Globally Optimal Gradient Descent for a ConvNet with Gaussian Inputs

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

Abstract

Deep learning models are often successfully trained using gradient descent, despite the worst case hardness of the underlying non-convex optimization problem. The key question is then under what conditions can one prove that optimization will succeed. Here we provide a strong result of this kind. We consider a neural net with one hidden layer and a convolutional structure with no overlap, and a ReLU activation function. For this architecture we show that learning is NP-complete in the general case, but that when the input distribution is Gaussian, gradient descent converges to the global optimum in polynomial time. To the best of our knowledge, this is the first global optimality guarantee of gradient descent on a convolutional neural network with ReLU activations.

1. Introduction

Deep neural networks have achieved state-of-the-art performance on many machine learning tasks in areas such as natural language processing (Wu et al., 2016), computer vision (Krizhevsky et al., 2012) and speech recognition (Hinton et al., 2012). Training of such networks is often successfully performed by minimizing a high-dimensional non-convex objective function, using simple first-order methods such as stochastic gradient descent.

Nonetheless, the success of deep learning from an optimization perspective is poorly understood theoretically. Current results are mostly pessimistic, suggesting that even training a 3-node neural network is NP-hard (Blum & Rivest, 1993), and that the objective function of a single neuron can admit exponentially many local minima (Auer et al., 1996; Safran & Shamir, 2016). There have been recent attempts to bridge this gap between theory and practice. Several works focus on the geometric properties of loss functions that neural networks attempt to minimize. For some simplified architectures, such as linear activations, it can be shown that there are no bad local minima (Kawaguchi, 2016). Extension of these results to the non-linear case currently requires very strong independence assumptions between the activations of the neurons and the inputs (Kawaguchi, 2016).

Since gradient descent is the main "work-horse" of deep learning it is of key interest to understand its convergence properties. However, there are no results showing that gradient descent is globally optimal for non-linear models, except for the case of many hidden neurons (Andoni et al., 2014) and non-linear activation functions that are not widely used in practice (Zhang et al., 2017).[1] Here we provide the first such result for a neural architecture that has two very common components: namely a ReLU activation function and a convolution layer.

The architecture considered in the current paper is shown in Figure 1. We refer to these models as *no-overlap* networks. A no-overlap network can be viewed as a simple convolution layer with non overlapping filters, followed by a ReLU activation function, and then average pooling. Formally, let $\boldsymbol{w} \in \mathbb{R}^m$ denote the filter coefficients, and assume the input $\boldsymbol{x}$ is in $\mathbb{R}^d$. Define $k = d/m$ and assume that k is integral. Partition $\boldsymbol{x}$ into k non-overlapping parts and denote $\boldsymbol{x}[i]$ the i^{th} part. Finally, define σ to be the ReLU activation function, namely $\sigma(z) = \max\{0, z\}$. Then the output of the network in Figure 1 is given by:

$$f(\boldsymbol{x}; \boldsymbol{w}) = \frac{1}{k} \sum_i \sigma(\boldsymbol{w} \cdot \boldsymbol{x}[i]) \tag{1}$$

We note that such architectures have been used in several works (Lin et al., 2013; Milletari et al., 2016), but we view them as important firstly because they capture key properties of general convolutional networks.

We address the realizable case, where training data is generated from a function as in Eq. 1 with weight vector $\boldsymbol{w}^*$. Training data is then generated by sampling n training points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$ from a distribution $\mathcal{D}$, and assigning them labels using $y = f(\boldsymbol{x}; \boldsymbol{w}^*)$. The learning problem is

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

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

[1]See more related work in Section 2.

then to find a $\boldsymbol{w}$ that minimizes the squared loss. In other words, solve the optimization problem:

$$\min_{\boldsymbol{w}} \frac{1}{n} \sum_i \left(f(\boldsymbol{x}_i; \boldsymbol{w}) - y_i\right)^2 \tag{2}$$

In the limit $n \to \infty$, this is equivalent to minimizing the population risk:

$$\ell(\boldsymbol{w}) = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}} \left[\left(f(\boldsymbol{x}; \boldsymbol{w}) - f(\boldsymbol{x}; \boldsymbol{w}^*)\right)^2\right] \tag{3}$$

Like several recent works (Hardt et al., 2016; Hardt & Ma, 2016) we focus on minimizing the population risk, leaving the finite sample case to future work. We believe the population risk captures the key characteristics of the problem, since the large data regime is the one of interest.

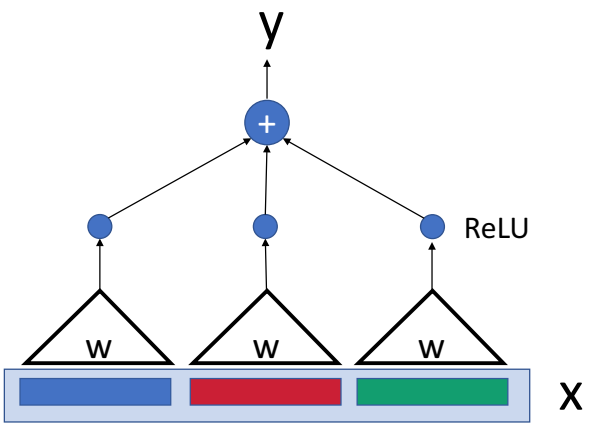

Figure 1. Convolutional neural network with non-overlapping filters. In the first layer, a filter $\boldsymbol{w}$ is applied to non-overlapping parts of the input vector $\boldsymbol{x}$, and the output passes through a ReLU activation function. The outputs of the neurons are then averaged to give the output y.

Our key results are as follows:

- **Worst Case Hardness:** Despite the simplicity of *No-Overlap Networks*, we show that learning them is in fact hard if $\mathcal{D}$ is unconstrained. Specifically, in Section 4, we show that learning *No-Overlap Networks* is NP complete via a reduction from a variant of the set splitting problem.
- **Distribution Dependent Tractability:** When $\mathcal{D}$ corresponds to independent Gaussian variables with $\mu = 0, \sigma^2 = 1$, we show in Section 5 that *No-Overlap Networks* can be learned in polynomial time using gradient descent.

The above two results nicely demonstrate the gap between worst-case intractability and tractability under assumptions on the data. We provide an empirical demonstration of this in Section 6 where gradient descent is shown to succeed in the Gaussian case and fail for a different distribution.

To further understand the role of overlap in the network, we consider networks that do have overlap between the filters. In Section 7.1 we show that in this case, even under Gaussian distributed inputs, there will be non-optimal local minima. Thus, gradient descent will no longer be optimal in the overlap case. In Section 7.2 we show empirically that these local optima may be overcome in practice by using gradient descent with multiple restarts.

Taken together, our results are the first to demonstrate distribution dependent optimality of gradient descent for learning a neural network with a convolutional like architecture and a ReLU activation function.

2. Related Work

Hardness of learning neural networks has been demonstrated for many different settings. For example, Blum & Rivest (1993) show that learning a neural network with one hidden layer with a sign activation function is NP-hard in the realizable case. Livni et al. (2014) extend this to other activation functions and bounded norm optimization. Hardness can also be shown for improper learning under certain cryptographic assumptions (e.g., see Daniely et al., 2014; Klivans, 2008; Livni et al., 2014). Note that these hardness results do not hold for the regression and tied parameter setting that we consider.

Due to the above hardness results, it is clear that the success of deep-learning can only be explained by making additional assumptions about the data generating distribution. The classic algorithm by Baum (1990) shows that intersection of halfspaces (i.e., a specific instance of a one hidden layer network) is PAC learnable under any symmetric distribution. This was later extended in Klivans et al. (2009) to log-concave distributions.

The above works do not consider gradient descent as the optimization method, leaving open the question of which assumptions can lead to global optimality of gradient descent. Such results have been hard to obtain, and we survey some recent ones below. One instance when gradient descent can succeed is when there are enough hidden units such that random initialization of the first layer can lead to zero error even if only the second layer is trained. Such over-specified networks have been considered (Andoni et al., 2014; Livni et al., 2014) and it was shown that gradient descent can globally learn them in some cases (Andoni et al., 2014). However, the assumption of over-specification is very restrictive and limits generalization. In contrast, we show convergence of gradient descent to a global optimum for any network size and consider convo-

lutional neural networks with shared parameters. Another interesting case is linear dynamical systems, where Hardt et al. (2016) show that under independence assumptions maximum likelihood is quasi-concave and hence solvable with gradient ascent.

Recent work by Mei et al. (2016) shows that regression with a *single neuron* and certain non-linear activation functions, can be learned with gradient descent for sub-Gaussian inputs. We note that their architecture is significantly simpler than ours, in that it uses a single neuron. In fact, their regression problem can also be solved via methods for generalized linear models (Kakade et al., 2011).

Shamir (2016) recently showed that there is a limit to what distribution dependent results can achieve. Namely, it was shown that for large enough one-hidden layer networks, no distributional assumption such as Gaussian inputs can make gradient descent tractable. Importantly, the construction in Shamir (2016) does not use parameter tying and thus is not applicable to the architecture we study here.

Several works have focused on understanding the loss surface of neural network objectives, but without direct algorithmic implications. Kawaguchi (2016) show that *linear* neural networks do not suffer from bad local minima. Hardt & Ma (2016) consider objectives of *linear residual* networks and prove that there are no critical points other than the global optimum. Soudry & Carmon (2016) show that in the objective of over-parameterized neural networks with dropout-like noise, all differentiable local minima are global. Other works (Safran & Shamir, 2016; Haeffele & Vidal, 2015) give similar results for over-specified networks. All of these results are purely geometric and do not have direct implications on convergence of optimization algorithms. Janzamin et al. (2015) and Goel et al. (2016), suggest alternatives to gradient-based methods for learning neural networks. However, these algorithms are not widely used in practice. Finally, Choromanska et al. (2015) use spin glass models to argue that, under certain generative modelling and architectural constraints, local minima are likely to have low loss values.

The theory of non-convex optimization is closely related to the theory of neural networks. Recently, there has been substantial progress in proving convergence guarantees of simple first-order methods in various machine learning problems, that don't correspond to typical neural nets. These include for example matrix completion (Ge et al., 2016) and tensor decompositions (Ge et al., 2015).

Finally, recent work by Zhang et al. (2016) shows that neural nets can perfectly fit random labelings of the data. Understanding this from an optimization perspective is largely an open problem.

3. Preliminaries

We use bold-faced letters for vectors and capital letters for matrices. The i^{th} row of a matrix A is denoted by $\mathbf{a}_i$.

In our analysis in Section 5 and Section 7.1 we assume that the input feature $\boldsymbol{x} \in \mathbb{R}^d$ is a vector of IID Gaussian random variables with zero mean and variance one.[2] Denote this distribution by $\mathcal{G}$. We consider networks with one hidden layer, and k hidden units. Our main focus will be on *No-Overlap Networks*, but we begin with a more general one-hidden-layer neural network with a fully-connected layer parameterized by $W \in \mathbb{R}^{k,d}$ followed by average pooling. The network output is then:

$$f(\boldsymbol{x}; W) = \frac{1}{k} \sum_i \sigma\left(\boldsymbol{w}_i \cdot \boldsymbol{x}\right) \tag{4}$$

where $\sigma\left(\right)$ is the pointwise ReLU function.

We consider the realizable setting where there exists a *true* W^* using which the training data is generated. The population risk (see Eq. 3) is then:

$$\ell(W) = \mathbb{E}_{\mathcal{G}}\left[\left(f(\boldsymbol{x}; W) - f(\boldsymbol{x}; W^*)\right)^2\right] , \tag{5}$$

As we show next, $\ell(W)$ can be considerably simplified. First, define:

$$g(\boldsymbol{u}, \boldsymbol{v}) = \mathbb{E}_{\mathcal{G}}\left[\sigma\left(\boldsymbol{u} \cdot \boldsymbol{x}\right) \sigma\left(\boldsymbol{v} \cdot \boldsymbol{x}\right)\right] \tag{6}$$

Simple algebra then shows that:

$$\ell(W) = \frac{1}{k^2} \sum_{i,j} \left[g(\boldsymbol{w}_i, \boldsymbol{w}_j) - 2g(\boldsymbol{w}_i, \boldsymbol{w}_j^*) + g(\boldsymbol{w}_i^*, \boldsymbol{w}_j^*)\right] \tag{7}$$

The next Lemma from Cho & Saul (2009) shows that $g(\boldsymbol{u}, \boldsymbol{v})$ has a simple form.

Lemma 3.1. *(Cho & Saul, 2009, Section 2) Given $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d$ denote by $\theta_{\boldsymbol{u},\boldsymbol{v}}$ the angle between $\boldsymbol{u}$ and $\boldsymbol{v}$. Then:*

$$g(\boldsymbol{u}, \boldsymbol{v}) = \frac{1}{2\pi} \left\|\boldsymbol{u}\right\| \left\|\boldsymbol{v}\right\| \left(\sin\theta_{\boldsymbol{u},\boldsymbol{v}} + \left(\pi - \theta_{\boldsymbol{u},\boldsymbol{v}}\right) \cos\theta_{\boldsymbol{u},\boldsymbol{v}} \right)$$

The gradient of g with respect to $\boldsymbol{u}$ also turns out to have a simple form, as stated in the lemma below. The proof is deferred to the supplementary material.

Lemma 3.2. *Let g be as defined in Eq. 6. Then g is differentiable at all points $\boldsymbol{u} \neq \mathbf{0}$ and*

$$\frac{\partial g(\boldsymbol{u}, \boldsymbol{v})}{\partial \boldsymbol{u}} = \frac{1}{2\pi} \left\|\boldsymbol{v}\right\| \frac{\boldsymbol{u}}{\left\|\boldsymbol{u}\right\|} \sin\theta_{\boldsymbol{u},\boldsymbol{v}} + \frac{1}{2\pi}\left(\pi - \theta_{\boldsymbol{u},\boldsymbol{v}}\right)\boldsymbol{v}$$

[2] The variance per variable can be arbitrary. We choose one for simplicity.

We conclude by special-casing the results above to *No-Overlap Networks*. In this case, the entire model is specified by a single *filter* vector $\boldsymbol{w} \in \mathbb{R}^m$. The rows $\boldsymbol{w}_i$ are mostly zeros, except for the indices $((i-1)m+1,\ldots,im)$ which take the values of $\boldsymbol{w}$. Namely, $\boldsymbol{w}_i = \left(\mathbf{0}_{(i-1)m}, \boldsymbol{w}, \mathbf{0}_{d-im}\right)$ where $\mathbf{0}_l \in \mathbb{R}^l$ is a zero vector. The same holds for the vectors $\boldsymbol{w}_i^*$ with a weight vector $\boldsymbol{w}^*$. This simplifies the loss considerably, since for all i: $g(\boldsymbol{w}_i, \boldsymbol{w}_i) = \frac{1}{2}\|\boldsymbol{w}\|^2$, and for all $i \neq j$: $g(\boldsymbol{w}_i, \boldsymbol{w}_j) = \frac{1}{2\pi}\|\boldsymbol{w}\|^2$ and $g(\boldsymbol{w}_i, \boldsymbol{w}_j^*) = \frac{1}{2\pi}\|\boldsymbol{w}\|\|\boldsymbol{w}^*\|$. Thus the loss $\ell(\boldsymbol{w})$ for *No-Overlap Networks* yields (up to additive factors in $\boldsymbol{w}^*$):

$$\ell(\boldsymbol{w}) = \frac{1}{k^2}\left[\gamma\|\boldsymbol{w}\|^2 - 2kg(\boldsymbol{w}, \boldsymbol{w}^*) - 2\beta\|\boldsymbol{w}\|\|\boldsymbol{w}^*\|\right] \quad (8)$$

where $\beta = \frac{k^2-k}{2\pi}$ and $\gamma = \beta + \frac{k}{2}$.

4. Learning *No-Overlap Networks* is NP-Complete

The *No-Overlap Networks* architecture is a simplified convolutional layer with average pooling. However, as we show here, learning it is still a hard problem. This will motivate our exploration of distribution dependent results in Section 5.

Recall that our focus is on minimizing the squared error in Eq. 3. For this section, we do not make any assumptions about $\mathcal{D}$. Thus $\mathcal{D}$ can be a distribution with uniform mass on training points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, recovering the empirical risk in Eq. 2. We know that $\ell(\boldsymbol{w})$ in Eq. 3 can be minimized by setting $\boldsymbol{w} = \boldsymbol{w}^*$ and the corresponding squared loss $\ell(\boldsymbol{w})$ will be zero. However, we of course do not know $\boldsymbol{w}^*$, and the question is how difficult is it to minimize $\ell(\boldsymbol{w})$. In what follows we show that this is hard. Namely, it is an NP-complete problem to find a $\boldsymbol{w}$ that comes ϵ_0 close to the minimum of $\ell(\boldsymbol{w})$, for some constant ϵ_0.

We begin by defining the *Set-Splitting-by-k-Sets* problem, which is a variant of the classic Set-Splitting problem (Garey & Johnson, 1990). After establishing the hardness of *Set-Splitting-by-k-Sets*, we will provide a reduction from it to learning *No-Overlap Networks*.

Definition 1. *The Set-Splitting-by-k-Sets decision problem is defined as follows: Given a finite set S of d elements and a collection $\mathcal{C}$ of at most $(k-1)d$ subsets C_j of S, do there exist disjoint sets $S_1, S_2, ..., S_k$ such that $\bigcup_i S_i = S$ and for all j and i, $C_j \not\subseteq S_i$?*

For $k = 2$ and without the upper bound on $|\mathcal{C}|$ this is known as the *Set-Splitting* decision problem which is NP-complete (Garey & Johnson, 1990). Next, we show that *Set-Splitting-by-k-Sets* is NP-complete. The proof is via a reduction from 3SAT and induction, and is provided in the supplementary material.

Proposition 4.1. *Set-Splitting-by-k-Sets is NP-complete for all $k \geq 2$.*

We next formulate the *No-Overlap Networks* optimization problem.

Definition 2. *The k-Non-Overlap-Opt problem is defined as follows. The input is a distribution $\mathcal{D}_{X,Y}$ over input-output pairs $\boldsymbol{x}, y$ where $\boldsymbol{x} \in \mathbb{R}^d$. If the input is realizable by a no-overlap network with k hidden neurons, then the output is a vector $\boldsymbol{w}$ such that:*

$$\mathbb{E}_{\mathcal{D}_{X,Y}}\left[\left(f(\boldsymbol{x}; \boldsymbol{w}) - y\right)\right)^2\right] < \frac{1}{4k^5 d} \quad (9)$$

Otherwise an arbitrary weight vector is returned.[3]

The above problem returns a $\boldsymbol{w}$ that minimizes the population-risk up to $\frac{1}{4k^5 d}$ accuracy. It is thus easier than minimizing the risk to an arbitrary precision ϵ (see Section 5, Theorem 5.2).

We prove the following theorem, which uses some ideas from Blum & Rivest (1993), but introduces additional constructions needed for the no overlap case.

Theorem 4.2. *For all $k \geq 2$, the k-Non-Overlap-Opt problem is NP-complete.*

Proof. We will show a reduction from *Set-Splitting-by-k-sets* to *k-Non-Overlap-Opt*. Assume a given instance of the *Set-Splitting-by-k-sets* problem with a set S and collection of subsets $\mathcal{C}$. Denote $S = \{1, 2, ..., d\}$ and $|\mathcal{C}| \leq (k-1)d$. Let $\mathbf{0}_d \in \mathbb{R}^d$ be the all zeros vector. For a vector $\boldsymbol{v} \in \mathbb{R}^d$, define the vector $\boldsymbol{d}_i(\boldsymbol{v}) \in \mathbb{R}^{kd}$ to be the concatenation of $i-1$ vectors $\mathbf{0}_d$, followed by $\boldsymbol{v}$ and $k-i$ vectors $\mathbf{0}_d$, and let $\boldsymbol{d}(\boldsymbol{v}) = (\boldsymbol{d}_1(\boldsymbol{v}), \boldsymbol{d}_2(\boldsymbol{v}), ..., \boldsymbol{d}_k(\boldsymbol{v})) \in \mathbb{R}^{k^2 d}$.

We next define a training set for *k-Non-Overlap-Opt*. For each element $i \in S$ define an input vector $\boldsymbol{x}_i = \boldsymbol{d}(\boldsymbol{e}_i)$, where $\boldsymbol{e}_i$ is the standard basis of $\mathbb{R}^d$. Assign the label $y_i = \frac{1}{k}$ to this input. In addition, for each subset $C_j \in \mathcal{C}$ define the vector $\boldsymbol{x}_{d+j} = \boldsymbol{d}(\sum_{i \in C_j} \boldsymbol{e}_i)$ and label $y_{d+j} = 0$. Thus we have $|S| + |\mathcal{C}|$ data points in $\mathbb{R}^{k^2 d}$. Let $\mathcal{D}_{X,Y}$ be a uniform distribution over the training set points (i.e., each point with probability at least $\frac{1}{kd}$ since $|\mathcal{C}| \leq (k-1)d$).

We will now show that the given instance of *Set-Splitting-by-k-sets* has a solution (i.e., there exist splitting sets) if and only if *k-Non-Overlap-Opt* returns a weight vector with low risk. First, assume there exist splitting sets $S_1, ..., S_k$.[4] For each $1 \leq l \leq k$ define the vector $\boldsymbol{a}^{S_l} \in \mathbb{R}^d$ such that for all $i \in S_l$, $a_i^{S_l} = 1$ and $a_i^{S_l} = -d$ otherwise. Define a *No-Overlap Network* with $k^2 d$ inputs and weight vector

[3] We assume that the population risk is efficiently computable.

[4] The sets are disjoint, their union is S and for all j and i, $C_j \not\subseteq S_i$.

$\boldsymbol{w} = (\boldsymbol{a}^{S_1}, \boldsymbol{a}^{S_2}, ..., \boldsymbol{a}^{S_k}) \in \mathbb{R}^{kd}$. Then for all $1 \leq i \leq d$ we have:

$$f(\boldsymbol{x}_i; \boldsymbol{w}) = \frac{\sum_{l=1}^{k} \sigma((\boldsymbol{a}^{S_l})^T \boldsymbol{e}_i)}{k} = \frac{1}{k} = y_i \tag{10}$$

and for all j:

$$f(\boldsymbol{x}_{d+j}; \boldsymbol{w}) = \frac{\sum_{l=1}^{k} \sigma((\mathbf{a}^{S_l})^T (\sum_{i \in C_j} \boldsymbol{e}_i))}{k} = 0 = y_{d+j} \tag{11}$$

where the last equality follows since for all l and j, $C_j \nsubseteq S_l$. There thus exists a $\boldsymbol{w}$ for which the error in Eq. 9 is zero and *k-Non-Overlap-Opt* will return a weight vector with low risk.

Conversely, assume that *k-Non-Overlap-Opt* returned a $\boldsymbol{w} \in \mathbb{R}^{kd}$ with risk less than $\frac{1}{4k^5 d}$ on $\mathcal{D}_{X,Y}$ above. Denote by $\boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2, ..., \boldsymbol{w}_k)$, where $\boldsymbol{w}_l \in \mathbb{R}^d$. We will show that this implies that there exist k splitting sets. For all $\boldsymbol{x}', y'$ in the training set it holds that:[5]

$$\frac{\left(f(\boldsymbol{x}'; \boldsymbol{w}) - y'\right)^2}{kd} \leq \mathbb{E}_{\mathcal{D}_{X,Y}}[(f(\boldsymbol{x}; \boldsymbol{w}) - y)^2] < \frac{1}{4k^5 d}$$

This implies that for all i and j,

$$|f(\boldsymbol{d}(\boldsymbol{e}_i); \boldsymbol{w}) - \frac{1}{k}| < \frac{1}{2k^2} \;\;,\;\; |f(\boldsymbol{d}(\sum_{i \in C_j} \boldsymbol{e}_i); \boldsymbol{w})| < \frac{1}{2k^2} \tag{12}$$

Define sets $S_l = \{i \mid \boldsymbol{w}_l^T \boldsymbol{e}_i > \frac{1}{2k}\}$ for $1 \leq l \leq k$ and WLOG assume they are disjoint by arbitrarily assigning points that belong to more than one set, to one of the sets they belong to. We will next show that these S_l are splitting. Namely, it holds that $\bigcup_l S_l = S$ and no subset C_j is a subset of some S_l.

Since $f(\boldsymbol{d}(\boldsymbol{e}_i); \boldsymbol{w}) = \frac{\sum_{l=1}^{k} \sigma(\boldsymbol{w}_l^T \boldsymbol{e}_i)}{k} > \frac{1}{k} - \frac{1}{2k^2} > \frac{1}{2k}$ for all i, it follows that for each $i \in S$ there exists $1 \leq l \leq k$ such that $\boldsymbol{w}_l^T \boldsymbol{e}_i > \frac{1}{2k}$. Therefore, by the definition of S_l we deduce that $\bigcup_l S_l = S$. To show the second property, assume by contradiction that for some j and m, $C_j \subseteq S_m$. Then $\boldsymbol{w}_m^T (\sum_{i \in C_j} \boldsymbol{e}_i) > \frac{|C_j|}{2k}$, which implies that $f(\boldsymbol{d}(\sum_{i \in C_j} \boldsymbol{e}_i); \boldsymbol{w}) = \frac{\sum_{l=1}^{k} \sigma(\boldsymbol{w}_l^T (\sum_{i \in C_j} \boldsymbol{e}_i))}{k} > \frac{|C_j|}{2k^2} \geq \frac{1}{2k^2}$, a contradiction. This concludes our proof. □

To conclude, we have shown that *No-Overlap Networks* are hard to learn if one does not make any assumptions about the training data. In fact we have shown that finding a $\boldsymbol{w}$ with loss at most $\frac{1}{4k^5 d}$ is hard. In the next section, we show that certain distributional assumptions make the problem tractable.

[5] The LHS is true because for a non-negative random variable X, $E[X] \geq p(x)x$ for all x, and in our case $p(x) \geq \frac{1}{kd}$.

5. *No-Overlap Networks* can be Learned for Gaussian Inputs

In this section we assume that the input features $\boldsymbol{x}$ are generated via a Gaussian distribution $\mathcal{G}$, as in Section 3. We will show that in this case, gradient descent will converge with high probability to the global optimum of $\ell(\boldsymbol{w})$ (Eq. 8) in polynomial time.

In order to analyze convergence of gradient descent on ℓ, we need a characterization of all the critical and non-differentiable points. We show that ℓ has a non-differentiable point and a degenerate saddle point.[6] Therefore, recent methods for showing global convergence of gradient-based optimizers on non-convex objectives (Lee et al., 2016; Ge et al., 2015) cannot be used in our case, because they assume all saddles are strict [7] and the objective function is continuously differentiable everywhere.

The characterization is given in the following proposition. The proof relies on the fact that $\ell(\boldsymbol{w})$ depends only on $\|\boldsymbol{w}\|$,$\|\boldsymbol{w}^*\|$ and $\theta_{\boldsymbol{w},\boldsymbol{w}^*}$, and therefore w.l.o.g. it can be assumed that $\boldsymbol{w}^*$ lies on one of the axes. Then by a symmetry argument, in order to prove properties of the gradient and the Hessian, it suffices to calculate partial derivatives with respect to at most three variables.

Proposition 5.1. *Let $\ell(\boldsymbol{w})$ be defined as in Eq. 8. Then the following holds:*

1. *$\ell(\boldsymbol{w})$ is differentiable if and only if $\boldsymbol{w} \neq \mathbf{0}$.*
2. *For $k > 1$, $\ell(\boldsymbol{w})$ has three critical points:*
   (a) *A local maximum at $\boldsymbol{w} = \mathbf{0}$.*
   (b) *A unique global minimum at $\boldsymbol{w} = \boldsymbol{w}^*$.*
   (c) *A degenerate saddle point at $\boldsymbol{w} = -(\frac{k^2-k}{k^2+(\pi-1)k})\boldsymbol{w}^*$.*

   For $k = 1$, $\boldsymbol{w} = \mathbf{0}$ is not a local maximum and the unique global minimum $\boldsymbol{w}^*$ is the only differentiable critical point [8].

We next consider a simple gradient descent update rule for minimizing $\ell(\boldsymbol{w})$ and analyze its convergence. Let $\lambda > 0$ denote the step size. Then the update at iteration t is simply:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \lambda \nabla \ell(\boldsymbol{w}_t) \tag{13}$$

Our main result, stated formally below, is that the above update is guaranteed to converge to an ϵ accurate solution after $O(\frac{1}{\epsilon^2})$ iterations. We note that the dependence of the

[6] A saddle point is degenerate if the Hessian at the point has only non-negative eigenvalues and at least one zero eigenvalue.

[7] A saddle point is strict if the Hessian at the point has at least one negative eigenvalue.

[8] See Figure 2.

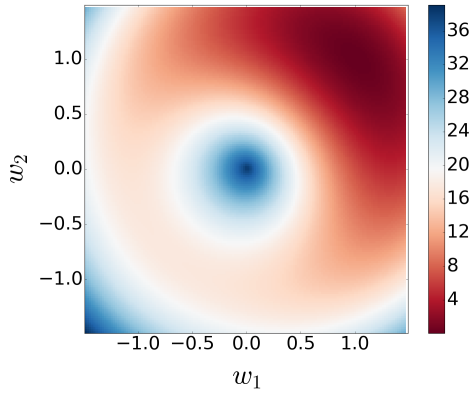

Figure 2. Colormap of $\ell(\boldsymbol{w})$ (Eq. 8) in 2 dimensions for $\boldsymbol{w}^* = (1, 1)$ and $k = 10$.

convergence rate on ϵ is similar to standard results on convergence of gradient descent to stationary points (e.g., see discussion in Allen-Zhu & Hazan, 2016).

Theorem 5.2. *Assume* $\|\boldsymbol{w}^*\| = 1$.[9] *For any* $\delta > 0$ *and* $0 < \epsilon < \frac{\delta \sin \pi\delta}{k}$, *there exists* $0 < \lambda < 1$ [10] *such that with probability at least* $1 - \delta$, *gradient descent initialized randomly from the unit sphere with learning rate* λ *will get to a point* $\boldsymbol{w}$ *such that* $\ell(\boldsymbol{w}) \leq O(\epsilon)$ *in* $O(\frac{1}{\epsilon^2})$ *iterations.*[11]

The complete proof is provided in the supplementary material. Here we provide a high level overview. In particular, we first explain why gradient descent will stay away from the two *bad* points mentioned in Lemma 5.1.

First we note that the gradient of $\ell(\boldsymbol{w})$ at $\boldsymbol{w}_t$ is given by:

$$\nabla \ell(\boldsymbol{w}_t) = -c_1(\boldsymbol{w}_t, \boldsymbol{w}^*)\boldsymbol{w}_t - c_2(\boldsymbol{w}_t, \boldsymbol{w}^*)\boldsymbol{w}^* , \quad (14)$$

where c_1 and c_2 are two functions such that $c_1 \geq -1$, $c_2 \geq 0$ and $c_2 = 0$ if and only if $\theta_{\boldsymbol{w}_t, \boldsymbol{w}^*} = \pi$. Thus the gradient is a sum of a vector in the direction of $\boldsymbol{w}_t$ and a vector in the direction of $\boldsymbol{w}^*$. At iteration $t + 1$ we have:

$$\boldsymbol{w}_{t+1} = (1 + \lambda c_1(\boldsymbol{w}_t, \boldsymbol{w}^*))\boldsymbol{w}_t + \lambda c_2(\boldsymbol{w}_t, \boldsymbol{w}^*)\boldsymbol{w}^* \quad (15)$$

It follows that for $\lambda < 1$ and $\theta_{\boldsymbol{w}_t, \boldsymbol{w}^*} \neq \pi$, we have $\theta_{\boldsymbol{w}_{t+1}, \boldsymbol{w}^*} < \theta_{\boldsymbol{w}_t, \boldsymbol{w}^*}$. Therefore, if $\theta_{\boldsymbol{w}_0, \boldsymbol{w}^*} \neq \pi$, we will never converge to the saddle point in Lemma 5.1.

Next, assuming $\|\boldsymbol{w}_0\| > 0$ and that $\theta_{\boldsymbol{w}_0, \boldsymbol{w}^*} \leq (1 - \delta)\pi$ (which occurs with probability $1 - \delta$), it can be shown that the norm of $\boldsymbol{w}_t$ is always bounded away from zero by a constant $M = \tilde{\Omega}(1)$.[12] The proof is quite technical and follows from the fact that $\boldsymbol{w} = \boldsymbol{0}$ is a local maximum.[13]

The fact that $\boldsymbol{w}_t$ stays away from the *problematic* points allows us to show that $\ell(\boldsymbol{w})$ has a Lipschitz continuous gradient on the line between $\boldsymbol{w}_t$ and $\boldsymbol{w}_{t+1}$, with constant $L = \tilde{O}(1)$.[12] By standard optimization analysis (Nesterov, 2004) it follows that after $T = O(\frac{1}{\epsilon^2})$ iterations we will have $\|\nabla l(\boldsymbol{w}_t)\| \leq O(\epsilon)$ for some $0 \leq t \leq T$. This in turn can be used to show that $\boldsymbol{w}_t$ is $O(\sqrt{\epsilon})$-close to $\boldsymbol{w}^*$. Finally, since $\ell(\boldsymbol{w}) \leq d\|\boldsymbol{w} - \boldsymbol{w}^*\|^2$, it follows that $\boldsymbol{w}_t$ approximates the global minimum to within $O(\epsilon)$ accuracy.

Theorem 5.2 implies that gradient descent converges to a point $\boldsymbol{w}$ such that $\ell(\boldsymbol{w}) \leq \frac{1}{d^2}$ in time $O(poly(d))$ where d is the input dimension.[14] The following corollary thus follows.

Corollary 5.3. *Gradient descent solves the k-Non-Overlap-Opt problem under the Gaussian assumption on* $\mathcal{D}$ *with high probability and in polynomial time.*

6. Empirical Illustration of Tractability Gap

The results in the previous sections showed that *No-Overlap Networks* optimization is hard in the general case, but tractable for Gaussian inputs. Here we empirically demonstrate both the easy and hard cases. The training data for the two cases will be generated by using the same $\boldsymbol{w}^*$ but different distributions over $\boldsymbol{x}$.

To generate the "hard" case, we begin with a set splitting problem. In particular, we consider a set S with 40 elements and a collection $\mathcal{C}$ of 760 subsets of S, each of size 20. We choose C_j such that there exists subsets S_1,S_2 that split the subsets C_j. We use the reduction in Section 4 to convert this into a *No-Overlap Networks* optimization problem. This results in a training set of size 800.

Since we know the $\boldsymbol{w}^*$ that solves the set splitting problem, we can use it to label data from a different distribution. Motivated by Section 5 we use a Gaussian distribution $\mathcal{G}$ as defined earlier and generate a training set of the same size (namely 800) and labels given by the *no-overlap* network with weight $\boldsymbol{w}^*$.

For these two learning problems we used AdaGrad (Duchi et al., 2011) to optimize the *empirical* risk (plain gradient descent also converges, but AdaGrad requires less tuning of step size). For both datasets we used a random normal initializer and for each we chose the best performing learning rate schedule. The training error for each setting as a function of the number of epochs is shown in Figure 3. It is clear that in the non-Gaussian case, AdaGrad gets trapped

[9] Assumed for simplicity, otherwise $\|\boldsymbol{w}^*\|$ is a constant factor.
[10] λ can be found explicitly.
[11] $O(\cdot)$ hides a linear factor in d.
[12] $\tilde{\Omega}$ and $\tilde{O}$ hide factors of $\|\boldsymbol{w}^*\|$, $\theta_{\boldsymbol{w}_0, \boldsymbol{w}^*}$, k and δ.
[13] The proof holds even for $k = 1$ where $\boldsymbol{w} = \boldsymbol{0}$ is not a local maximum.
[14] Note that the complexity of a gradient descent iteration is polynomial in d.

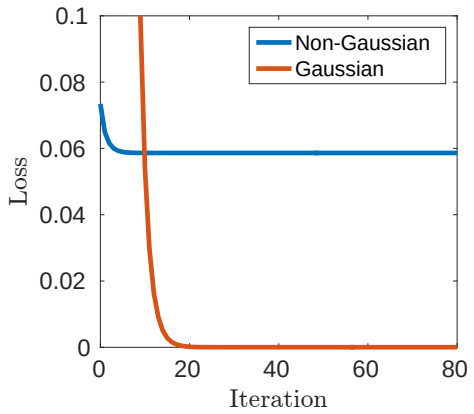

Figure 3. Training loss of Adagrad on the Gaussian and Non-Gaussian datasets. See Section 6 for details.

at a sub-optimal point, whereas the Gaussian case is solved optimally.[15] In the Gaussian case AdaGrad converged to $\boldsymbol{w}^*$. Therefore, given the Gaussian dataset we were able to recover the true weight vector $\boldsymbol{w}^*$, whereas given the data constructed via the reduction we were not, even though both datasets were of the same size. We conclude that these empirical findings are in line with our theoretical results.

7. Networks with Overlapping Filters

Thus far we showed that the non-overlapping case becomes tractable under Gaussian inputs. A natural question is then what happens when overlaps are allowed (namely, the stride is smaller than the filter size). Will gradient descent still find a global optimum? Here we show that this is in fact *not* the case, and that with probability greater than $\frac{1}{4}$ gradient descent will get stuck in a sub-optimal region. In Section 7.1 we analyze this setting for a two dimensional example and provide bounds on the level of suboptimality. In Section 7.2 we report on an empirical study of optimization for networks with overlapping filters. Our results suggest that by restarting gradient descent a constant number of times, it will converge to the global minimum with high probability. Complete proofs of the results are provided in the supplementary material.

7.1. Suboptimality of Gradient Descent for $\mathbb{R}^2$

We consider an instance where there are $k = d - 1$ neurons and matrices $W, W^* \in \mathbb{R}^{k\times d}$ correspond to an overlapping filter of size 2 with stride 1, i.e., for all $1 \le i \le k$ $\boldsymbol{w}_i = (\mathbf{0}_{i-1}, \boldsymbol{w}, \mathbf{0}_{d-i-1})$, $\boldsymbol{w}_i^* = (\mathbf{0}_{i-1}, \boldsymbol{w}^*, \mathbf{0}_{d-i-1})$ where $\mathbf{0}_l = (0, 0, ..., 0) \in \mathbb{R}^l$, $\boldsymbol{w} = (w_1, w_2)$ is a vector of 2 parameters and $\boldsymbol{w}^* = (-w^*, w^*) \in \mathbb{R}^2$, $w^* > 0$. Define the following vectors $\boldsymbol{w}_r = (w_1, w_2, 0)$, $\boldsymbol{w}_l = (0, w_1, w_2)$, $\boldsymbol{w}_r^* = (-w^*, w^*, 0)$, $\boldsymbol{w}_l^* = (0, -w^*, w^*)$ and denote by $\theta_{\boldsymbol{w},\boldsymbol{v}}$ the angle between two vectors $\boldsymbol{w}$ and $\boldsymbol{v}$.

One might wonder why the analysis of the overlapping case should be any different than the non-overlapping case. However, even for a filter of size two, as above, the loss function and consequently the gradient, are more complex in the overlapping case. Indeed, the loss function in this case is given by:

$$\begin{aligned}\ell(\boldsymbol{w}) = \alpha(\|\boldsymbol{w}\|^2 + \|\boldsymbol{w}^*\|^2) - \beta g(\boldsymbol{w}, \boldsymbol{w}^*) \\ + (\beta - 2)(g(\boldsymbol{w}_r, \boldsymbol{w}_l) - g(\boldsymbol{w}_l, \boldsymbol{w}_r^*) \\ - g(\boldsymbol{w}_r, \boldsymbol{w}_l^*) + g(\boldsymbol{w}_r^*, \boldsymbol{w}_l^*)) - \gamma \|\boldsymbol{w}\| \|\boldsymbol{w}^*\|\end{aligned} \tag{16}$$

where $\alpha = \frac{1}{k^2}\left(\frac{k}{2} + \frac{k^2-3k+2}{2\pi}\right)$, $\beta = 2k$ and $\gamma = \frac{k^2-3k+2}{\pi}$.

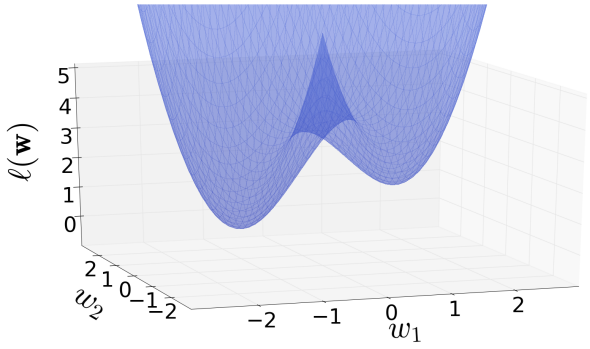

Figure 4. The population risk for a network with overlapping filters, with a two dimensional filter $\boldsymbol{w}^* = [-1, 1]$, $k = 4$, $d = 5$, and Gaussian inputs.

Compared to the objective in Eq. 8 which depends only on $\|\boldsymbol{w}\|$, $\|\boldsymbol{w}\|$ and $\theta_{\boldsymbol{w},\boldsymbol{w}^*}$, we see that the objective in Eq. 16 has new terms such as $g(\boldsymbol{w}_r, \boldsymbol{w}_l^*)$ which has a more complicated dependence on the weight vectors $\boldsymbol{w}^*$ and $\boldsymbol{w}$. This does not only have implications on the analysis, but also on the geometric properties of the loss function and the dynamics of gradient descent. In particular, in Figure 4 we see that the objective has a large sub-optimal region which is not the case when the filters are non-overlapping.

As in the previous section we consider gradient descent updates as in Eq. 13. The following Proposition shows that if $\boldsymbol{w}$ is initialized in the interior of the fourth quadrant of $\mathbb{R}^2$, then it will stay there for all remaining iterations. The proof is a straightforward inspection of the components of the gradient, and is provided in the supplementary.

[15] We note that the value of 0.06 attained by the non-Gaussian case is quite high, since the zero weight vector in this case has loss of order 0.1.

Proposition 7.1. *For any $\lambda \in (0, \frac{1}{3})$, if $\boldsymbol{w}_t$ is in the interior of the fourth quadrant of $\mathbb{R}^2$ then so is $\boldsymbol{w}_{t+1}$.*

Note that in our example the global optimum $\boldsymbol{w}^*$ is in the second quadrant (it's easy to show that it is also unique). Hence, if initialized at the fourth quadrant, gradient descent will remain in a sub-optimal region. The sub-optimality can be clearly seen in Figure 4. In the proposition below we formalize this observation by giving a tight lower bound on the values of $\ell(\boldsymbol{w})$ for $\boldsymbol{w}$ in the fourth quadrant. Specifically, we show that the sub-optimality scales with $O(\frac{1}{k^2})$. The proof idea is to express all angles between all the vectors that appear in Eq. 16 via a single angle parameter θ between $\boldsymbol{w}$ in the fourth quadrant and the positive x-axis. Then it is possible to prove the relatively simpler one dimensional inequality that depends on θ.

Proposition 7.2. *Let $h(k) = \frac{k^2-3k+2}{\pi} + \frac{\sqrt{3}(k-1)}{\pi} + \frac{2(k-1)}{3}$, then for all $\boldsymbol{w}$ in the fourth quadrant $l(\boldsymbol{w}) \geq \frac{2h(k)+1}{k^2(2h(k)+2)}\|\boldsymbol{w}^*\|^2$ and this lower bound is attained by $\tilde{\boldsymbol{w}} = -\frac{h(k)}{h(k)+1}\boldsymbol{w}^*$.*

The above two propositions result in the following characterization of the sub-optimality of gradient descent for $\boldsymbol{w} \in \mathbb{R}^2$ and overlapping filters.

Theorem 7.3. *Define $h(k)$ as in Proposition 7.2. Then with probability $\geq \frac{1}{4}$, a randomly initialized gradient descent with learning rate $\lambda \in (0, \frac{1}{3})$ will get stuck in a sub-optimal region, where each point in this region has loss at least $\frac{2h(k)+1}{k^2(2h(k)+2)}\|\boldsymbol{w}^*\|^2$ and this bound is tight.*

7.2. Empirical study of Gradient Descent for $m > 2$

In Section 7.1 we showed that already for $m = 2$, networks with $\boldsymbol{w} \in \mathbb{R}^m$ and filter overlaps exhibit more complex behavior than those without overlap. This leaves open the question of what happens in the general case under the Gaussian assumption, for various values of d, m and overlaps. We leave the theoretical analysis of this question to future work, but here report on empirical findings that hint at what the solution should look like.

We experimented with a range of d, m and overlap values (see supplementary material for details of the experimental setup). For each value of d, m and overlap we sampled 90 values of $\boldsymbol{w}^*$ from various uniform input distributions with different supports and several pre-defined deterministic values. This resulted in more than 1200 different sampled $\boldsymbol{w}^*$. For each such $\boldsymbol{w}^*$ we ran gradient descent multiple times, each initialized randomly from a different $\boldsymbol{w}_0$. Using the results from these runs, we could estimate the probability of sampling a $\boldsymbol{w}_0$ that would converge to the *unique* global minimum. Viewed differently, this is the probability mass of the basin of attraction of the global optimum. We note that the uniqueness of the global minimum follows easily from equating the population risk (Eq. 3) to 0 and the full proof is deferred to the supplementary material.

Our results are that across all values of d, m, overlap and $\boldsymbol{w}^*$, the probability mass of the basin of attraction is at least $\frac{1}{17}$. The practical implication is that multiple restarts of gradient descent (in this case a few dozen) will find the global optimum with high probability. We leave formal analysis of this intriguing fact for future work.

8. Discussion

The key theoretical question in deep learning is why it succeeds in finding good models despite the non-convexity of the training loss. It is clear that an answer must characterize specific settings where deep learning provably works. Despite considerable recent effort, such a case has not been shown. Here we provide the first analysis of a non-linear architecture where gradient descent is globally optimal, for a certain input distribution, namely Gaussian. Thus our specific characterization is both in terms of architecture (no-overlap networks, single hidden layer, and average pooling) and input distribution. We show that learning in no-overlap architectures is hard, so that some input distribution restriction is necessary for tractability. Note however, that it is certainly possible that other, non-Gaussian, distributions also result in tractability. Some candidates would be sub-Gaussian and log-concave distributions.

Our derivation addressed the population risk, which for the Gaussian case can be calculated in closed form. In practice, one minimizes an empirical risk. Our experiments in Section 6 suggest that optimizing the empirical risk in the Gaussian case is tractable. It would be interesting to prove this formally. It is likely that measure concentration results can be used to get similar results to those we had for the population risk (e.g., see Mei et al., 2016; Xu et al., 2016, for use of such tools).

Convolution layers are among the basic building block of neural networks. Our work is among the first to analyze optimization for these. The architecture we study is similar in structure to convolutional networks, in the sense of using parameter tying and pooling. However, most standard convolutional layers have overlap and use max pooling. In Section 7 we provide initial results for the case of overlap, showing there is hope for proving optimality for gradient descent with random restarts. Analyzing max pooling would be very interesting and is left for future work.

Finally, we note that distribution dependent tractability has been shown for intersection of halfspaces (Klivans et al., 2009), which is a non-convolutional architecture. However, these results do not use gradient descent. It would be very interesting to use our techniques to try and understand gradient descent for the population risk in these settings.

Acknowledgements

This work was supported by the Blavatnik Computer Science Research Fund, the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI), and an ISF Centers of Excellence grant.

References

Allen-Zhu, Zeyuan and Hazan, Elad. Variance reduction for faster non-convex optimization. *arXiv preprint arXiv:1603.05643*, 2016.

Andoni, Alexandr, Panigrahy, Rina, Valiant, Gregory, and Zhang, Li. Learning polynomials with neural networks. In *Proceedings of the 31th International Conference on Machine Learning*, pp. 1908–1916, 2014.

Auer, Peter, Herbster, Mark, Warmuth, Manfred K, et al. Exponentially many local minima for single neurons. *Advances in neural information processing systems*, pp. 316–322, 1996.

Baum, Eric B. A polynomial time algorithm that learns two hidden unit nets. *Neural Computation*, 2(4):510–522, 1990.

Blum, Avrim L and Rivest, Ronald L. Training a 3-node neural network is np-complete. In *Machine learning: From theory to applications*, pp. 9–28. Springer, 1993.

Cho, Youngmin and Saul, Lawrence K. Kernel methods for deep learning. In *Advances in neural information processing systems*, pp. 342–350, 2009.

Choromanska, Anna, Henaff, Mikael, Mathieu, Michael, Arous, Gérard Ben, and LeCun, Yann. The loss surfaces of multilayer networks. In *AISTATS*, 2015.

Daniely, Amit, Linial, Nati, and Shalev-Shwartz, Shai. From average case complexity to improper learning complexity. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, pp. 441–448. ACM, 2014.

Duchi, John, Hazan, Elad, and Singer, Yoram. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

Garey, Michael R. and Johnson, David S. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990. ISBN 0716710455.

Ge, Rong, Huang, Furong, Jin, Chi, and Yuan, Yang. Escaping from saddle points-online stochastic gradient for tensor decomposition. In *COLT*, pp. 797–842, 2015.

Ge, Rong, Lee, Jason D, and Ma, Tengyu. Matrix completion has no spurious local minimum. In *Advances in Neural Information Processing Systems*, pp. 2973–2981, 2016.

Goel, Surbhi, Kanade, Varun, Klivans, Adam, and Thaler, Justin. Reliably learning the ReLU in polynomial time. *arXiv preprint arXiv:1611.10258*, 2016.

Haeffele, Benjamin D and Vidal, René. Global optimality in tensor factorization, deep learning, and beyond. *arXiv preprint arXiv:1506.07540*, 2015.

Hardt, Moritz and Ma, Tengyu. Identity matters in deep learning. *arXiv preprint arXiv:1611.04231*, 2016.

Hardt, Moritz, Ma, Tengyu, and Recht, Benjamin. Gradient descent learns linear dynamical systems. *arXiv preprint arXiv:1609.05191*, 2016.

Hinton, Geoffrey, Deng, Li, Yu, Dong, Dahl, George E, Mohamed, Abdel-rahman, Jaitly, Navdeep, Senior, Andrew, Vanhoucke, Vincent, Nguyen, Patrick, Sainath, Tara N, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29 (6):82–97, 2012.

Janzamin, Majid, Sedghi, Hanie, and Anandkumar, Anima. Beating the perils of non-convexity: Guaranteed training of neural networks using tensor methods. *arXiv preprint arXiv:1506.08473*, 2015.

Kakade, Sham M, Kanade, Varun, Shamir, Ohad, and Kalai, Adam. Efficient learning of generalized linear and single index models with isotonic regression. In *Advances in Neural Information Processing Systems 24*, pp. 927–935. 2011.

Kawaguchi, Kenji. Deep learning without poor local minima. In *Advances In Neural Information Processing Systems*, pp. 586–594, 2016.

Klivans, Adam. Cryptographic hardness of learning. In *Encyclopedia of Algorithms*, pp. 210–212. Springer, 2008.

Klivans, Adam R, Long, Philip M, and Tang, Alex K. Baums algorithm learns intersections of halfspaces with respect to log-concave distributions. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pp. 588–600. Springer, 2009.

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

Lee, Jason D., Simchowitz, Max, Jordan, Michael I., and Recht, Benjamin. Gradient descent only converges to minimizers. In *Proceedings of the 29th Conference on Learning Theory*, pp. 1246–1257, 2016.

Lin, Min, Chen, Qiang, and Yan, Shuicheng. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.

Livni, Roi, Shalev-Shwartz, Shai, and Shamir, Ohad. On the computational efficiency of training neural networks. In *Advances in Neural Information Processing Systems*, pp. 855–863, 2014.

Mei, Song, Bai, Yu, and Montanari, Andrea. The landscape of empirical risk for non-convex losses. *arXiv preprint arXiv:1607.06534*, 2016.

Milletari, Fausto, Navab, Nassir, and Ahmadi, Seyed-Ahmad. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3D Vision (3DV), 2016 Fourth International Conference on*, pp. 565–571. IEEE, 2016.

Nesterov, Yurii. Introductory lectures on convex optimization. pp. 22–29, 2004.

Safran, Itay and Shamir, Ohad. On the quality of the initial basin in overspecified neural networks. In *Proceedings of the 33nd International Conference on Machine Learning*, pp. 774–782, 2016.

Shamir, Ohad. Distribution-specific hardness of learning neural networks. *arXiv preprint arXiv:1609.01037*, 2016.

Soudry, Daniel and Carmon, Yair. No bad local minima: Data independent training error guarantees for multilayer neural networks. *arXiv preprint arXiv:1605.08361*, 2016.

Wu, Yonghui, Schuster, Mike, Chen, Zhifeng, Le, Quoc V., Norouzi, Mohammad, Macherey, Wolfgang, Krikun, Maxim, Cao, Yuan, Gao, Qin, Macherey, Klaus, Klingner, Jeff, Shah, Apurva, Johnson, Melvin, Liu, Xiaobing, Kaiser, Lukasz, Gouws, Stephan, Kato, Yoshikiyo, Kudo, Taku, Kazawa, Hideto, Stevens, Keith, Kurian, George, Patil, Nishant, Wang, Wei, Young, Cliff, Smith, Jason, Riesa, Jason, Rudnick, Alex, Vinyals, Oriol, Corrado, Greg, Hughes, Macduff, and Dean, Jeffrey. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.

Xu, Ji, Hsu, Daniel J, and Maleki, Arian. Global analysis of expectation maximization for mixtures of two gaussians. In *Advances in Neural Information Processing Systems*, pp. 2676–2684, 2016.

Zhang, Chiyuan, Bengio, Samy, Hardt, Moritz, Recht, Benjamin, and Vinyals, Oriol. Understanding deep learning requires rethinking generalization. *CoRR*, abs/1611.03530, 2016. URL http://arxiv.org/abs/1611.03530.

Zhang, Qiuyi, Panigrahy, Rina, Sachdeva, Sushant, and Rahimi, Ali. Electron-proton dynamics in deep learning. *arXiv preprint arXiv:1702.00458*, 2017.