non-diverse points (see supplementary for an example).

[3] This definition only holds for $d \geq 4$.

[4] For simplicity, we consider this setting of equal number of positive and negative points in the training set.

[5] In practice it is common to set γ to 1. In our analysis we will need $\gamma \geq 8$ to guarantee generalization. In the supplementary material we show empirically, that for this task, setting γ to be larger than 1 results in better test performance than setting $\gamma = 1$.

[6] Note that Hoffer et al. (2018) show that fixing the last layer to ± 1 does not degrade performance in various tasks. This assumption also appeared in Brutzkus et al. (2018); Li & Yuan (2017).



(a) (b) (c) (d)

Figure 3: Overparameterization and generalization in the XORD problem. The vectors in blue are the vectors $\boldsymbol{w}_t^{(i)}$ and in red are the vectors $\boldsymbol{u}_t^{(i)}$. (a) Exploration at initialization (t=0) for $k = 100$ (b) Clustering and convergence to global minimum that recovers f^* for $k = 100$ (c) Non-sufficient exploration at initialization (t=0) for $k = 2$. (d) Convergence to global minimum with non-zero test error for $k = 2$.

6. XORD Theoretical Analysis

In Sec. 5 we saw a case where overparameterized networks generalize better than smaller ones. This was due to the fact that the training set was a "decoy" in the sense that it could be explained by a subset of the discriminative patterns. Due to the under-exploration of weights in the smaller model this led to zero training error but non-zero test error.

We proceed to formulate this intuition. Our theoretical results will show that for diverse training sets, networks with $k \geq 120$ will converge with high probability to a solution with zero training error that recovers f^* (Sec. 6.1). On the other hand, networks with $k = 2$ will converge with constant probability to zero training error solutions which do not recover f^* (Sec. 6.2). Finally, we show that in a PAC setting these results imply a sample complexity gap between large and small networks (Sec. 6.3).

We assume that the training set consists of m positive diverse points and m negative diverse points. For the analysis, without loss of generality, we can assume that the training set consists of one positive diverse point $\boldsymbol{x}^+$ and one negative diverse point $\boldsymbol{x}^-$. This follows since the network and its gradient have the same value for two different positive diverse points and two different negative diverse points. Therefore, this holds for the loss function in Eq. 4 as well.

For the analysis, we need a few more definitions. Define the following sets for each $1 \leq i \leq 4$:

$$W_t^+(i) = \left\{ j \mid \arg\max_{1 \leq l \leq 4} \boldsymbol{w}_t^{(j)} \cdot \boldsymbol{p}_l = i \right\}$$
$$U_t^+(i) = \left\{ j \mid \arg\max_{1 \leq l \leq 4} \boldsymbol{u}_t^{(j)} \cdot \boldsymbol{p}_l = i \right\}$$

For each set of binary patterns $A \subseteq \{\pm 1\}^2$ define p_A to be the probability to sample a point $\boldsymbol{x}$ such that $P_{\boldsymbol{x}} = A$. Let $A_1 = \{2\}$, $A_2 = \{4\}$, $A_3 = \{2, 4, 1\}$ and $A_4 = \{2, 4, 3\}$.

The following quantity will be useful in our analysis:

$$p^* = \min_{1 \leq i \leq 4} p_{A_i} \tag{5}$$

Finally, we let $a^+(t)$ be the number of iterations $0 \leq t' \leq t$ such that $N_{W_{t'}}(\boldsymbol{x}^+) < \gamma$ and $c \leq 10^{-10}$ be a negligible constant.

6.1. Overparameterized Network

As in Sec. 3.2, we will show that both exploration at initialization and clustering will imply good performance of overparameterized networks. Concretely, they will imply convergence to a global minimum that recovers f^*. However, the analysis in XORD is significantly more involved.

We assume that $k \geq 120$ and gradient descent runs with parameters $\eta = \frac{c_\eta}{k}$ where $c_\eta \leq \frac{1}{410}$, $\sigma_g \leq \frac{c_\eta}{16k^{\frac{3}{2}}}$ and $\gamma \geq 8$.

In the analysis there are several instances of exploration and clustering effects. Due to space limitations, here we will show one such instance. In the following lemma we show an example of exploration at initialization. The proof is a direct application of a concentration bound.

Lemma 6.1. ***Exploration.*** *With probability at least* $1 - 4e^{-8}$*, it holds that* $\left| \left| W_0^+(1) \cup W_0^+(3) \right| - \frac{k}{2} \right| \leq 2\sqrt{k}$.

Next, we characterize the dynamics of filters in $W_0^+(1) \cup W_0^+(3)$ for all t.

Lemma 6.2. ***Clustering Dynamics.*** *Let* $i \in \{1, 3\}$*. With probability* $\geq 1 - \frac{\sqrt{2k}}{\sqrt{\pi} e^{8k}}$*, for all* $t \geq 0$ *and* $j \in W_0^+(i)$ *there exists a vector* $\boldsymbol{v}_t$ *such that* $v_t \cdot \boldsymbol{p}_i > 0$*,* $|v_t \cdot \boldsymbol{p}_2| < 2\eta$ *and* $\boldsymbol{w}_t^{(j)} = a^+(t)\eta \boldsymbol{p}_i + \boldsymbol{v}_t$.

We note that $a^+(t)$ is a non-decreasing sequence such that $a^+(0) = 1$ with high probability. Therefore, the above lemma suggests that the weights in $W_0^+(1) \cup W_0^+(3)$ tend to get clustered as t increases.

By combining Lemma 6.1, Lemma 6.2 and other similar lemmas given in the supplementary (for other sets

$W_0^+(i), U_0^+(i)$), the following convergence theorem can be shown. The proof consists of a careful and lengthy analysis of the dynamics of gradient descent and is given in the supplementary.

Theorem 6.3. *With probability at least* $\left(1 - c - 16e^{-8}\right)$ *after running gradient descent for* $T \geq \frac{28(\gamma+1+8c_\eta)}{c_\eta}$ *iterations, it converges to a global minimum which satisfies* $\text{sign}\left(N_{W_T}(\boldsymbol{x})\right) = f^*(\boldsymbol{x})$ *for all* $\boldsymbol{x} \in \{\pm 1\}^{2d}$. *Furthermore, for* $i \in \{1, 3\}$ *and all* $j \in W_0^+(i)$, *the angle between* $\boldsymbol{w}_T^{(j)}$ *and* $\boldsymbol{p}_i$ *is at most* $\arccos\left(\frac{\gamma-1-2c_\eta}{\gamma-1+c_\eta}\right)$. [7]

This result shows if the training set consists only of diverse points, then with high probability over the initialization, overparameterized networks converge to a global minimum which realizes f^* in a constant number of iterations.

6.2. Small Network

Next we consider the case of the small network $k = 2$, and show that it has inferior generalization due to under-exploration. We assume that gradient descent runs with parameter values of η, σ_g and γ which are similar to the previous section but in a slightly broader set of values (see supplementary for details). The main result of this section shows that with constant probability, gradient descent converges to a global minimum that does not recover f^*.

Theorem 6.4. *With probability at least* $(1 - c)\frac{33}{48}$, *gradient descent converges to a global minimum that does not recover* f^*. *Furthermore, there exists* $1 \leq i \leq 4$ *such that the global minimum misclassifies all points* $\boldsymbol{x}$ *such that* $P_{\boldsymbol{x}} = A_i$.

The proof follows due to an *under-exploration* effect. Concretely, let $\boldsymbol{w}_T^{(1)}$, $\boldsymbol{w}_T^{(2)}$, $\boldsymbol{u}_T^{(1)}$ and $\boldsymbol{u}_T^{(2)}$ be the filters of the network at the iteration T in which gradient descent converges to a global minimum (convergence occurs with high constant probability). The proof shows that gradient descent will not learn f^* if one of the following conditions is met: a) $W_T^+(1) = \emptyset$. b) $W_T^+(3) = \emptyset$. c) $\boldsymbol{u}_T^{(1)} \cdot \boldsymbol{p}_2 > 0$ and $\boldsymbol{u}_T^{(2)} \cdot \boldsymbol{p}_2 > 0$. d) $\boldsymbol{u}_T^{(1)} \cdot \boldsymbol{p}_4 > 0$ and $\boldsymbol{u}_T^{(2)} \cdot \boldsymbol{p}_4 > 0$. Then by using a symmetry argument which is based on the symmetry of the initialization and the training data it can be shown that one of the above conditions is met with high constant probability.

6.3. A Sample Complexity Gap

In the previous analysis we assumed that the training set was diverse. Here we consider the standard PAC setting of a distribution over inputs, and show that indeed overparameterized models enjoy better generalization. Recall that the sample complexity $m(\epsilon, \delta)$ of a learning algorithm is the minimal number of samples required for learning a model with test error at most ϵ with confidence greater than $1 - \delta$ (Shalev-Shwartz & Ben-David, 2014).

We are interested in the sample complexity of learning with $k \geq 120$ and $k = 2$. Denote these two functions by $m_1(\epsilon, \delta)$ and $m_2(\epsilon, \delta)$. The following result states that there is a gap between the sample complexity of the two models, where the larger model in fact enjoys better complexity.

Theorem 6.5. *Let* $\mathcal{D}$ *be a distribution with paramaters* p_+, p_- *and* p^* *(see Eq. 5). Let* $\delta \geq 1 - p_+p_-(1 - c - 16e^{-8})$ *and* $0 \leq \epsilon < p^*$. *Then* $m_1(\epsilon, \delta) \leq 2$ *whereas* $m_2(\epsilon, \delta) \geq \frac{2\log\left(\frac{48\delta}{33(1-c)}\right)}{\log(p_+p_-)}$. [8]

The proof (see supplementary material) follows from Theorem 6.3 and Theorem 6.4 and the fact that the probability to sample a training set with only diverse points is $(p_+p_-)^m$.

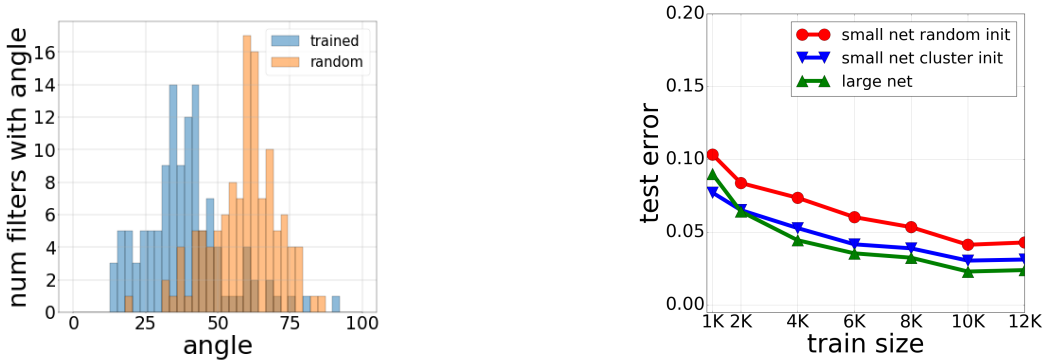
We will illustrate the guarantee of Theorem 6.5 with several numerical examples. Assume that for the distribution $\mathcal{D}$, the probability to sample a positive point is $\frac{1}{2}$ and $p^* = \min\left\{\frac{1-p_+}{4}, \frac{1-p_-}{4}\right\}$ (it is easy to construct such distributions). First, consider the case $p_+ = p_- = 0.98$ and $\delta = 1 - 0.98^2(1 - c - 16e^{-8}) \leq 0.05$. Here we get that for any $0 \leq \epsilon < 0.005$, $m_1(\epsilon, \delta) \leq 2$ whereas $m_2(\epsilon, \delta) \geq 129$. Next, consider the case where $p_+ = p_- = 0.92$. It follows that for $\delta = 0.16$ and any $0 \leq \epsilon < 0.02$ it holds that $m_1(\epsilon, \delta) \leq 2$ and $m_2(\epsilon, \delta) \geq 17$. In contrast, for sufficiently small p_+ and p_- (e.g., when $p_+, p_- \leq 0.7$), our bound does not guarantee a generalization gap.

7. Experiments on MNIST

We next demonstrate how our theoretical insights from the XORD problem are also manifest when learning a neural net on the MNIST dataset. The network we use for learning is quite similar to the one use for XORD. It is a three layer network: the first layer is a convolution with 3×3 filters and multiple channels (we vary the number of channels), followed by 2×2 max pooling and then a fully connected layer. We use Adam (Kingma & Ba, 2014) for optimization. In the supplementary we show empirical results for other filter sizes. Further details of the experiments are given there. Below we show how our two main theoretical insights for XORD are clearly exhibited in the MNIST data.

[7] We do not provide clustering guarantees at global minimum for other filters. However, we do characterize their dynamics similar to Lemma 6.2.

[8] We note that this generalization gap holds for global minima (0 train error). Therefore, the theorem can be read as follows. For $k \geq 120$, given 2 samples, with probability at least $1 - \delta$, gradient descent converges to a global minimum with at most ϵ test error. On the other hand, for $k = 2$ and given number of samples less than $\frac{2\log\left(\frac{48\delta}{33(1-c)}\right)}{\log(p_+p_-)}$, with probability greater than δ, gradient descent converges to a global minimum with error greater than ϵ.



(a) (b)

Figure 4: Clustering and Exploration in MNIST (a) Distribution of angle to closest center in trained and random networks. (b) The plot shows the test error of the small network (4 channels) with standard training (red), the small network that uses clusters from the large network (blue), and the large network (120 channels) with standard training (green). It can be seen that the large network is effectively compressed without losing much accuracy.

We first check the clustering observation. Namely, that optimization tends to converge to clusters of similar filters. We train the three layer network described above with 120 channels on 6000 randomly sampled MNIST images. Then, we normalize each filter of the trained network to have unit norm. We then cluster all 120 9-dimensional vectors using k-means to four clusters. Finally, for each filter we calculate its angle with its closest cluster center. In the second experiment we perform exactly the same procedure, but with a network with randomly initialized weights.

Fig. 4a shows the results for this experiment. It can be clearly seen that in the trained network, most of the 9-dimensional filters have a relatively small angle with their closest center. Furthermore, the distributions of angles to closest center are significantly different in the case of trained and random networks. This suggests that there is an inductive bias towards solutions with clustered weights, as predicted by the theory.

We next explore the effect of exploration. Namely, to what degree do larger models explore useful regions in weight space. The observation in our theoretical analysis is that both small and large networks can find weights that arrive at zero training error. But large networks will find a wider variety of weights, which will also generalize better.

Here we propose to test this via the following setup: first train a large network. Then cluster its weights into k clusters and use the centers to initialize a smaller network with k filters. If these k filters generalize better than k filters learned from random initialization, this would suggest that the larger network indeed explored weight space more effectively.

To apply this idea to MNIST, We trained an "over-parameterized" 3-layer network with 120 channels. We clustered its filters with k-means into 4 clusters and used the cluster centers as initialization for a small network with 4 channels. Then we trained only the fully connected layer and the bias of the first layer in the small network. In Fig. 4b we show that for various training set sizes, the performance of the small network improves with the new initialization and nearly matches the performance of the over-parameterized network. This suggests that the large network explored better features in the convolutional layer than the smaller one.

8. Conclusions

In this paper we consider a simplified learning task on binary vectors to study generalization of overparameterized networks. In this setting, we prove that clustering of weights and exploration of the weight space imply better generalization performance for overparameterized networks. We empirically verify our findings on the MNIST task.

In (Ji & Telgarsky, 2019), it is shown that for a linear network trained with gradient descent, the weight matrices are asymptotically of rank 1. It would be interesting to connect this result with our clustering observation.

For future work, it would be interesting to consider more complex classification tasks such as filters of higher dimension or non-binary data.

Acknowledgements

This research is supported by the Blavatnik Computer Science Research Fund and by the Yandex Initiative in Machine Learning.

References

Allen-Zhu, Z., Li, Y., and Liang, Y. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018.

Arora, S., Cohen, N., Golowich, N., and Hu, W. A convergence analysis of gradient descent for deep linear neural networks. *ICLR*, 2019.

Brutzkus, A. and Globerson, A. Globally optimal gradient descent for a convnet with gaussian inputs. In *International Conference on Machine Learning*, pp. 605–614, 2017.

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

Daniely, A. Sgd learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems*, pp. 2422–2430, 2017.

Du, S. S. and Lee, J. D. On the power of over-parametrization in neural networks with quadratic activation. In *International Conference on Machine Learning*, pp. 1328–1337, 2018.

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

Du, S. S., Lee, J. D., and Tian, Y. When is a convolutional filter easy to learn? *ICLR*, 2018b.

Du, S. S., Lee, J. D., Tian, Y., Singh, A., and Poczos, B. Gradient descent learns one-hidden-layer cnn: Dont be afraid of spurious local minima. In *International Conference on Machine Learning*, pp. 1338–1347, 2018c.

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.

Hoffer, E., Hubara, I., and Soudry, D. Fix your classifier: the marginal value of training the last weight layer. *ICLR*, 2018.

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

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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. and Yuan, Y. Convergence analysis of two-layer neural networks with relu activation. In *Advances in Neural Information Processing Systems*, pp. 597–607, 2017.

Li, Y., Ma, T., and Zhang, H. Algorithmic regularization in over-parameterized matrix sensing and neural networks with quadratic activations. In *Conference On Learning Theory*, pp. 2–47, 2018.

Neyshabur, B., Tomioka, R., and Srebro, N. In search of the real inductive bias: On the role of implicit regularization in deep learning. *arXiv preprint arXiv:1412.6614*, 2014.

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y., and Srebro, N. Towards understanding the role of over-parametrization in generalization of neural networks. *ICLR*, 2019.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study. *ICLR*, 2018.

Shalev-Shwartz, S. and Ben-David, S. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Soltanolkotabi, M., Javanmard, A., and Lee, J. D. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 2018.