Learning One Convolutional Layer with Overlapping Patches

Surbhi Goel [1] **Adam Klivans** [1] **Raghu Meka** [2]

Abstract

We give the first provably efficient algorithm for learning a one hidden layer convolutional network with respect to a general class of (potentially overlapping) patches under mild conditions on the underlying distribution. We prove that our framework captures commonly used schemes from computer vision, including one-dimensional and two-dimensional "patch and stride" convolutions. Our algorithm– *Convotron*– is inspired by recent work applying isotonic regression to learning neural networks. Convotron uses a simple, iterative update rule that is stochastic in nature and tolerant to noise (requires only that the conditional mean function is a one layer convolutional network, as opposed to the realizable setting). In contrast to gradient descent, Convotron requires no special initialization or learning-rate tuning to converge to the global optimum. We also point out that learning one hidden convolutional layer with respect to a Gaussian distribution and just *one* disjoint patch P (the other patches may be arbitrary) is *easy* in the following sense: Convotron can efficiently recover the hidden weight vector by updating *only* in the direction of P.

1. Introduction

Developing *provably* efficient algorithms for learning commonly used neural network architectures continues to be a core challenge in machine learning. The underlying difficulty arises from the highly non-convex nature of the optimization problems posed by neural networks. Obtaining provable guarantees for learning even very basic architectures remains open.

In this paper we consider a simple convolutional neural network with a single filter and overlapping patches followed by average pooling (Figure 1). More formally, for an input image x, we consider k *patches* of size r indicated by *selection* matrices $P_1, \ldots, P_k \in \{0,1\}^{r \times n}$ where each matrix has exactly one 1 in each row and at most one 1 in each column. The neural network is computed as $f_w(x) = \frac{1}{k}\sum_{i=1}^{k} \sigma(w^T P_i x)$ where σ is the activation function and $w \in \mathbb{R}^r$ is the weight vector corresponding to the convolution filter. We focus on ReLU and leaky ReLU activation functions.

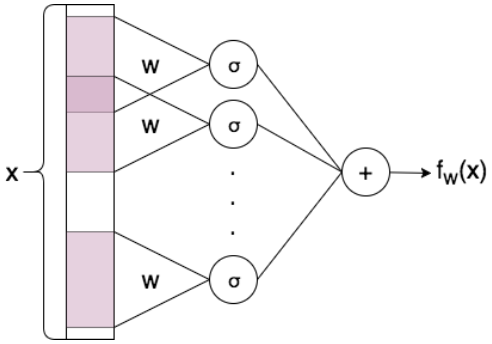

Figure 1. Architecture of convolutional network with one hidden layer and average pooling. Each purple rectangle corresponds to a patch.

1.1. Our Contributions

The main contribution of this paper is a simple, stochastic update algorithm *Convotron* (Algorithm 1) for provably learning the above convolutional architecture. The algorithm has the following properties:

- Works for general classes of overlapping patches and requires mild distributional conditions.
- Proper recovery of the unknown weight vector.
- Stochastic in nature with a "gradient-like" update step.
- Requires no special/random initialization scheme or tuning of the learning rate.
- Tolerates noise and succeeds in the *probabilistic concept* model of learning.
- Logarithmic convergence in $1/\epsilon$, the error parameter, in the realizable setting.

*Equal contribution [1]Department of Computer Science, University of Texas at Austin [2]Department of Computer Science, UCLA. Correspondence to: Surbhi Goel <surbhi@cs.utexas.edu>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

This is the first efficient algorithm for learning general classes of overlapping patches (and the first algorithm for any class of patches that succeeds under mild distributional assumptions). Prior work has focused on analyzing SGD in the realizable/noiseless setting with the caveat of requiring either disjoint patches (Brutzkus & Globerson, 2017; Du et al., 2017b) with Gaussian inputs or technical conditions linking the underlying true parameters and the "closeness of patches" (Du et al., 2017a).

In contrast, our conditions depend only on the patch structure itself and can be efficiently verified. Commonly used patch structures in computer vision applications such as 1D/2D grids satisfy our conditions. Additionally, we require only that the underlying distribution on samples is symmetric and induces a covariance matrix on the patches with polynomially bounded condition number[1]. All prior work handles only continuous distributions. Another major difference from prior work is that we give guarantees using purely empirical updates. That is, we do not require an assumption that we have access to exact quantities such as the population gradient of the loss function.

We further show that in the commonly studied setting of Gaussian inputs and non-overlapping patches, updating with respect to a single non-overlapping patch is sufficient to guarantee convergence. This indicates that the Gaussian/no-overlap assumption is quite strong.

1.2. Our Approach

Our approach is to exploit the monotonicity of the activation function instead of the strong convexity of the loss surface. We use ideas from isotonic regression and extend them in the context of convolutional networks. These ideas have been successful for learning generalized linear models (Kakade et al., 2011), improperly learning fully connected, depth-three neural networks (Goel & Klivans, 2017b), and learning graphical models (Klivans & Meka, 2017).

1.3. Related Work

It is known that in the worst case, learning even simple neural networks is computationally intractable. For example, in the non-realizable (agnostic) setting, it is known that learning a single ReLU (even for bounded distributions and unit norm hidden weight vectors) with respect to square-loss is as hard as learning sparse parity with noise (Goel et al., 2016), a notoriously difficult problem from computational learning theory. For learning one hidden layer convolutional networks, Brutzkus and Globerson (Brutzkus & Globerson, 2017) proved that distribution-free recoverability of the unknown weight vector is NP-hard, even if we restrict to disjoint patch structures.

As such, a major open question is to discover the mildest assumptions that lead to polynomial-time learnability for simple neural networks. In this paper, we consider the very popular class of convolutional neural networks (for a summary of other recent approaches for learning more general architectures see (Goel & Klivans, 2017a)). For convolutional networks, all prior research has focused on analyzing conditions under which (Stochastic) Gradient Descent converges to the hidden weight vector in polynomial-time.

Along these lines, Brutzkus and Globerson (Brutzkus & Globerson, 2017) proved that with respect to the spherical Gaussian distribution and for disjoint (non-overlapping) patch structures, gradient descent recovers the weight vector in polynomial-time. Zhong et al. (Zhong et al., 2017a) showed that gradient descent combined with tensor methods can recover one hidden layer involving multiple weight vectors but still require a Gaussian distribution and non-overlapping patches. Du et al. (Du et al., 2017b) proved that gradient descent recovers a hidden weight vector involved in a type of two-layer convolutional network under the assumption that the distribution is a spherical Gaussian, the patches are disjoint, and the learner has access to the true population gradient of the loss function.

We specifically highlight the work of Du, Lee, and Tian (Du et al., 2017a), who proved that gradient descent recovers a hidden weight vector in a one-layer convolutional network under certain technical conditions that are more general than the Gaussian/no-overlap patch scenario. Their conditions involve a certain "alignment" of the unknown patch structure, the hidden weight vector, and the (continuous) marginal distribution. However, it is unclear which concrete patch-structure/distributional combinations their framework captures. We also note that all of the above results assume there is no noise; i.e.,, they work in the realizable setting.

Other related works analyzing gradient descent with respect to the Gaussian distribution (but for non-convolutional networks) include (Soltanolkotabi, 2017; Ge et al., 2017; Zhong et al., 2017b; Tian, 2016; Li & Yuan, 2017; Zhang et al., 2017).

In contrast, we consider an alternative to gradient descent, namely Convotron, that is based on isotonic regression. The exploration of alternative algorithms to gradient descent is a feature of our work, as it may lead to new algorithms for learning deeper networks.

[1] Brutzkus and Globerson (Brutzkus & Globerson, 2017) proved that the problem, even with disjoint patches, is NP-hard in general, and so some distributional assumption is needed for efficient learning.

2. Preliminaries

$||\cdot||$ corresponds to the l_2 -norm for vectors and the spectral norm for matrices. The identity matrix is denoted by I. We denote the input-label distribution by $\mathcal{D}$ over input drawn from $\mathcal{X}$ and label drawn from $\mathcal{Y}$. The marginal distribution on the input is denoted by $\mathcal{D}_{\mathcal{X}}$ and the corresponding probability density function is denoted by $P_{\mathcal{X}}$.

In this paper we consider a simple convolution neural network with one hidden layer and average pooling. Given input $x \in \mathbb{R}^n$, the network computes k patches of size r where each patch's location is indicated by matrices $P_1, \ldots, P_k \in \{0,1\}^{r \times n}$. Each matrix P_i has exactly one 1 in each row and at most one 1 in every column. As before, the neural network is computed as follows:

$$f_w(x) = \frac{1}{k} \sum_{i=1}^{k} \sigma\left(w^T P_i x\right)$$

where σ is the activation function and $w \in \mathbb{R}^r$ is the weight vector corresponding to the convolution filter.

We study the problem of learning the teacher network with true weight w_* under the square loss from noisy labels, that is, we wish to find a w such that

$$L(w) := \mathbb{E}_{x \sim \mathcal{D}_{\mathcal{X}}}\left[(f_w(x) - f_{w_*}(x))^2\right] \leq \epsilon.$$

Assumptions 1. *We make the following assumptions:*

(a) ***Learning Model****: Probabilistic Concept Model (Kearns & Schapire, 1990), that is, for all $(x, y) \sim \mathcal{D}$, $y = f_{w_*}(x) + \xi$, for some unknown w_* where ξ is noise with $\mathbb{E}[\xi|x] = 0$ and $\mathbb{E}[\xi^4|x] \leq \rho$ for some $\rho > 0$. Note we do not require that the noise is independent of the instance.*[2]

(b) ***Distribution****: The marginal distribution on the input space $\mathcal{D}_{\mathcal{X}}$ is a symmetric distribution about the origin, that is, for all x, $P_{\mathcal{X}}(x) = P_{\mathcal{X}}(-x)$.*

(c) ***Patch Structure****: The minimum eigenvalue of $P_\Sigma := \sum_{i,j=1}^{k} P_i \Sigma P_j^T$ where $\Sigma = \mathbb{E}_{x \sim \mathcal{D}_{\mathcal{X}}}[xx^T]$ and the maximum eigenvalue of $P := \sum_{i,j=1}^{k} P_i P_j^T$ are polynomially bounded.*

(d) ***Activation Function****: The activation function has the following form:*

$$\sigma(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{otherwise} \end{cases}$$

for some constant $\alpha \in [0, 1]$.

[2] In the realizable setting, as in previous works, it is assumed that $\xi = 0$.

The distributional assumption includes common assumptions such as Gaussian inputs, but is far less restrictive. For example, we do not require the distribution to be continuous nor do we require it to have identity covariance. In Section 4, we show that commonly used patch schemes from computer vision satisfy our patch requirements. The assumption on activation functions is satisfied by popular activations such as ReLU ($\alpha = 0$) and leaky ReLU ($\alpha > 0$).

2.1. Some Useful Properties

The activations we consider in this paper have the following useful property under the stated distributional assumption:

Lemma 1. *For all $a, b \in \mathbb{R}$,*

$$\mathbb{E}_{x \sim \mathcal{D}_{\mathcal{X}}}[\sigma(a^T x)(b^T x)] = \frac{1+\alpha}{2} \mathbb{E}_{x \sim \mathcal{D}_{\mathcal{X}}}[(a^T x)(b^T x)].$$

The loss function can be upper bounded by the l_2-norm distance of weight vectors using the following lemma.

Lemma 2. *For any w, we have*

$$L(w) \leq \frac{1+\alpha}{2} \lambda_{\max}(\Sigma) ||w_* - w||^2.$$

Lemma 3. *For all w and x,*

$$(f_{w_*}(x) - f_w(x))^2 \leq ||w_* - w||^2 ||x||^2$$

The Gershgorin Circle Theorem, stated below, is useful for bounding the eigenvalues of matrices.

Theorem 1 ((Weisstein, 2003))**.** *For a $n \times n$ matrix A, define $R_i := \sum_{j=1, j \neq i}^{n} |A_{i,j}|$. Each eigenvalue of A must lie in at least one of the disks $\{z : |z - A_{i,i}| \leq R_i\}$.*

Note: The proofs of lemmas in this section have been deferred to the Supplemental section.

3. The Convotron Algorithm

In this section we describe our main algorithm *Convotron* and give a proof of its correctness. Convotron is an iterative algorithm similar in flavor to SGD with a modified (aggressive) gradient update. Unlike SGD (Algorithm 3), Convotron comes with provable guarantees and also does not need a good initialization scheme for convergence.

The following theorem describes the convergence rate of our algorithm:

Theorem 2. *If Assumptions 1 are satisfied then for $\eta = \Omega\left(\frac{\lambda_{\min}(P_\Sigma)}{k \lambda_{\max}(P)} \min\left(\frac{1}{\mathbb{E}_x[||x||^4]}, \frac{\epsilon\delta||w_*||^2}{\sqrt{\rho \mathbb{E}_x[||x||^4]}}\right)\right)$ and $T = O\left(\frac{k}{\eta \lambda_{\min}(P_\Sigma)} \log\left(\frac{1}{\epsilon\delta}\right)\right)$, with probability $1 - \delta$, the weight vector w computed by Convotron satisfies*

$$||w - w_*||^2 \leq \epsilon ||w_*||^2.$$

Algorithm 1 Convotron

Initialize $w_1 := 0 \in \mathbb{R}^r$.
for $t = 1$ **to** T **do**
  Draw $(x_t, y_t) \sim \mathcal{D}$
  Let $G_t = (y_t - f_{w_t}(x_t)) \left(\sum_{i=1}^k P_i x_t\right)$
  Set $w_{t+1} = w_t + \eta G_t$
end for
Return w_{T+1}

Proof. Define $S_t = \{(x_1, y_1), \ldots, (x_t, y_t)\}$ The dynamics of Convotron can be expressed as follows:

$$\begin{aligned}&\mathbb{E}_{x_t,y_t}[||w_t - w_*||^2 - ||w_{t+1} - w_*||^2 | S_{t-1}] \\ &= 2\eta \mathbb{E}_{x_t,y_t}[(w_* - w_t)^T G_t | S_{t-1}] \\ &\quad - \eta^2 \mathbb{E}_{x_t,y_t}[||G_t||^2 | S_{t-1}]\end{aligned}$$

We need to bound the RHS of the above equation. We have,

$$\begin{aligned}&\mathbb{E}_{x_t,y_t}[(w_* - w_t)^T G_t | S_{t-1}] \\ &= \mathbb{E}_{x_t,y_t}\left[(w_* - w_t)^T (y_t - f_{w_t}(x_t)) \left(\sum_{i=1}^k P_i x_t\right) \middle| S_{t-1}\right] \\ &= \mathbb{E}_{x_t,\xi_t}\left[(w_* - w_t)^T (f_{w_*}(x_t) + \xi_t \right. \\ &\qquad \left. - f_{w_t}(x_t)) \left(\sum_{i=1}^k P_i x_t\right) \middle| S_{t-1}\right] \\ &= \mathbb{E}_{x_t}\left[(w_* - w_t)^T (f_{w_*}(x_t) - f_{w_t}(x_t)) \left(\sum_{i=1}^k P_i x_t\right) \middle| S_{t-1}\right] \quad (1) \\ &= \frac{1}{k} \sum_{1 \le i,j \le k} \mathbb{E}_{x_t}[(\sigma(w_*^T P_i x_t) - \sigma(w_t^T P_i x_t)) \quad (2) \\ &\qquad (w_*^T - w_t^T) P_j x_t | S_{t-1}] \\ &= \frac{1+\alpha}{2k} \sum_{1 \le i,j \le k} \mathbb{E}_{x_t}[((w_*^T - w_t^T) P_i x_t) \\ &\qquad ((w_*^T - w_t^T) P_j x_t) | S_{t-1}] \quad (3) \\ &= \frac{1+\alpha}{2k} (w_*^T - w_t^T) \left(\sum_{1 \le i \le k} P_i\right) \mathbb{E}_{x_t}[x_t x_t^T] \\ &\qquad \left(\sum_{1 \le j \le k} P_j^T\right) (w_* - w_t) \\ &= \frac{1+\alpha}{2k} (w_*^T - w_t^T) \left(\sum_{1 \le i,j \le k} P_i \Sigma P_j^T\right) (w_* - w_t) \\ &= \frac{1+\alpha}{2k} (w_*^T - w_t^T) P_\Sigma (w_* - w_t) \\ &\ge \frac{1+\alpha}{2k} \lambda_{\min}(P_\Sigma) ||w_* - w_t||^2. \quad (4)\end{aligned}$$

(1) follows using linearity of expectation and the fact that that $\mathbb{E}[\xi_t | x_t] = 0$ and (3) follows from using Lemma 1. (4) follows from observing that P_Σ is symmetric, thus $\forall x, x^T P_\Sigma x \ge \lambda_{\min}(P_\Sigma) ||x||^2$.

Now we bound the variance of G_t. Note that $\mathbb{E}[G_t] = 0$. Further,

$$\begin{aligned}&\mathbb{E}_{x_t,y_t}[||G_t||^2 | S_{t-1}] \\ &= \mathbb{E}_{x_t,y_t}\left[(y_t - f_{w_t}(x_t))^2 \left|\left|\sum_{i=1}^k P_i x_t\right|\right|^2 \middle| S_{t-1}\right] \\ &\le \lambda_{\max}(P) \mathbb{E}_{x_t,y_t}\left[(y_t - f_{w_t}(x_t))^2 ||x_t||^2 \middle| S_{t-1}\right] \quad (5) \\ &= \lambda_{\max}(P) \mathbb{E}_{x_t,\xi_t}\left[(f_{w_*}(x_t) + \xi_t - f_{w_t}(x_t))^2 ||x_t||^2 \middle| S_{t-1}\right] \\ &= \lambda_{\max}(P) \mathbb{E}_{x_t,\xi_t}\left[((f_{w_*}(x_t) - f_{w_t}(x_t))^2 + \xi_t^2 \right. \\ &\quad \left. + 2(f_{w_*}(x_t) - f_{w_t}(x_t))\xi_t ||x_t||^2 \middle| S_{t-1}\right] \\ &= \lambda_{\max}(P) \left(\mathbb{E}_{x_t}\left[(f_{w_*}(x_t) - f_{w_t}(x_t))^2 ||x_t||^2 \middle| S_{t-1}\right] \right. \\ &\quad \left. + \mathbb{E}_{x_t,\xi_t}[\xi_t^2 ||x_t||^2]\right) \quad (6) \\ &\le \lambda_{\max}(P) \left(\mathbb{E}_{x_t}[||x_t||^4] ||w_* - w_t||^2 + \sqrt{\rho \mathbb{E}_{x_t}[||x_t||^4]}\right) \quad (7)\end{aligned}$$

(5) follows from observing $\left|\left|\sum_{i=1}^k P_i x\right|\right|^2 \le \lambda_{\max}(P)||x||^2$ for all x, (6) follows from observing $\mathbb{E}_\xi[\xi | x] = 0$ and (7) follows from applying Lemma 3 and bounding $\mathbb{E}_{x_t,\xi_t}[\xi_t^2 ||x_t||^2]$ using Cauchy-Schwartz inequality.

Combining the above equations and taking expectation over S_{t-1}, we get

$$\begin{aligned}&\mathbb{E}_{S_t}[||w_{t+1} - w_*||^2] \\ &\quad \le (1 - 3\eta\beta + \eta^2\gamma) \mathbb{E}_{S_{t-1}}[||w_t - w_*||^2] + \eta^2 B\end{aligned}$$

for $\beta = \frac{1+\alpha}{3k} \lambda_{\min}(P_\Sigma)$, $\gamma = \lambda_{\max}(P) \mathbb{E}_x[||x||^4]$ and $B = \lambda_{\max}(P) \sqrt{\rho \mathbb{E}_x[||x||^4]}$.

We set $\eta = \beta \min\left(\frac{1}{\gamma}, \frac{\epsilon\delta||w_*||^2}{B}\right)$ and break the analysis to two cases:

Case 1: $\mathbb{E}_{S_{t-1}}[||w_t - w_*||^2] > \frac{\eta B}{\beta}$. This implies that $\mathbb{E}_{S_t}[||w_{t+1} - w_*||^2] \le (1 - \eta\beta) \mathbb{E}_{S_{t-1}}[||w_t - w_*||^2]$.

Case 2: $\mathbb{E}_{S_{t-1}}[||w_t - w_*||^2] \le \frac{\eta B}{\beta} \le \epsilon ||w_*||^2$.

Observe that once Case 2 is satisfied, we have $\mathbb{E}_{S_t}[||w_{t+1} - w_*||^2] \le (1 - 2\eta\beta)\frac{\eta B}{\beta} + \eta^2 B \le \frac{\eta B}{\beta}$. Hence, for any iteration $> t$, Case 2 will continue to hold true. This implies that either at each iteration $\mathbb{E}_{S_{t-1}}[||w_t - w_*||^2]$ decreases by a factor $(1 - \eta\beta)$ or it is less than $\epsilon\delta||w_*||^2$. Thus if Case 1 is not satisfied for any iteration up to T, then we have,

$$\mathbb{E}_{S_T}[||w_{T+1} - w||^2] \le (1 - \eta\beta)^T ||w_*||^2 \le e^{-\eta\beta T} ||w_*||^2$$

since at initialization $||w_1 - w_*|| = ||w_*||$. Setting $T = O\left(\frac{1}{\eta\beta}\log\left(\frac{1}{\epsilon\delta}\right)\right)$ and using Markov's inequality, with probability $1-\delta$, over the choice of S_T, $||w_{T+1} - w_*|| \leq \epsilon||w_*||^2$. □

By using Lemma 2, we can get a bound on $L(w_T) \leq \epsilon||w_*||^2$ by appropriately scaling ϵ.

3.1. Convotron in the Realizable Case

For the realizable (no noise) setting, that is, for all $(x, y) \sim \mathcal{D}$, $y = f_{w_*}(x)$, for some unknown w_*, Convotron achieves faster convergence rates.

Corollary 1. *If Assumptions 1 are satisfied with the learning model restricted to the realizable case, then for suitably choosen η, after $T = O\left(\frac{k^2\lambda_{\max}(P)\mathbb{E}_x[||x||^4]}{\lambda_{\min}(P_\Sigma)^2}\log\left(\frac{1}{\epsilon\delta}\right)\right)$ iterations, with probability $1-\delta$, the weight vector w computed by Convotron satisfies*

$$||w - w_*||^2 \leq \epsilon||w_*||^2.$$

Proof. Since the setting has no noise, $\rho = 0$. Setting that parameter in Theorem 2 gives us $\eta = \Omega\left(\frac{\lambda_{\min}(P_\Sigma)}{k\lambda_{\max}(P)\mathbb{E}_x[||x||^4]}\right)$ as $\frac{\epsilon\delta||w_*||^2}{\sqrt{\rho\mathbb{E}_x[||x||^4]}}$ tends to infinity as ρ tends to 0 and taking the minimum removes this dependence from η. Substituting this η gives us the required result. □

Observe that the dependence of ϵ in the convergence rate is $\log(1/\epsilon)$ for the realizable setting, compared to the $1/\epsilon$ dependence in the noisy setting.

4. Which Patch Structures are Easy to Learn?

In this section, we will show that the commonly used convolutional filters in practice ("patch and stride") have good eigenvalues giving us fast convergence by Theorem 2. We will start with the 1D case and then subsequently extend the result for the 2D case.

4.1. 1D Convolution

Here we formally describe a patch and stride convolution in the one-dimensional setting. Consider a 1D image of dimension n. Let the patch size be r and stride be d. Let the patches be indexed from 1 and let patch i start at position $(i-1)d+1$ and be contiguous through position $(i-1)d+r$. The matrix P_i of dimension $r \times n$ corresponding to patch i looks as follows,

$$P_i = \left(0_{r\times((i-1)d+1)} I_r 0_{r\times(n-r-(i-1)d)}\right)$$

where $0_{a\times b}$ indicates a matrix of dimension $a \times b$ with all zeros and I_a indicates the identity matrix of size a.

Thus, the total number of patches is $k = \lfloor\frac{n-r}{d}\rfloor + 1$. We will assume that $n \geq 2r-1$ and $r \geq d$. The latter condition is to ensure there is some overlap, non-overlapping case, which is easier, is handled in the next section.

We will bound the extremal eigenvalues of $P = \sum_{i,j=1}^{k} P_i P_j^T$. Simple algebra gives us the following structure for P,

$$P_{i,j} = \begin{cases} k-a & \text{if } |i-j| = ad \\ 0 & \text{otherwise} \end{cases}$$

For understanding, we show the matrix structure for $d = 1$ and $n \geq 2r$.

$$\begin{pmatrix} k & k-1 & \dots & k-r+1 \\ k-1 & k & \dots & k-r+2 \\ \vdots & \vdots & \ddots & \vdots \\ k-r+1 & k-r+2 & \dots & k \end{pmatrix}.$$

4.1.1. Bounding Extremal Eigenvalues of P

The following lemmas bound the extremal eigenvalues of P.

Lemma 4. *Maximum eigenvalue of P satisfies $\lambda_{\max}(P) \leq k(p+1) - (p-p_2)(p_2+1) = O(kp)$ where $p = \lfloor\frac{r-1}{d}\rfloor$ and $p_2 = \lfloor\frac{p}{2}\rfloor$.*

Proof. Using Theorem 1, we have $\lambda_{\max}(P) \leq \max_i\left(P_{i,i} + \sum_{j\neq i}|P_{i,j}|\right) = \max_i \sum_{j=1}^{k} P_{i,j}$. Observe that P is bisymmetric thus $\sum_{j=1}^{k} P_{i,j} = \sum_{j=1}^{k} P_{r-i+1,j}$ and we can restrict to the top half of the matrix. The structure of P indicates that in a fixed row, the diagonal entry is maximum and the non-zero entries decrease monotonically by 1 as we move away from the diagonal. Also, there can be at most $p+1$ non-zero entries in any row. Thus the sum is maximized when there are $p+1$ non-zero entries and the diagonal entry is the middle entry, that is at position $p_2 d + 1$. By simple algebra,

$$\begin{aligned}\lambda_{\max}(P) &\leq \sum_{j=1}^{k} P_{p_2 d+1,j} \\ &= k + 2\sum_{j=1}^{p_2}(k-j) + (p-2p_2)(k-p_2-1) \\ &= k(p+1) - (p-p_2)(p_2+1).\end{aligned}$$

□

Lemma 5. *Minimum eigenvalue of P satisfies $\lambda_{\min}(P) \geq 0.5$.*

Proof. We break the analysis into following two cases:

$\mathbf{d < r/2}$: We can show that $\lambda_{\max}(P^{-1}) \geq 2$ using the structure of P (see Lemma A and B in Supplemental). Since $\lambda_{\min}(P) = 1/\lambda_{\max}(P^{-1})$, we have $\lambda_{\min}(P) \geq 0.5$.

$\mathbf{d \geq r/2}$: In this case we directly bound the minimum eigenvalue of P. Using Theorem 1, we know that $\lambda_{\min}(P) \geq \min_i \left(P_{i,i} - \sum_{j \neq i} |P_{i,j}|\right)$. For $P_{i,j} \neq 0$, $|i-j| = ad$ for some a. The maximum value that $|i-j|$ can take is $r-1$ and since $d \geq r/2$, a must be either 0 or 1. Also, for any i, there exists a unique j such that $|i-j| = d$ since $r/2 \leq d < r$, thus there are exactly 2 non-zero entries in each row of P, $P_{i,i}$. This gives us, for each i, $\sum_{j \neq i} P_{i,j} = k-1$. Thus, we get that $\lambda_{\min}(P) \geq \min_i \left(P_{i,i} - \left|\sum_{j \neq i} P_{i,j}\right|\right) = k - (k-1) = 1$.

Combining both, we get the required result. □

4.1.2. Learning Result for 1D

Augmenting the above analysis with Theorem 2 gives us learnability of 1D convolution filters.

Corollary 2. *If Assumptions 1(a),(b), and (d) are satisfied and the patches have a patch and stride structure with parameters n, r, d, then for suitably chosen η and $T = O\left(\frac{n^3 r}{d^4 \lambda_{\min}(\Sigma)^2} \max\left(\mathbb{E}_x[||x||^4], \frac{\sqrt{\rho \mathbb{E}_x[||x||^4]}}{\epsilon ||w_*||^2}\right) \log\left(\frac{1}{\epsilon\delta}\right)\right)$, with probability $1-\delta$, the weight vector w output by Convotron satisfies*

$$||w - w_*||^2 \leq \epsilon ||w_*||^2.$$

Proof. Combining the above Lemmas gives us that $\lambda_{\max}(P) = O(pk) = O(nr/d^2)$ and $\lambda_{\min}(P) = \Omega(1)$. Observe that $\lambda_{\min}(P_\Sigma) \geq \lambda_{\min}(P)\lambda_{\min}(\Sigma)$. Substituting these values in Theorem 2 gives us the desired result. □

Comparing with SGD, (Brutzkus & Globerson, 2017) showed that even for $r = 2$ and $d = 1$, Gradient descent can get stuck in a local minima with probability $\geq 1/4$.

4.2. 2D Convolution

Here we formally define stride and patch convolutions in two dimensions. Consider a 2D image of dimension $n_1 \times n_2$. Let the patch size be $r_1 \times r_2$ and stride in both directions be d_1, d_2 respectively. Enumerate patches such that patch (i,j) starts at position $((i-1)d_1 + 1, (j-1)d_2 + 1)$ and is a rectangle with diagonally opposite point $((i-1)d_2 + r_1, (j-1)d_2 + r_2)$. Let $k_1 = \lfloor \frac{n_1 - r_1}{d_1} \rfloor + 1$ and $k_2 = \lfloor \frac{n_2 - r_2}{d_2} \rfloor + 1$. Let us vectorize the image row-wise into a $n_1 n_2$ dimension vector and enumerate each patch row-wise to get a $r_1 r_2$ dimensional vector. Let $Q_{(i,j)}$ be the indicator matrix of dimension $r_1 r_2 \times n_1 n_2$ with 1 at (a,b) if the ath location of patch (i,j) is b. More formally, $(Q_{(i,j)})_{a,b} = 1$ for all $a = pr_2 + q + 1$ for $0 \leq p < r_1$, $0 \leq q < r_2$, and $b = ((i-1)d_1 + p)n_2 + jd_2 + q + 1$ else 0. Note that there are $k_1 \cdot k_2$ patches in total with the corresponding patch matrices being $Q_{(i,j)}$ for $1 \leq i \leq k_1, 1 \leq j \leq k_2$.

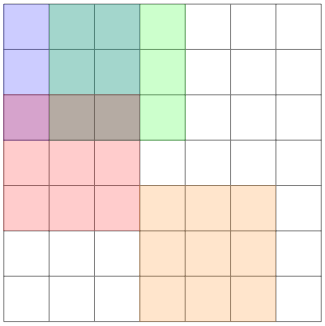

Figure 2. 2D convolution patches for image size $n_1 = n_2 = 7$, patch size $r_1 = r_2 = 3$, and stride $d_1 = 2$, $d_2 = 1$. Blue box corresponds to patch $(1,1)$, red to patch $(2,1)$ green to patch $(1,2)$ and orange to patch $(3,4)$.

4.2.1. Bounding Extremal Eigenvalues of Q

We will bound the extremal eigenvalues of $Q = \sum_{i,p=1}^{k_1} \sum_{j,q=1}^{k_2} Q_{(i,j)} Q_{(p,q)}^T$. Let $P_i^{(1)}$'s be the patch matrices corresponding to the 1D convolution for parameters n_1, r_1, d_1 defined as in the previous section and let $P^{(1)} = \sum_{i,j=1}^{k_1} P_i^{(1)} (P_j^{(1)})^T$. Define $P_i^{(2)}$'s for $1 \leq i \leq k_2$ and $P^{(2)}$ similarly with parameters n_2, r_2, d_2 instead of n_1, r_1, d_1.

Lemma 6. $Q_{(i,j)} = P_i^{(1)} \otimes P_j^{(2)}$.

Proof. Intuitively $P_i^{(1)}$ and $P_j^{(2)}$ give the indices corresponding to the row and column of the 2D patch and the Kronecker product vectorizes it to give us the (i,j)th patch. More formally, we will show that $(Q_{(i,j)})_{a,b} = 1$ iff $(P_i^{(1)} \otimes P_j^{(2)})_{a,b} = 1$.

Let $a = pr_2 + q + 1$ with $0 \leq p < r_1$, $0 \leq q < r_2$ and $b = rn_2 + s + 1$ with $0 \leq r < n_1$, $0 \leq s < n_2$. Then, $(P_i^{(1)} \otimes P_j^{(2)})_{a,b} = 1$ iff $(P_i^{(1)})_{p,r} = 1$ and $(P_j^{(2)})_{q,s} = 1$. We know that $(P_i^{(1)})_{p,r} = 1$ iff $r = (i-1)d_1 + p + 1$ and $(P_j^{(2)})_{q,s} = 1$ iff $s = (j-1)d_2 + q + 1$. This gives us that $b = ((i-1)d_1 + p)n_1 + (j-1)d_2 + q + 1$, which is the same condition for $(Q_{(i,j)})_{a,b} = 1$. Thus $Q_{(i,j)} = P_i^{(1)} \otimes P_j^{(2)}$. □

Lemma 7. $Q = P^{(1)} \otimes P^{(2)}$.

Proof. We have,

$$Q = \sum_{i,p=1}^{k_1} \sum_{j,q=1}^{k_2} Q_{(i,j)} Q_{(p,q)}^T$$

$$
\begin{aligned}
&= \sum_{i,p=1}^{k_1}\sum_{j,q=1}^{k_2}(P_i^{(1)}\otimes P_j^{(2)})(P_p^{(1)}\otimes P_q^{(2)})^T \\
&= \sum_{i,p=1}^{k_1}\sum_{j,q=1}^{k_2}(P_i^{(1)}\otimes P_j^{(2)})((P_p^{(1)})^T\otimes (P_q^{(2)})^T) \\
&= \sum_{i,p=1}^{k_1}\sum_{j,q=1}^{k_2}(P_i^{(1)}(P_p^{(1)})^T)\otimes (P_j^{(2)}(P_q^{(2)})^T) \\
&= \left(\sum_{i,p=1}^{k_1} P_i^{(1)}(P_p^{(1)})^T\right)\otimes\left(\sum_{j,q=1}^{k_2} P_j^{(2)}(P_q^{(2)})^T\right) \\
&= P^{(1)}\otimes P^{(2)}.
\end{aligned}
$$

□

Lemma 8. *We have* $\lambda_{\min}(Q) \geq 0.25$ *and* $\lambda_{\max}(Q) = O(k_1p_1k_2p_2)$ *where* $p_1 = \lfloor \frac{r_1-1}{d_1} \rfloor$ *and* $p_2 = \lfloor \frac{r_2-1}{d_2} \rfloor$.

Proof. Since $Q = P^{(1)} \otimes P^{(2)}$ and $Q, P^{(1)}, P^{(2)}$ are positive semi-definite, $\lambda_{\min}(Q) = \lambda_{\min}(P)\lambda_{\min}(P^{(2)})$ and $\lambda_{\max}(Q) = \lambda_{\max}(P^{(1)})\lambda_{\max}(P^{(2)})$. Using the lemmas from the previous section gives us the required result. □

Note that this technique can be extended to higher dimensional patch structures as well.

4.2.2. Learning Result for 2D

Similar to the 1D case, combining the above analysis with Theorem 2 gives us learnability of 2D convolution filters.

Corollary 3. *If Assumptions 1(a),(b), and (d) are satisfied and the patches have a 2D patch and stride structure with parameters* $n_1, n_2, r_1, r_2, d_1, d_2$, *then for suitably chosen* η *and* $T = O\left(\frac{n_1^3 n_2^3 r_1 r_2}{d_1^3 d_2^3 \lambda_{\min}(\Sigma)^2}\max\left(\mathbb{E}_x[||x||^4], \frac{\sqrt{\rho\mathbb{E}_x[||x||^4]}}{\epsilon||w_*||^2}\right)\log\left(\frac{1}{\epsilon\delta}\right)\right)$, *with probability* $1-\delta$, *the weight vector* w *output by Convotron satisfies*

$$||w - w_*||^2 \leq \epsilon||w_*||^2.$$

Proof. Lemma 8 gives us that $\lambda_{\max}(Q) = O(n_1n_2r_1r_2/(d_1d_2)^2)$ and $\lambda_{\min}(P) = \Omega(1)$. Observe that $\lambda_{\min}(P_\Sigma) \geq \lambda_{\min}(P)\lambda_{\min}(\Sigma)$. Substituting these values in Theorem 2 gives us the desired result. □

5. Non-overlapping Patches are Easy

In this section, we will show that if there is one patch that does not overlap with any patch and the covariance matrix is identity then we can easily learn the filter even if the other patches have arbitrary overlaps. This includes the commonly used Gaussian assumption. WLOG we assume that P_1 is the patch that does not overlap with any other patch implying $P_1P_j^T = P_j^TP_1 = 0$ for all $j \neq 1$.

Algorithm 2 Convotron-No-Overlap

Initialize $w_1 := 0 \in \mathbb{R}^r$.
for $t = 1$ **to** T **do**
  Draw $(x_t, y_t) \sim \mathcal{D}$
  Let $G_t = (y_t - f_{w_t}(x_t))P_1x_t$
  Set $w_{t+1} = w_t + \eta G_t$
end for
Return w_{T+1}

Observe that the algorithm ignores the directions of all other patches and yet succeeds. This indicates that with respect to a Gaussian distribution, in order to have an interesting patch structure (for one layer networks), it is necessary to avoid having even a single disjoint patch. The following theorem shows the convergence of Convotron-No-Overlap.

Theorem 3. *If Assumptions 1 are satisfied with* $\Sigma = I$, *then for* $\eta = \frac{(1+\alpha)}{3k}\min\left(\frac{1}{\mathbb{E}_x[||x||^4]}, \frac{\epsilon\delta||w_*||^2}{\sqrt{\rho\mathbb{E}_x[||x||^4]}}\right)$ *and* $T \geq \frac{1}{\eta\delta}\log\left(\frac{1}{\epsilon\delta}\right)$, *with probability* $1-\delta$, *the weight vector* w *outputted by Convotron-No-Overlap satisfies*

$$||w - w_*||^2 \leq \epsilon||w_*||^2.$$

Proof. The proof follows the outline of the Convotron proof very closely. We use the same definitions as in the previous proof. We have,

$$
\begin{aligned}
&\mathbb{E}_{x_t,y_t}[(w_* - w_t)^TG_t|S_{t-1}] \\
&= \frac{1}{k}\sum_{1\leq i\leq k}\mathbb{E}_{x_t}[(\sigma(w_*^TP_ix_t) - \sigma(w_t^TP_ix_t))(w_*^T \\
&\qquad - w_t^T)P_1x_t|S_{t-1}] \\
&= \frac{1+\alpha}{2k}\sum_{1\leq i\leq k}\mathbb{E}_{x_t}[((w_*^T - w_t^T)P_ix_t)((w_*^T \\
&\qquad - w_t^T)P_1x_t)|S_{t-1}] \\
&= \frac{1+\alpha}{2k}(w_*^T - w_t^T)\left(\sum_{1\leq i\leq k}P_i\right)\mathbb{E}_{x_t}[x_tx_t^T]P_1(w_* - w_t) \\
&= \frac{1+\alpha}{2k}||w_*^T - w_t^T||^2
\end{aligned}
$$

The last equality follows since $P_i^TP_1 = 0$ for all $i \neq 1$ and $P_1^TP_1$ is a permutation of identity.

Similarly,

$$
\begin{aligned}
&\mathbb{E}_{x_t,y_t}[||G_t||^2|S_{t-1}] \\
&= \mathbb{E}_{x_t,y_t}\left[(y_t - f_{w_t}(x_t))^2\,||P_ix_t||^2\middle|S_{t-1}\right] \\
&\leq \mathbb{E}_{x_t,y_t}\left[(y_t - f_{w_t}(x_t))^2||x_t||^2\middle|S_{t-1}\right]
\end{aligned}
$$

Algorithm 3 SGD

Randomly initialize $w_1 \in \mathbb{R}^r$.
for $t = 1$ **to** T **do**
  Draw $(x_t, y_t) \sim \mathcal{D}$
  Let $G_t = (y_t - f_{w_t}(x_t)) \left(\sum_{i=1}^{k} \sigma'(w_t^T P_i x_t) P_i x_t \right)$
  Set $w_{t+1} = w_t + \eta G_t$
end for
Return w_{T+1}

$$\leq \mathbb{E}_{x_t}[||x_t||^4]||w_* - w_t||^2 + \sqrt{\rho \mathbb{E}_{x_t}[||x_t||^4]}$$

Following the rest of the analysis for η and T as in the theorem statement gives us the required result. □

6. Experiments: SGD vs Convotron

To further support our theoretical findings, we empirically compare the performance of SGD (Algorithm 3) with our algorithm Convotron. We measure performance based on the failure probability, that is, the fraction of runs the algorithm fails to converge on randomly initialized runs (the randomness is over both the choice of initialization for SGD and the draws from the distribution). More formally, we say that the algorithm fails if the closeness in l_2-norm of the difference of the final weight vector obtained (w_T) and the true weight parameter (w_*), that is, $||w_T - w_*||$ is greater than a threshold θ. We choose this measure because in practice, due to the high computation time of training neural networks, random restarts are expensive.

In the experiments, given a fixed true weight vector, for varying learning rates (increments of 0.01), we choose 50 random initializations and run the two algorithms with them as starting points. We plot the failure probability ($\theta = 0.1$) with varying learning rate. Note that the lowest learning rate we use is 0.01 as making the learning rate too small requires high number of iterations for convergence for both algorithms.

We first test the performance on a simple 1D convolution case with $(n, k, d, T) = (8, 4, 1, 6000)$ and 2D case with $(n_1, n_2, k_1, k_2, d_1, d_2, T) = (5, 5, 3, 3, 1, 1, 15000)$ on inputs drawn from a normalized (l_2 norm 1) Gaussian distribution with identity covariance matrix. We adversarially choose a fixed weight vector (l_2-norm 1). For example, we take the vector to be $[1, -1, 1, -1]$ in the 1D case and normalize. This weight vector can be viewed as an edge detection filter, that is, counting the number of times image goes from black (negative) to white (positive). Figure 3 (Top) shows that SGD has a small data dependent range where it succeeds but may fail with almost 0.5 probability outside this region whereas Convotron always returns a good solution for small enough η chosen according to Theorem 2. The failure points observed for SGD show the prevalence of bad local minima where SGD gets stuck.

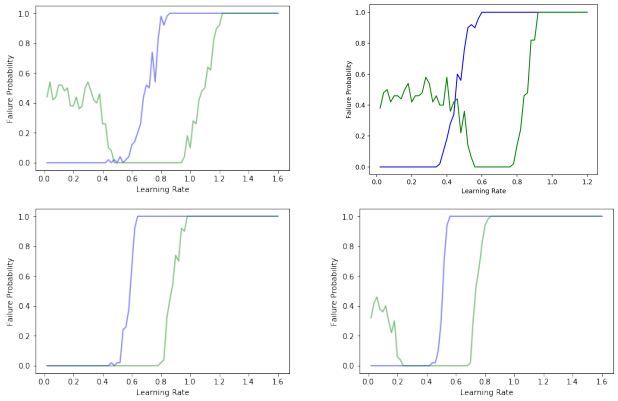

Figure 3. Failure probability of SGD (green) vs Convotron (blue) with varying learning rate η. Experiment 1: Patch and stride 1D (Top-left) and 2D (Top-right). Experiment 2: Input distribution has mean 0 and covariance matrix identity (Bottom-left) and non-identity covariance matrix (Bottom-right). The curves are shifted due to scaling difference of updates.

For the second experiment, we choose a fixed weight vector for which SGD performs well with very high probability on a normalized Gaussian input distribution with identity covariance matrix (see Figure 3 (Bottom-left)). However, on choosing a different covariance matrix with higher condition number ~ 60, the performance of SGD worsens whereas Convotron always succeeds (see Figure 3 (Bottom-Right)). The covariance matrix is generated by choosing random matrices followed by symmetrizing them and adding cI for $c > 0$ to make the eigenvalues positive.

These experiments demonstrate that techniques for fine-tuning SGD's learning rate are necessary, even for very simple architectures. In contrast, no fine-tuning is necessary for Convotron: the correct learning rate can be easily computed given the learner's desired patch structure and estimate of the covariance martix.

7. Conclusions and Future Work

We have given the first efficient algorithm with provable guarantees for learning general one layer convolutional networks under symmetric, well-conditioned distributions. The obvious open question is to extend our algorithm to higher depth networks and weaken the distributional assumptions.

Acknowledgments

We thank Jessica Hoffmann and Philipp Krähenbühl for useful discussions.

References

Brutzkus, A. and Globerson, A. Globally optimal gradient descent for a convnet with gaussian inputs. *arXiv preprint arXiv:1702.07966*, 2017.

Du, S. S., Lee, J. D., and Tian, Y. When is a convolutional filter easy to learn? *arXiv preprint arXiv:1709.06129*, 2017a.

Du, S. S., Lee, J. D., Tian, Y., Poczos, B., and Singh, A. Gradient descent learns one-hidden-layer cnn: Don't be afraid of spurious local minima. *arXiv preprint arXiv:1712.00779*, 2017b.

Ge, R., Lee, J., and Ma, T. Learning one-hidden-layer neural networks with landscape design. *arXiv preprint arXiv:1711.00501*, 2017.

Goel, S. and Klivans, A. Eigenvalue decay implies polynomial-time learnability for neural networks. In *Advances in Neural Information Processing Systems*, pp. 2189–2199, 2017a.

Goel, S. and Klivans, A. Learning depth-three neural networks in polynomial time. *arXiv preprint arXiv:1709.06010*, 2017b.

Goel, S., Kanade, V., Klivans, A., and Thaler, J. Reliably learning the relu in polynomial time. *arXiv preprint arXiv:1611.10258*, 2016.

Kakade, S. M., Kanade, V., Shamir, O., and Kalai, A. Efficient learning of generalized linear and single index models with isotonic regression. In *Advances in Neural Information Processing Systems*, pp. 927–935, 2011.

Kearns, M. J. and Schapire, R. E. Efficient distribution-free learning of probabilistic concepts. In *Foundations of Computer Science, 1990. Proceedings., 31st Annual Symposium on*, pp. 382–391. IEEE, 1990.

Klivans, A. and Meka, R. Learning graphical models using multiplicative weights. *arXiv preprint arXiv:1706.06274*, 2017.

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.

Soltanolkotabi, M. Learning relus via gradient descent. In *Advances in Neural Information Processing Systems*, pp. 2004–2014, 2017.

Tian, Y. Symmetry-breaking convergence analysis of certain two-layered neural networks with relu nonlinearity. 2016.

Weisstein, E. W. Gershgorin circle theorem. 2003.

Zhang, Q., Panigrahy, R., Sachdeva, S., and Rahimi, A. Electron-proton dynamics in deep learning. *arXiv preprint arXiv:1702.00458*, 2017.

Zhong, K., Song, Z., and Dhillon, I. S. Learning non-overlapping convolutional neural networks with multiple kernels. *arXiv preprint arXiv:1711.03440*, 2017a.

Zhong, K., Song, Z., Jain, P., Bartlett, P. L., and Dhillon, I. S. Recovery guarantees for one-hidden-layer neural networks. *arXiv preprint arXiv:1706.03175*, 2017b.