An Optimization and Generalization Analysis for Max-Pooling Networks

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

[1]The Blavatnik School of Computer Science, Tel Aviv University

Abstract

Max-Pooling operations are a core component of deep learning architectures. In particular, they are part of most convolutional architectures used in machine vision, since pooling is a natural approach to pattern detection problems. However, these architectures are not well understood from a theoretical perspective. For example, we do not understand when they can be globally optimized, and what is the effect of over-parameterization on generalization. Here we perform a theoretical analysis of a convolutional max-pooling architecture, proving that it can be globally optimized, and can generalize well even for highly over-parameterized models. Our analysis focuses on a data generating distribution inspired by pattern detection problem, where a "discriminative" pattern needs to be detected among "spurious" patterns. We empirically validate that CNNs significantly outperform fully connected networks in our setting, as predicted by our theoretical results.

1 INTRODUCTION

Convolutional neural networks (CNNs) have achieved remarkable performance in various computer vision tasks [Krizhevsky et al., 2012, Xu et al., 2015, Taigman et al., 2014]. Such networks typically combine convolution and max-pooling layers, and can thus be used for detecting complex patterns in the input. In practice, CNNs typically have more parameters than needed to achieve zero train error (i.e., are overparameterized). Despite the potential problem of non-convexity in optimization and overfitting because of overparameterization, training these models with gradient based methods leads to solutions with low test error. Furthermore, overparameterized CNNs significantly outperform fully connected networks (FCNs) on classifying image data [Malach and Shalev-Shwartz, 2020a]. Thus, a key question immediately arises:

Why do overparameterized CNNs generalize well on image data and outperform FCNs?

To the best of our knowledge, this question remains largely unanswered. We note that the question contains two significant challenges: the first is to show that minimization of the non-convex training loss leads to high training accuracy (where non-convexity is a result of both max-pooling and ReLU activations), and the other is that over-fitting is avoided despite over-parameterization. The latter challenge is known as the question of inductive bias of gradient descent [Zhang et al., 2017], and understanding it is a key goal of deep learning theory.

In this work, we provide the first results which address the above question. We theoretically analyze learning a simplified pattern recognition task with overparameterized CNNs and overparameterized FCNs. We consider a CNN with a convolution layer, max pooling and fully connected layer and compare it to a one-hidden layer non-linear FCN. Figure 1 shows an example of our setup. We summarize our contributions as follows:

1. **Expressive Power of CNNs with max-pooling:** We prove a novel VC dimension lower bound in our setting which is exponential in d, the filter dimension of the CNN. This result implies that there exists ERM algorithms which have sample complexity which is exponential in d in our setting.

2. **Optimization and Generalization for learning CNNs with max-pooling:** We analyze learning overparamaterized CNNs with a layerwise gradient descent optimizer. We show that the algorithm converges to zero training loss and the learning has a sample complexity of $O(d)$. This is despite the above VC result, which shows that general ERM optimizers can potentially overfit. In our proof, we analyze the dynamics of training the first layer. We show that it induces a

Accepted for the 37th Conference on Uncertainty in Artificial Intelligence (UAI 2021).

representation in the last layer which is separable with large margin and thus implies a good generalization guarantee.

3. **Generalization of FCNs:** We apply recent results of Brutzkus et al. [2018] which show a generalization bound for overparameterized FC networks that is independent of the network size. We prove that in our setting, their bound can be at best $O(d^{2r})$ for $r \geq 1$, and can thus be much larger than the sample complexity we derive for the CNN.

4. **Empirical Evaluation:** We empirically validate our theoretical results. We show that CNNs generalize well and significantly outperform FCNs in our setting as predicted by our theory. We empirically confirm that this holds also for several extensions of our setup.

Our results make a significant headway on the challenging problem of understanding why overparameterized CNNs can generalize better than overparameterized FCNs on image classification tasks. In particular, to the best of our knowledge, we provide the first optimization and generalization results for overparameterized CNNs with max pooling.

2 RELATED WORK

Two recent works have provided theoretical support that that CNNs outperform FCNs. Li et al. [2020] consider a simplified image classification task and prove a sample complexity gap between FCNs and *single* channel CNNs. Malach and Shalev-Shwartz [2020a] prove that for simplified pattern detection tasks, there is a *computational* separation between overparameterized CNNs and FCNs. Their generalization bound for overparameterized CNNs depends on the number of channels of the CNN. Therefore, both works do not show that over-parameterized CNNs are reslient to over-fitting, which is the main focus of our work.

Several recent works have studied the generalization properties of overparameterized CNNs. Some of these propose generalization bounds that depend on the number of channels [Long and Sedghi, 2020, Jiang et al., 2019]. Others provide guarantees for CNNs with constraints on the weights [Zhou and Feng, 2018, Li et al., 2018]. Convergence of gradient descent to KKT points of the max-margin problem is shown in Lyu and Li [2020] and Nacson et al. [2019] for homogeneous models. However, their results do not provide generalization guarantees in our setting. Gunasekar et al. [2018] study the inductive bias of linear CNNs.

Yu et al. [2019] study a pattern classification problem similar to ours. However, their in analysis the sample complexity guarantee depends on the network size, and thus does not explain why large CNNs do not overfit. Other works have studied learning under certain ground truth distributions. For example, Brutzkus and Globerson [2019] study a simple extension of the XOR problem, showing that overparameterized CNNs generalize better than smaller CNNs. Single-channel CNNs are analyzed in [Du et al., 2018b,a, Brutzkus and Globerson, 2017, Du et al., 2018c]. CNNs were analyzed via the NTK approximation [Li et al., 2019, Arora et al., 2019c]. Our analysis does not assume the NTK approximation. For example, we require a mild overparameterization in our results which does not depend on the number of samples, in contrast to NTK analyses. Furthermore, our results hold for sufficiently small initialization, which is not the regime of NTK analysis.

Other works study the inductive bias of gradient descent on fully connected linear or non-linear networks [Ji and Telgarsky, 2019a, Arora et al., 2019a, Wei et al., 2019, Brutzkus et al., 2018, Dziugaite and Roy, 2017, Allen-Zhu et al., 2019, Chizat and Bach, 2020]. Fully connected networks were also analyzed via the NTK approximation [Du et al., 2019, 2018d, Arora et al., 2019b, Fiat et al., 2019]. Kushilevitz and Roth [1996], Shvaytser [1990] study the learnability of visual patterns distribution. However, our focus is on learnability using a specific algorithm and architecture: gradient descent trained on overparameterized CNNs.

3 PRELIMINARIES

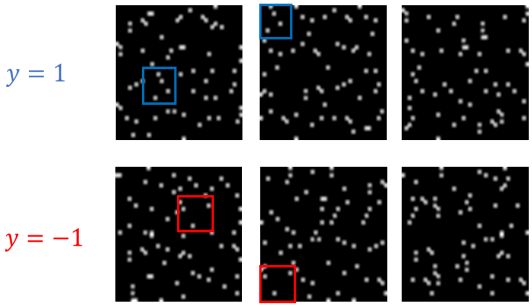
Data Generating Distribution: We consider a learning problem that captures a key property of visual classification. Many visual classes are characterized by the existence of certain patterns. For example an 8 will typically contain an x-like pattern somewhere in the image. Here we consider an abstraction of this behavior where images consist of a set of patterns. Furthermore, each class is characterized by a pattern that appear exclusively in it. We define this formally below.

Let $\mathcal{O}$ be a set of $3 \leq l \leq d$ orthogonal vectors in $\mathbb{R}^d$. For simplicity, we assume that $\|\boldsymbol{o}\|_2 = 1$ for all $\boldsymbol{o} \in \mathcal{O}$. We denote $\mathcal{O} = \{\boldsymbol{o}_1, \boldsymbol{o}_2, ..., \boldsymbol{o}_l\}$ and refer to vectors $\boldsymbol{o}_i$ as patterns. For convenience, we will also refer to sets of patterns as sets of their corresponding indices. For example, for a set of patterns $\mathcal{A}$ we use the notation $i \in \mathcal{A}$ to denote $\boldsymbol{o}_i \in \mathcal{A}$.

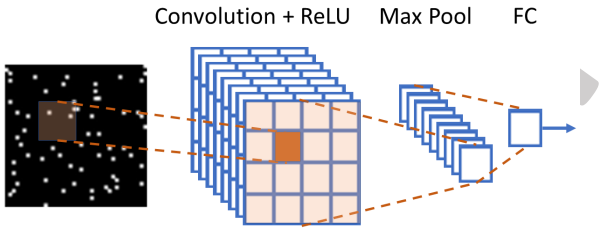
We consider input vectors $\boldsymbol{x}$ with n patterns. Formally, $\boldsymbol{x} = (\boldsymbol{x}[1], ..., \boldsymbol{x}[n]) \in \mathbb{R}^{nd}$ where $\boldsymbol{x}[i] \in \mathbb{R}^d$ is the ith pattern of $\boldsymbol{x}$.[1] We say that $\boldsymbol{x}$ contains $\boldsymbol{p}$ if there exists j such that $\boldsymbol{x}[j] = \boldsymbol{p}$. We denote $\boldsymbol{p} \in \boldsymbol{x}$ if $\boldsymbol{x}$ contains the pattern $\boldsymbol{p} \in \mathbb{R}^d$. Let $\mathcal{P}(\boldsymbol{x}) = \left\{\boldsymbol{p} \in \boldsymbol{x} \mid \boldsymbol{p} \in \mathbb{R}^d\right\}$ denote the set of all patterns in $\boldsymbol{x}$.

Next, we define how labeled points are generated. In our setting we consider three types of patterns: positive, negative and spurious. We will refer to the pattern $\boldsymbol{o}_1$ as positive, the pattern $\boldsymbol{o}_2$ as negative and the patterns $\boldsymbol{o}_3, \ldots, \boldsymbol{o}_l$ as

[1] We will generally use the notation $\boldsymbol{v}[i] \in \mathbb{R}^d$ for any vector $\boldsymbol{v} \in \mathbb{R}^{nd}$.



(a)



(b)

Figure 1: (a) An example of the pattern detection tasks we consider. Input images consist of 4 rows of 4 patches each. Each image consists of a discriminative pattern. All other patterns are spurious and may appear in both classes. In the two leftmost images of each class, the corresponding discriminative pattern is shown. (b) An illustration of the architecture of the 3-layer overparameterized CNN we analyze in our setting.

spurious. We let $\mathcal{S} = \{3, ..., l\}$.

We consider distributions $\mathcal{D}$ over $(\boldsymbol{x}, y) \in \mathbb{R}^{nd} \times \{\pm 1\}$. In the distribution $\mathcal{D}$, each positive sample contains the positive pattern and $n-1$ randomly sampled spurious patterns. Similarly, a negative sample has a single negative pattern and $n-1$ spurious patterns. Formally, we define $\mathcal{D}$ with the following properties:

1. $\mathbb{P}(y=1) = \mathbb{P}(y=-1) = \frac{1}{2}$.
2. Given $y=1$, a vector $\boldsymbol{x}$ is sampled as follows. Randomly sample an index $1 \le j_+ \le n$ for placing the positive pattern, and set $\boldsymbol{x}[j_+] = \boldsymbol{o}_1$. Then, for each $1 \le j \le n$ such that $j \ne j_+$, randomly choose $i_j \in \mathcal{S}$ and set $\boldsymbol{x}[j] = \boldsymbol{o}_{i_j}$.
3. Given $y=-1$, do the same as $y=1$, using $\boldsymbol{o}_2$ instead of $\boldsymbol{o}_1$.

Fig. 1a shows an example of the above distribution $\mathcal{D}$.

CNN Architecture: For learning the above distributions, we consider a 3-layer CNN that consists of a convolutional layer with non-overlapping filters, followed by ReLU, max pooling and a fully-connected layer. The network is parametrized by $\boldsymbol{\theta} = (W, \boldsymbol{a})$ where $W \in \mathbb{R}^{k \times n}$ and each row i of W, denoted by $\boldsymbol{w}_i \in \mathbb{R}^d$, corresponds to a different channel. The vector $\boldsymbol{a} = (a_1, ..., a_k) \in \mathbb{R}^k$ corresponds to the weights of the fully connected layer.

For an input $\boldsymbol{x} = (\boldsymbol{x}[1], ..., \boldsymbol{x}[n]) \in \mathbb{R}^{nd}$ where $\boldsymbol{x}[i] \in \mathbb{R}^d$, the output of the network is:

$$N_{\text{CNN}}(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{i=1}^{k} a_i \Big[\max_j \{\sigma(\boldsymbol{w}_i \cdot \boldsymbol{x}[j])\} \Big] \qquad (1)$$

where $\sigma(x) = \max\{0, x\}$ is the ReLU activation. For simplicity, we will usually denote $N_{\text{CNN}}(\boldsymbol{x})$ when $\boldsymbol{\theta}$ is clear from the context. We define $\mathcal{H}_{\text{CNN}}(\mathcal{X})$ to be the hypothesis class of all functions $\text{sign}(N_{\text{CNN}}) : \mathcal{X} \to \{\pm 1\}$, where $\mathcal{X} \subseteq \mathbb{R}^{nd}$.[2]

CNN Training Algorithm: For the analysis of learning CNNs, we will consider a layerwise optimization algorithm which performs gradient updates layer-by-layer, starting from the first layer. Layerwise optimization algorithms are used in practice and have been shown to achieve performance that is comparable to end-to-end methods, e.g., on ImageNet [Belilovsky et al., 2019]. Furthermore, the assumption on layerwise optimization has been used previously for theoretically analyzing neural networks [Malach and Shalev-Shwartz, 2020b].

For a set of points $A \subseteq \mathbb{R}^{nd} \times \{\pm 1\}$ we consider minimizing the loss:

$$\mathcal{L}[A](\boldsymbol{\theta}) = \frac{1}{|A|} \sum_{(\boldsymbol{x}, y) \in A} \ell\left(y N_{\text{CNN}}((;\boldsymbol{\theta})\, \boldsymbol{x})\right) \qquad (2)$$

where $\ell(x) = \log(1 + e^{-x})$ is the binary cross entropy loss. Let $S = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_m, y_m)\}$ be a training set with m IID samples from $\mathcal{D}$. For the analysis, we partition $S = S_1 \cup S_2$ to two disjoint sets S_1 and S_2 such that $S_1 = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_{\lceil \frac{m}{2} \rceil}, y_{\lceil \frac{m}{2} \rceil})\}$. We denote $\mathcal{L}_i = \mathcal{L}[S_i]$, and $m_i = |S_i|$ for $i \in \{1, 2\}$. For convenience, we will say that $\boldsymbol{x} \in S_i$ if there exists $y \in \{\pm 1\}$ such that $(\boldsymbol{x}, y) \in S_i$. We denote the set of positive samples in S_i by $S_i^+ = \{\boldsymbol{x} \mid (\boldsymbol{x}, 1) \in S_i\}$ and the negative samples in S_i by $S_i^- = \{\boldsymbol{x} \mid (\boldsymbol{x}, -1) \in S_i\}$.

The layerwise optimization algorithm for learning CNNs is given in Figure 2. The reason we optimize over two losses is technical: we need a fresh IID sample (S_2) in the sec-

[2] We assume WLOG that $\text{sign}(0) = -1$. Furthermore, we note that the network N_{CNN} can have any number of channels k.

Algorithm 1 LW_{CNN}

Input: Training set $S \subseteq \mathbb{R}^{nd} \times \{\pm 1\}$, numbers of iterations $T_1, T_2 \in \mathbb{N}$ and learning rates $\eta_1, \eta_2 \in \mathbb{R}$.
Initialize $W^{(0)}$ and $a^{(0)}$.
for $t = 1, ..., T_1$ **do**:
$W^{(t)} \leftarrow W^{(t-1)} - \eta_1 \frac{\partial \mathcal{L}_1}{\partial W}\left(W^{(t-1)}, \boldsymbol{a}^{(0)}\right)$.
for $t = 1, ..., T_2$ **do**:
$\boldsymbol{a}^{(t)} \leftarrow \boldsymbol{a}^{(t-1)} - \eta_2 \frac{\partial \mathcal{L}_2}{\partial \boldsymbol{a}}\left(W^{(T_1)}, \boldsymbol{a}^{(t-1)}\right)$.
return $\left(W^{(T_1)}, \boldsymbol{a}^{(T_2)}\right)$.

Figure 2: Layerwise optimization algorithm for CNNs.

ond layer optimization for the generalization analysis (see Section 5).

We define $\boldsymbol{w}_i^{(t)}$ to be the ith row of $W^{(t)}$. For $\boldsymbol{x} \in S$, $t > 0$ and $1 \leq i \leq k$, define $j_i^{(t)}(\boldsymbol{x}) = \arg\max_{1 \leq j \leq n} \boldsymbol{w}_i^{(t)} \cdot \boldsymbol{x}[j]$, i.e., $j_i^{(t)}(\boldsymbol{x})$ corresponds to the pattern in $\boldsymbol{x}$ that maximally activates $\boldsymbol{w}_i^{(t)}$. If $\boldsymbol{w}_i^{(t)} \cdot \boldsymbol{x}\left[j_i^{(t)}(\boldsymbol{x})\right] > 0$, define $\boldsymbol{p}_i^{(t)}(\boldsymbol{x}) = \boldsymbol{x}\left[j_i^{(t)}(\boldsymbol{x})\right]$. Otherwise, define $\boldsymbol{p}_i^{(t)}(\boldsymbol{x}) = 0$. Notice that the following equality holds:

$$\max_j \left\{\sigma\left(\boldsymbol{w}_i^{(t)} \cdot \boldsymbol{x}[j]\right)\right\} = \boldsymbol{w}_i^{(t)} \cdot \boldsymbol{p}_i^{(t)}(\boldsymbol{x}) \tag{3}$$

Remark 3.1. *We note that it is necessary to make assumptions regarding the data distribution because the general case is intractable for optimization (because it includes neural net learning as a special case). We believe that our data generating distribution does reflect core aspects of pattern detection problems. Furthermore, the analysis of overparameterized max pooling networks has not been performed for any task, and analysis of simplified tasks has been shown to be fruitful for understanding CNNs [Li et al., 2020, Malach and Shalev-Shwartz, 2020a]. Additionally, non-overlapping filters are used in practice, and multiple theoretical works have analyzed CNNs with non-overlapping filters due to their tractability [Sharir and Shashua, 2018]. Finally, we note that in Section 7 we show that our analysis is in line with the performance of CNNs and FCNs in more complex tasks.*

4 VC DIMENSION BOUND

Thus far we described a data generating distribution and a neural architecture. We now ask how expressive is this neural architecture. Because of the pooling layer, it may seem that the network has limited capacity, even for an unbounded number of channels. However, as we show next the capacity in terms of VC dimension is in fact exponential in d in this case. This in turn means that the network can separate datasets of size up to exponential in d, and can thus potentially overfit badly. As we show in later sections, overfitting is avoided when learning using gradient descent.

Fix $\mathcal{X} \subseteq \mathbb{R}^{nd}$ to be the support of the distribution $\mathcal{D}$, i.e., each input vector consist of either a positive or negative pattern and $n-1$ spurious patterns. Denote the VC dimension of $\mathcal{H}_{\text{CNN}}(\mathcal{X})$ by $\text{VCdim}\left(\mathcal{H}_{\text{CNN}}(\mathcal{X})\right)$. If we find $\text{VCdim}\left(\mathcal{H}_{\text{CNN}}(\mathcal{X})\right)$, then we can apply generalization bounds which show that any Empirical Minimization algorithm (ERM) has sample complexity of $O\left(\text{VCdim}\left(\mathcal{H}_{\text{CNN}}(\mathcal{X})\right)\right)$ [Blumer et al., 1989], and there exists an ERM with a tight lower bound.[3] Thus, lower bounding the VC dimension leads to a worst-case lower bound on sample complexity,

We begin by recalling the definition of the VC dimension.

Definition 4.1. *Let $\mathcal{H}$ be a hypothesis class of functions from $\mathcal{X}$ to $\{\pm 1\}$. For any non-negative integer m, define:*

$$\Pi_{\mathcal{H}}(m) = \max_{x_1,...,x_m \in \mathcal{X}} \left|\{(h(x_1), ..., h(x_m)) \mid h \in \mathcal{H}\}\right| \tag{4}$$

If $\left|\{(h(x_1), ..., h(x_m)) \mid h \in \mathcal{H}\}\right| = 2^m$, we say that $\mathcal{H}$ shatters the set $\{x_1, ..., x_m\}$. The VC dimension of $\mathcal{H}$, denoted by $\text{VCdim}(\mathcal{H})$, is the size of the largest shattered set, or equivalently, the largest m such that $\Pi_{\mathcal{H}}(m) = 2^m$.

In the next theorem we show that $\text{VCdim}\left(\mathcal{H}_{\text{CNN}}(\mathcal{X})\right)$ is at least exponential in d. Therefore, the best generalization bound we can hope for using a VC dimension analysis scales exponentially with d.[4]

Theorem 4.2. *Assume that $d = 2n$ and $n \geq 2$. Then $\text{VCdim}\left(\mathcal{H}_{\text{CNN}}(\mathcal{X})\right) \geq 2^{\frac{d}{2}-1}$.*

Proof. We will construct a set $B \subseteq \mathcal{X}$ of size $2^{n-1} = 2^{\frac{d}{2}-1}$ that can be shattered. For a given $I \in \{0,1\}^{n-1}$ let $I[j]$ be its jth entry. For any such I, define a point $\boldsymbol{x}_I$ such that for any $1 \leq j \leq n-1$, $\boldsymbol{x}_I[j] = I[j]\boldsymbol{o}_{2j+1} + (1 - I[j])\boldsymbol{o}_{2j+2}$. Furthermore, arbitrarily choose $\boldsymbol{x}_I[n] = \boldsymbol{o}_1$ or $\boldsymbol{x}_I[n] = \boldsymbol{o}_2$ and define $B = \left\{\boldsymbol{x}_I \mid I \in \{0,1\}^{n-1}\right\}$.

Now, assume that each point $\boldsymbol{x}_I \in B$ has label y_I. We will show that there is a network $N_{\text{CNN}} \in \mathcal{H}_{\text{CNN}}(\mathcal{X})$ such that $N_{\text{CNN}}(\boldsymbol{x}_I) = y_I$ for all I. For each $I \in \{0,1\}^{n-1}$, define $\boldsymbol{w}^{(I)} = \max\{\alpha_I, 0\} \sum_{1 \leq j \leq n-1} \boldsymbol{x}_I[j]$ and $\boldsymbol{u}^{(I)} = \max\{-\alpha_I, 0\} \sum_{1 \leq j \leq n-1} \boldsymbol{x}_I[j]$, where $\{\alpha_I\}$ is the unique solution of the following linear system with 2^{n-1} equations.

[3] Recall that an ERM algorithm is any algorithm which minimizes the empirical risk. See Shalev-Shwartz and Ben-David [2014] for details.

[4] By fixing $\mathcal{X}$ to be the support of $\mathcal{D}$ we get a more accurate VC lower bound than the case where $\mathcal{X} = \mathbb{R}^{nd}$. This is because in the case where $\mathcal{X} = \mathbb{R}^{nd}$, shattered sets that are impossible to sample from $\mathcal{D}$ may be considered in the lower bound.

For each $I \in \{0,1\}^{n-1}$ the system has the following equation:

$$\sum_{I' \in \{0,1\}^{n-1} \setminus \{I\}} \alpha_{I'} = y_{I^c} \tag{5}$$

where for any $I \in \{0,1\}^{n-1}$, $I^c \in \{0,1\}^{n-1}$ is defined such that $I^c[j] = 1 - I[j]$ for all $1 \le j \le n-1$. There is a unique solution because the corresponding matrix of the linear system is the difference between an all 1's matrix and the identity matrix. By the Sherman-Morrison formula [Sherman and Morrison, 1950], this matrix is invertible, where in the formula the outer product rank-1 matrix is the all 1's matrix and the invertible matrix is minus the identity matrix.

Set W to be the matrix with rows $\boldsymbol{w}^{(I)}$ followed by rows $\boldsymbol{u}^{(I)}$. Let $\boldsymbol{a}$ be the a vector of dimension 2^n such that $\boldsymbol{a} = (\underbrace{1,...,1}_{2^{n-1}}, \underbrace{-1,...,-1}_{2^{n-1}})$.

Then, for N_{CNN} with parameters $\boldsymbol{\theta} = (W, \boldsymbol{a})$ and any $\boldsymbol{x}_I$:

$$\begin{aligned}
&N_{\text{CNN}}(\boldsymbol{x}_I; \boldsymbol{\theta}) = \sum_{I' \in \{0,1\}^{n-1}} \Big[\max_j \left\{ \sigma\left(\boldsymbol{w}^{(I')} \cdot \boldsymbol{x}[j] \right) \right\} \\
&- \max_j \left\{ \sigma\left(\boldsymbol{u}^{(I')} \cdot \boldsymbol{x}[j] \right) \right\} \Big] \\
&= \sum_{I' \in \{0,1\}^{n-1}} \alpha_{I'} \max_j \left\{ \sigma\left(\sum_{1 \le i \le n-1} \boldsymbol{x}_{I'}[i] \cdot \boldsymbol{x}_I[j] \right) \right\} \\
&= \sum_{I' \in \{0,1\}^{n-1} \setminus \{I^c\}} \alpha_{I'} = y_I
\end{aligned}$$

by the definition of N_{CNN}, the orthogonality of the patterns $\{\boldsymbol{o}_i\}_i$, and Eq. 5. We have shown that any labeling y_I can be achieved, and hence the set is shattered, completing the proof. □

The main limitation of the VC analysis is that it does not take into account the specific implementation of the ERM algorithm [Shalev-Shwartz and Ben-David, 2014]. In the next section, we will show a more fine-grained analysis which is specific to the layerwise optimization algorithm, and can thus benefit from the specific inductive bias of this algorithm. As a result, we will obtain a significantly better generalization guarantee.

5 GENERALIZATION ANALYSIS OF GRADIENT DESCENT

In this section we analyze the optimization and generalization performance of the layer-wise gradient descent algorithm LW_{CNN} for training overparameterized CNNs (Eq. 1). We will show that it converges to zero training loss and its sample complexity is $O(d)$. This is in contrast to the result of the previous section which shows a VC dimension lower bound which is exponential in d, and therefore there are other ERM algorithms that can result in arbitrarily bad test error.

For simplicity of the analysis, we assume that we initialize each filter $\boldsymbol{w}_i^{(0)}$ from the $(d-1)$-sphere of radius r, namely, $\left\{ \boldsymbol{z} \in \mathbb{R}^d \mid \|\boldsymbol{z}\| = r \right\}$. We sample each $a_i^{(0)} \in \mathbb{R}$ uniformly at random from $\{\pm 1\}$. Additionally, the parameters $W^{(0)}$ and $\boldsymbol{a}^{(0)}$ are sampled independently. Our main result is summarized in the following theorem.

Theorem 5.1. *Let S be an IID training set of size m sampled from $\mathcal{D}$. Assume that we run LW_{CNN} with $T_1 > 0$, $\eta_1 \le \frac{1}{4k(T_1+1)}$ and $\eta_2 < 8k$. Assume that $r \le \frac{\eta_1}{200}$ and $k > 8d^3$. Then, with probability at least $(1-\delta)(1 - 4e^{-d} - 4e^{-\frac{m}{36}})$, the following holds:*[5]
(1) $\lim_{T_2 \to \infty} \mathcal{L}_2\left(\left(W^{(T_1)}, \boldsymbol{a}^{(T_2)} \right) \right) = 0$.
(2) $\lim_{T_2 \to \infty} \mathbb{P}_{(\boldsymbol{x},y) \sim \mathcal{D}} \left(\text{sign}\left(N_{\text{CNN}}\left(\boldsymbol{x}; \left(W^{(T_1)}, \boldsymbol{a}^{(T_2)} \right) \right) \right) \neq y \right) = O\left(\sqrt{\frac{d}{m}} \right)$.[6]

The first part of the theorem is an optimization result stating that the LW_{CNN} will converge to zero $\mathcal{L}_2$ loss. We note that this is despite the non-convexity of the loss $\mathcal{L}_2$. The second part of the theorem states that the learned classifier will have a test error of order $\sqrt{\frac{d}{m}}$. Thus, the sample complexity is *linear* in d. This is in contrast to the VC dimension bound which is exponential in d.

Before proving the theorem, we make several remarks on the result. First, for simplicity we present asymptotic results for T_2. We can provide convergence rates that depend linearly on d by changing the second layer optimization hyper-parameters (initialization and step size) and use recent results of Ji and Telgarsky [2019c]. See supplementary for details. Second, note that $k > 8d^3$ is a mild overparameterization condition, compared to other results which require k to depend on the number of samples [Du et al., 2018d, Ji and Telgarsky, 2019b].

Proof of Theorem 5.1. We will prove the theorem in three parts. We defer the proofs of technical lemmas to the supplementary. We first outline the main ideas of the proof. In the first part we will prove a property of the initialization of the first layer. We show that at initialization there are sufficiently many "lucky" filters $\boldsymbol{w}_i^{(0)}$ in the following sense. Either the pattern in $\mathcal{O}$ that maximally activates them is $\boldsymbol{o}_1$ and $a_i^{(0)} = 1$, or the maximum activating pattern is $\boldsymbol{o}_2$ and $a_i^{(0)} = -1$. In essence, these filters are "good" detectors be-

[5] The factor e^{-d} in the confidence guarantee can be improved to $e^{-\Theta(k)}$. Note that the algorithm can be boosted with multiple restarts. We note also that $O(\cdot)$ hides a dependence on δ.

[6] The O hides an additive term which depends on δ.

cause they detect the discriminative patterns, with the right sign of $a_i^{(0)}$.

In the second part we analyze the dynamics of the filters in the first layer. We will show that the "lucky" filters continue to detect the discriminative patterns and their projection on either $\boldsymbol{o}_1$ or $\boldsymbol{o}_2$ becomes larger in each iteration. In contrast, we upper bound the norm of the filters that are "non-lucky". Thus, after training the first layer, LW_{CNN} creates a new representation of the data in the second layer with the following properties: there are sufficiently many discriminative features with sufficiently large absolute values, and the remaining features have a bounded absolute value.

In the third part, we analyze the optimization of the second layer on the new representation. Using the properties of the representation, proved in the second part, we show that this representation induces a distribution on the samples which is linearly separable. Furthermore, it can be classified with margin 1 by a linear classifier of low norm. Then, we apply a result of Soudry et al. [2018], which implies that training the second layer, which is equivalent to logistic regression on the new representation, converges to a low norm solution with zero training loss. Finally, we apply a norm-based generalization bound [Shalev-Shwartz and Ben-David, 2014] to obtain the sample complexity guarantee.

Part 1: Properties of the Initialization:

Define the sets $\mathcal{A}^+ = \left\{i \mid a_i^{(0)} = 1\right\}$, $\mathcal{A}^- = \left\{i \mid a_i^{(0)} = -1\right\}$ and the following sets:

$$\mathcal{W}_t^+ = \left\{i \mid \arg\max_{l \in \mathcal{O} \setminus \{2\}} \boldsymbol{w}_i^{(t)} \cdot \boldsymbol{o}_l = 1, \boldsymbol{w}_i^{(t)} \cdot \boldsymbol{o}_1 > 0\right\}$$
$$\mathcal{W}_t^- = \left\{i \mid \arg\max_{l \in \mathcal{O} \setminus \{1\}} \boldsymbol{w}_i^{(t)} \cdot \boldsymbol{o}_l = 2, \boldsymbol{w}_i^{(t)} \cdot \boldsymbol{o}_2 > 0\right\} \quad (6)$$

The sets $\mathcal{W}_0^+ \cap \mathcal{A}^+$ and $\mathcal{W}_0^- \cap \mathcal{A}^-$ correspond to the set of "lucky" filters.[7] We prove a lower and upper bound on the size of these sets.

Lemma 5.2. *With probability at least* $1 - 4e^{-d}$*:*

$$\frac{k}{4d} \leq |\mathcal{W}_0^+ \cap \mathcal{A}^+|, |\mathcal{W}_0^- \cap \mathcal{A}^-| \leq \frac{k}{d} \quad (7)$$

The proof uses the fact that $\mathbb{P}\left(i \in \mathcal{W}_0^+ \cap \mathcal{A}^+\right) = \frac{\left(1-2^{-d+1}\right)}{2(d-1)}$. Then, by concentration of measure for $k \geq d^3$, roughly $\frac{k}{2d}$ filters will be in $\mathcal{W}_0^+ \cap \mathcal{A}^+$. The same argument holds for $\mathcal{W}_0^- \cap \mathcal{A}^-$. The proof is given in the supplementary.

[7] We exclude the pattern $\boldsymbol{o}_2$ from the argmax in $\mathcal{W}_t^+$ because we only need to consider the argmax over patterns that appear in positive points. Recall that the pattern $\boldsymbol{o}_2$ does not appear in positive points. The same reasoning applies for $\mathcal{W}_t^-$.

Part 2: First Layer Dynamics:

The following lemma shows the dynamics of the "lucky" neurons that detect the positive patterns.

Lemma 5.3. *With probability at least* $1 - 4e^{-d} - 4e^{-\frac{m}{36}}$*, for all* $0 \leq t \leq T_1$ *and all* $i \in \mathcal{W}_0^+ \cap \mathcal{A}^+$ *the following holds:*

1. $\boldsymbol{o}_1 \cdot \boldsymbol{w}_i^{(t)} \geq \frac{t\eta_1}{9}$.
2. *For all* $j \neq 1$*, it holds that* $\boldsymbol{o}_j \cdot \boldsymbol{w}_i^{(t)} \leq r$.

Furthermore, for all $\boldsymbol{x}_+ \in S_1^+$, $\boldsymbol{p}_i^{(t)}(\boldsymbol{x}_+) = \boldsymbol{o}_1$.

The lemma shows that the projection of the filter on $\boldsymbol{o}_1$ grows significantly, while the projection on other $\boldsymbol{o}_i$ remains small. Finally, it shows that for any positive point in S_1, the pattern which maximally activates the filter is $\boldsymbol{o}_1$. Thus, the filter is correctly detecting the positive pattern. The proof is technical and shows that the properties above hold by induction on t. It is given in the supplementary.

By the symmetry of our setting we get by Lemma 5.3 a similar result for the "lucky" neurons that detect negative patterns.

Corollary 5.4. *With probability at least* $1 - 4e^{-d} - 4e^{-\frac{m}{36}}$*, for all* $0 \leq t \leq T_1$ *and all* $i \in \mathcal{W}_0^- \cap \mathcal{A}^-$ *the following holds:*

1. $\boldsymbol{o}_2 \cdot \boldsymbol{w}_i^{(t)} \geq \frac{t\eta_1}{9}$.
2. *For all* $j \neq 2$*, it holds that* $\boldsymbol{o}_j \cdot \boldsymbol{w}_i^{(t)} \leq r$.

Furthermore, for all $\boldsymbol{x}_- \in S_1^-$, $\boldsymbol{p}_i^{(t)}(\boldsymbol{x}_-) = \boldsymbol{o}_2$.

Finally, we provide a simple bound on the output of all neurons (including the "non-lucky" ones).

Lemma 5.5. *For all* $1 \leq t \leq T_1$*,* $1 \leq i \leq k$*,* $1 \leq j \leq d$ *and* $\boldsymbol{x}$ *sampled from* $\mathcal{D}$*, it holds that* $\boldsymbol{x}[j] \cdot \boldsymbol{w}_i^{(t)} \leq 2\eta_1 t$.

The proof is given in the supplementary.

Part 3: Optimizing the Second Layer:

We conclude the proof of the theorem by analyzing the optimization of the second layer. Here we sketch the analysis and defer the details to the supplementary.

For each $\boldsymbol{x}$ sampled from $\mathcal{D}$, we define $\boldsymbol{z}(\boldsymbol{x}) \in \mathbb{R}^k$ such that for all $1 \leq i \leq k$, its ith entry is $z_i(\boldsymbol{x}) = \max_j \left\{\sigma\left(\boldsymbol{w}_i^{(T_1)} \cdot \boldsymbol{x}[j]\right)\right\}$ (namely, these are the values of the output of the pooling of each channel, which serve as features for the second layer). Then, we define a new distribution of points $\mathcal{D}_{\boldsymbol{z}}$ over $\mathbb{R}^k \times \{\pm 1\}$, which samples a point $(\boldsymbol{z}(\boldsymbol{x}), y)$ where $(\boldsymbol{x}, y) \sim \mathcal{D}$.

Using the results of the first layer dynamics, we show that $\mathcal{D}_{\boldsymbol{z}}$ is linearly separable and can be separated with margin 1 by a classifier $\boldsymbol{v}$ with $\|\boldsymbol{v}\| = O\left(\sqrt{\frac{d}{k}}\right)$. Then, we use recent

results on logistic regression [Soudry et al., 2018], to show that by optimizing the second layer, LW_{CNN} will converge to a low norm solution with zero training loss. Finally, we apply norm-based generalization bounds [Shalev-Shwartz and Ben-David, 2014]. Since for all $\boldsymbol{x}$, $\|\boldsymbol{z}(\boldsymbol{x})\| = O(\sqrt{k})$, we obtain a sample complexity guarantee for LW_{CNN} of order $O\left(\|\boldsymbol{v}\|^2 \max_{\boldsymbol{x}} \|\boldsymbol{z}(\boldsymbol{x})\|^2\right) = O(d)$. □

6 COMPARISON WITH FCNS

In the previous section we showed that overparameterized CNNs have good sample complexity for learning the pattern distributions $\mathcal{D}$ in Section 3. How do overparameterized fully connected networks compare with CNNs in our setting? To address this question, we apply recent results of Brutzkus et al. [2018]. They provide generalization guarantees for one-hidden layer overparameterized fully connected networks on linearly separable data. We will show that their bound for FC networks can be $O(d^{2r})$ for any $r \geq 1$. In contrast, Theorem 5.1 shows a generalization bound for CNNs which is linear in d. We note that to fully demonstrate a gap between the methods we also need a lower bound on the FCN for the distribution $\mathcal{D}$, and we leave this for future work. Nonetheless, we show empirically that these generalization bounds predict the performance gap between CNNs and FCNs in our setting.

We begin by noting that the distribution $\mathcal{D}$ is linearly separable in $\boldsymbol{x}$, because one can set $\boldsymbol{w} \in \mathbb{R}^{nd}$ to be a concatenation of n copies of the pattern difference $\boldsymbol{o}_1 - \boldsymbol{o}_2$ and because of orthogonality this will correctly classify the data. We next explain how Brutzkus et al. [2018] can be used to obtain a sample complexity bound for learning this data with a fully connected leaky ReLU net.

Assume that $\mathcal{D}$ is linearly separable with margin 1 by a classifier $\boldsymbol{w}^*$, i.e., for all $(\boldsymbol{x}, y) \in \mathcal{D}$, $y\boldsymbol{w}^* \cdot \boldsymbol{x} \geq 1$. Brutzkus et al. [2018] consider the following fully connected network:

$$N_{\text{FC}}(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{i=1}^{k} a_i \psi(\boldsymbol{w}_i \cdot \boldsymbol{x}) \tag{8}$$

for $\boldsymbol{\theta} = (W, \boldsymbol{a})$ where in our setting $\boldsymbol{w}_i \in \mathbb{R}^{nd}$ is the ith row of $W \in \mathbb{R}^{k \times nd}$, $\boldsymbol{a} \in \mathbb{R}^k$ and $\boldsymbol{x} \in \mathbb{R}^{nd}$ and $\psi(x) = \max\{\alpha x, x\}$ is the Leaky ReLU activation.

They show that SGD converges to a zero training error solution with sample complexity of $O(\|\boldsymbol{w}^*\|^2 R^2)$, where R is the maximum norm of the data, $R = \max_{\boldsymbol{x}} \|\boldsymbol{x}\|$. In our setting it holds that $R^2 = n$ (because each point $\boldsymbol{x}$ consists of n patterns, each of norm 1). Importantly, this bound is independent of the network size k.

We note that the bound $O(\|\boldsymbol{w}^*\|^2 R^2)$ also holds for the hard-margin linear SVM [Shalev-Shwartz and Ben-David, 2014]. Therefore, our following conclusions hold for this algorithm as well. In the next section we show experiments that compare CNNs, FCNs and SVMs in our setting and corroborate our findings.

The generalization bound of $O(\|\boldsymbol{w}^*\|^2 R^2)$ holds for *any* $\boldsymbol{w}^*$ which separates with margin 1. Thus, the best bound can be achieved with $\boldsymbol{w}^*$ that has the lowest norm and separates the data with margin 1. Next we show that the lowest norm is at least $\sqrt{n}$.

Proposition 6.1. *Assume that:*

$$\hat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w} \in \mathbb{R}^{nd}} \|\boldsymbol{w}\|^2 \;\; s.t. \;\; \forall (\boldsymbol{x}, y) \sim \mathcal{D} \;\; y\boldsymbol{w} \cdot \boldsymbol{x} \geq 1 \tag{9}$$

Then $\|\hat{\boldsymbol{w}}\|^2 \geq n$.

Proof. Assume by contradiction that $\|\hat{\boldsymbol{w}}\|^2 = \sum_{1 \leq i \leq n} \|\hat{\boldsymbol{w}}[i]\|^2 < n$. Then, there exists $1 \leq i \leq n$ such that $\|\hat{\boldsymbol{w}}[i]\|^2 < 1$. Define a positive point $(\boldsymbol{x}_+, 1)$ such that $\boldsymbol{x}_+[i] = \boldsymbol{o}_1$ and $\boldsymbol{x}_+[j] = \boldsymbol{o}_3$ for $j \neq i$. Similarly, define a negative point $(\boldsymbol{x}_-, -1)$ such that $\boldsymbol{x}_-[i] = \boldsymbol{o}_2$ and $\boldsymbol{x}_-[j] = \boldsymbol{o}_3$ for $j \neq i$. Then it holds that:

$$\hat{\boldsymbol{w}} \cdot \boldsymbol{x}_+ = \sum_{1 \leq j \leq n} \hat{\boldsymbol{w}}[j] \boldsymbol{x}_+[j] = \boldsymbol{o}_1 \cdot \hat{\boldsymbol{w}}[i] + \sum_{j \neq i} \hat{\boldsymbol{w}}[j] \cdot \boldsymbol{o}_3 \geq 1 \tag{10}$$

and similarly

$$\hat{\boldsymbol{w}} \cdot \boldsymbol{x}_- = \boldsymbol{o}_2 \cdot \hat{\boldsymbol{w}}[i] + \sum_{j \neq i} \hat{\boldsymbol{w}}[j] \cdot \boldsymbol{o}_3 \leq -1 \tag{11}$$

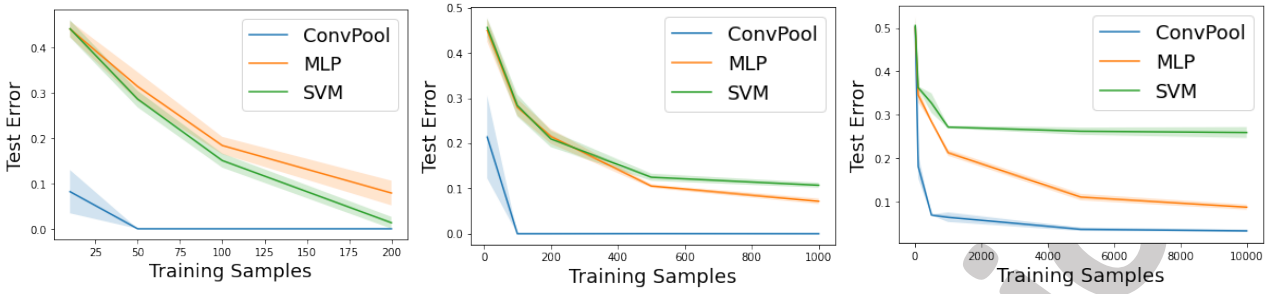
By subtracting Eq. 11 from Eq. 10 we get:

$$\hat{\boldsymbol{w}}[i] \cdot (\boldsymbol{o}_1 - \boldsymbol{o}_2) \geq 2 \tag{12}$$

but since $|\hat{\boldsymbol{w}}[i] \cdot (\boldsymbol{o}_1 - \boldsymbol{o}_2)| \leq 2\|\hat{\boldsymbol{w}}[i]\|$, we have by Eq. 12 $\|\hat{\boldsymbol{w}}[i]\| \geq 1$, which is a contradiction. □

Proposition 6.1 implies that the best possible bound of Brutzkus et al. [2018] for FC networks, or margin bound for linear SVM is $O(n^2)$ in our setting. Thus for $n = \Theta(d^r)$, $r \geq 1$ the bounds for FC networks and linear SVM are $O(d^{2r})$. In contrast, Theorem 5.1 shows a generalization guarantee for CNNs of $O(d)$ for any n. This gap suggests that CNNs should significantly outperform FCNs and linear SVM in our setting. In the next section, we provide empirical evidence for this.

Finally, we note that the bounds suggest why CNNs perform better than FCNs in our setting. While the generalization of CNNs depend on the *pattern dimension*, the generalization bound of FCNs depends on the *number of patterns*. The CNN architecture is invariant to the patch position of each pattern. Furthermore, the distribution structure is also invariant to the position of patterns (e.g., the specific position of the positive pattern does not change the label). By Theorem 5.1, these two facts imply that the CNN can generalize based



(a) (b) (c)

Figure 3: Empirical evaluation of the ConvPool architecture to different baselines. (a) Test error as a function of train sample size, for data sampled according to our pattern detection distribution. Note: this data is linearly separable in $\boldsymbol{x}$ by construction, and we verified that all models had zero training error. (b) Test error as a function of train sample size, for data that as in (a), but with added noise vector $\boldsymbol{v}$ with $\|\boldsymbol{v}\| \leq 1$. The resulting data is not linearly separable. Note: we verified that for the linear model training error was non-zero in many of the cases. For the other models training error was zero. (c) Results on an MNIST pattern detection problem.

on the appearance of the discriminative patterns in any patch in the image. Specifically, the total number of patterns is not relevant for generalization. On the other hand, the FCN bound suggests that FCNs need more samples to generalize because the generalization depends on the number of patterns.

7 EXPERIMENTS

In this section we provide empirical evaluation of learning with our pooling architecture and compare it to several other models. As baselines we consider:

- **ConvPool**: Our convolution and max-pooling model in Eq. 1. We verified that layer-wise training performs very similarly to standard training, and thus we report results on standard training with Adam [Kingma and Ba, 2014] in what follows.
- **MLP**: A standard fully connected neural network with one hidden layer. The network receives the complete $\boldsymbol{x}$ as input (with all patterns). We use a number of hidden neurons that results in the same number of parameters as **ConvPool**.
- **SVM**: A hard-margin linear SVM with $\boldsymbol{x}$ as input. This will return zero training errors only when the data is linearly separable. This is the case for our distribution $\mathcal{D}$, but no longer the case when we add noise to the patterns (see below).

All experiments used a test set of size 1000, and were repeated 5 times with mean and std reported on figures.

We begin with a toy data setting. We created data for a detection problem where all $\boldsymbol{o} \in \mathbb{R}^{20}$ vectors were uniformly sampled from the rows of a uniformly sampled orthogonal matrix and $n = 10$. ConvPool used 500 channels. Figure 3a shows results for this setting. ConvPool can be seen to outperform the other methods. In Figure 3b we go beyond our analyzed setting, and add independent random noise $\boldsymbol{v}$ to each pattern where $\|\boldsymbol{v}\| \leq 1.0$. This makes the problem non linearly-separable. As expected, the linear method now fails, but ConvPool performs well and outperforms MLP.

Figure 4: Data examples in the MNIST detection problem we experiment with in Section 7.

Next, we consider the effect of the number of patterns n on performance. As shown in Proposition 6.1, the norm of the max-margin linear classifier is lower bounded by n. Thus, increasing n is expected to result in worse performance for MLP and SVM by the results in the previous section. In Figure 5, we vary the number of patterns, and indeed observe that performances of MLP and SVM deteriorate while that of ConvPool is only mildly affected (we used the same parameters as above and noise level $\|\boldsymbol{v}\| \leq 1$).

Finally, we evaluate on the MNIST data set. We create a detection problem as in Fig. 4 where the discriminative patterns are the digits three and five and the spurious patterns are all other digits. Each input image contains four patterns (i.e., four digits). We used a relatively small number of patterns to make the problem not linearly separable for moderate sample sizes. We trained a 3 layer convolutional

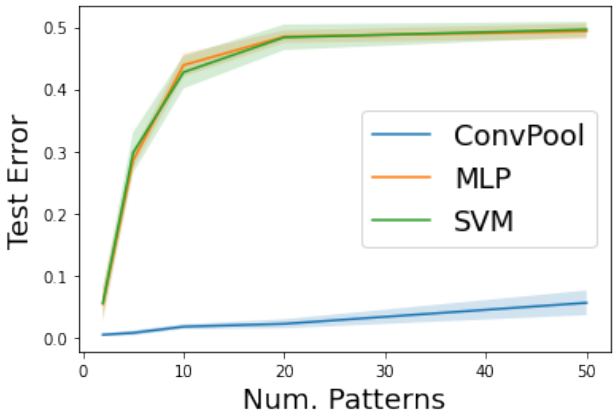


Figure 5: The effect of changing n, the number of patterns per image. It can be seen that this deteriorates the performance of the other methods while only mildly affecting the pooling model.

network as in Eq. 1 with 500 channels. Results in Fig. 3c again show excellent performance of the pooling model compared to the baselines.

8 DISCUSSION

In this paper we presented the first analysis of a convolutional max-pooling architecture in terms of optimization and generalization under over-parameterization. Our analysis is for a natural setting of a detection problem where certain patterns "identify" the class and the others are irrelevant. Our analysis predicts a significant performance gap between CNNs and FCNs, which we observe in experiments.

While our analysis is the first step towards understanding pattern detection architectures, many open problems remain. The first is extending the pattern structure from orthogonal patterns to more general distributions. For example, we can consider the discriminative pattern to be a combination of patterns across the image (e.g., the class of the image is positive only if certain multiple patterns appear in the image). Second, it would be interesting to extend the convolution so that there are overlaps between filters (although this is known to generate local optima even for simpler settings [Brutzkus and Globerson, 2017]). Finally, a challenging extension is to a multi-layer architecture with repeated application of pooling.

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). AB is supported by the Google Doctoral Fellowship in Machine Learning.

References

Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in neural information processing systems*, pages 6155–6166, 2019.

Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. In *Advances in Neural Information Processing Systems*, pages 7411–7422, 2019a.

Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332, 2019b.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Ruslan Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. 2019c.

Eugene Belilovsky, Michael Eickenberg, and Edouard Oyallon. Greedy layerwise learning can scale to imagenet. In *International conference on machine learning*, pages 583–593. PMLR, 2019.

Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K Warmuth. Learnability and the vapnik-chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.

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

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

Alon Brutzkus, Amir Globerson, Eran Malach, and Shai Shalev-Shwartz. SGD learns over-parameterized networks that provably generalize on linearly separable data. *International Conference on Learning Representations*, 2018.

Lenaic Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. *arXiv preprint arXiv:2002.04486*, 2020.

Simon Du, Jason Lee, Yuandong Tian, Aarti Singh, and Barnabas Poczos. Gradient descent learns one-hidden-layer cnn: Don't be afraid of spurious local minima. In *International Conference on Machine Learning*, pages 1339–1348, 2018a.

Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International Conference on Machine Learning*, pages 1675–1685, 2019.

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

Simon S Du, Yining Wang, Xiyu Zhai, Sivaraman Balakrishnan, Russ R Salakhutdinov, and Aarti Singh. How many samples are needed to estimate a convolutional neural network? In *Advances in Neural Information Processing Systems*, pages 373–383, 2018c.

Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *International Conference on Learning Representations*, 2018d.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Jonathan Fiat, Eran Malach, and Shai Shalev-Shwartz. Decoupling gating from linearity. *arXiv preprint arXiv:1906.05032*, 2019.

Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. In *Advances in Neural Information Processing Systems*, pages 9461–9471, 2018.

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

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

Ziwei Ji and Matus Telgarsky. A refined primal-dual analysis of the implicit bias. *arXiv preprint arXiv:1906.04540*, 2019c.

Yiding Jiang, Behnam Neyshabur, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2019.

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

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Eyal Kushilevitz and Dan Roth. On learning visual concepts and DNF formulae. *Machine Learning*, 24(1):65–85, 1996.

Xingguo Li, Junwei Lu, Zhaoran Wang, Jarvis Haupt, and Tuo Zhao. On tighter generalization bound for deep neural networks: Cnns, resnets, and beyond. *arXiv preprint arXiv:1806.05159*, 2018.

Zhiyuan Li, Ruosong Wang, Dingli Yu, Simon S Du, Wei Hu, Ruslan Salakhutdinov, and Sanjeev Arora. Enhanced convolutional neural tangent kernels. *arXiv preprint arXiv:1911.00809*, 2019.

Zhiyuan Li, Yi Zhang, and Sanjeev Arora. Why are convolutional nets more sample-efficient than fully-connected nets? *arXiv preprint arXiv:2010.08515*, 2020.

Philip M Long and Hanie Sedghi. Generalization bounds for deep convolutional neural networks. *ICLR*, 2020.

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

Eran Malach and Shai Shalev-Shwartz. Computational separation between convolutional and fully-connected networks. *arXiv preprint arXiv:2010.01369*, 2020a.

Eran Malach and Shai Shalev-Shwartz. The implications of local correlation on learning some deep functions. *Advances in Neural Information Processing Systems*, 33, 2020b.

Mor Shpigel Nacson, Suriya Gunasekar, Jason Lee, Nathan Srebro, and Daniel Soudry. Lexicographic and depth-sensitive margins in homogeneous and non-homogeneous deep models. In *International Conference on Machine Learning*, pages 4683–4692, 2019.

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

Or Sharir and Amnon Shashua. On the expressive power of overlapping architectures of deep learning. In *International Conference on Learning Representations*, 2018.

Jack Sherman and Winifred J Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, 21(1):124–127, 1950.

Haim Shvaytser. Learnable and nonlearnable visual concepts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(5):459–466, 1990.

Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

Colin Wei, Jason D Lee, Qiang Liu, and Tengyu Ma. Regularization matters: Generalization and optimization of neural nets vs their induced kernel. In *Advances in Neural Information Processing Systems*, pages 9709–9721, 2019.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057, 2015.

Bing Yu, Junzhao Zhang, and Zhanxing Zhu. On the learning dynamics of two-layer nonlinear convolutional neural networks. *arXiv preprint arXiv:1905.10157*, 2019.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. *ICLR*, 2017.

Pan Zhou and Jiashi Feng. Understanding generalization and optimization performance of deep CNNs. In *International Conference on Machine Learning*, pages 5960–5969, 2018.