Why Lottery Ticket Wins? A Theoretical Perspective of Sample Complexity on Pruned Neural Networks

Shuai Zhang
Rensselaer Polytechnic Institute
Troy, NY, USA 12180
zhangs21@rpi.edu

Meng Wang
Rensselaer Polytechnic Institute
Troy, NY, USA 12180
wangm7@rpi.edu

Sijia Liu
Michigan State University
East Lansing, MI, USA 48824
MIT-IBM Watson AI Lab, IBM Research
liusiji5@msu.edu

Pin-Yu Chen
IBM Research
Yorktown Heights, NY, USA 10562
Pin-Yu.Chen@ibm.com

Jinjun Xiong
University at Buffalo
Buffalo NY, USA 14260
jinjun@buffalo.edu

Abstract

The *lottery ticket hypothesis* (LTH) [20] states that learning on a properly pruned network (the *winning ticket*) improves test accuracy over the original unpruned network. Although LTH has been justified empirically in a broad range of deep neural network (DNN) involved applications like computer vision and natural language processing, the theoretical validation of the improved generalization of a winning ticket remains elusive. To the best of our knowledge, our work, for the first time, characterizes the performance of training a pruned neural network by analyzing the geometric structure of the objective function and the sample complexity to achieve zero generalization error. We show that the convex region near a desirable model with guaranteed generalization enlarges as the neural network model is pruned, indicating the structural importance of a winning ticket. Moreover, when the algorithm for training a pruned neural network is specified as an (accelerated) stochastic gradient descent algorithm, we theoretically show that the number of samples required for achieving zero generalization error is proportional to the number of the non-pruned weights in the hidden layer. With a fixed number of samples, training a pruned neural network enjoys a faster convergence rate to the desired model than training the original unpruned one, providing a formal justification of the improved generalization of the winning ticket. Our theoretical results are acquired from learning a pruned neural network of one hidden layer, while experimental results are further provided to justify the implications in pruning multi-layer neural networks.

1 Introduction

Neural network pruning can reduce the computational cost of model training and inference significantly and potentially lessen the chance of overfitting [33, 26, 15, 25, 28, 51, 58, 41]. The recent *Lottery Ticket Hypothesis* (LTH) [20] claims that a randomly initialized dense neural network always contains a so-called "winning ticket," which is a sub-network bundled with the corresponding initialization, such that when trained in isolation, this winning ticket can achieve at least the same testing accuracy as that of the original network by running at most the same amount of training time. This so-called "improved generalization of winning tickets" is verified empirically in [20]. LTH has attracted a significant amount of recent research interests [45, 70, 39]. Despite the empirical success [19, 63, 55, 11], the theoretical justification of winning tickets remains elusive except for a few recent works. [39] provides the first theoretical evidence that within a randomly initialized neural network, there exists a good sub-network that can achieve the same test performance as the original network. Meanwhile, recent work [42] trains neural network by adding the ℓ_1 regularization term to obtain a relatively sparse neural network, which has a better performance numerically.

wrong claim

However, the theoretical foundation of network pruning is limited. The existing theoretical works usually focus on finding a sub-network that achieves a tolerable loss in either expressive power or training accuracy, compared with the original dense network [2, 71, 61, 43, 4, 3, 35, 5, 59]. To the best of our knowledge, there exists no theoretical support for the *improved* generalization achieved by winning tickets, i.e., pruned networks with faster convergence and better test accuracy.

Contributions: This paper provides the *first* systematic analysis of learning pruned neural networks with a finite number of training samples in the oracle-learner setup, where the training data are generated by a unknown neural network, the *oracle*, and another network, the *learner*, is trained on the dataset. Our analytical results also provide a justification of the LTH from the perspective of the sample complexity. In particular, we provide the *first* theoretical justification of the improved generalization of winning tickets. Specific contributions include:

1. **Pruned neural network learning via accelerated gradient descent (AGD)**: We propose an AGD algorithm with tensor initialization to learn the pruned model from training samples. Our algorithm converges to the oracle model linearly, which has guaranteed generalization.

2. **First sample complexity analysis for pruned networks**: We characterize the required number of samples for successful convergence, termed as the *sample complexity*. Our sample complexity bound depends linearly on the number of the non-pruned weights and is a significant reduction from directly applying conventional complexity bounds in [69, 66, 67].

3. **Characterization of the benign optimization landscape of pruned networks**: We show analytically that the empirical risk function has an *enlarged* convex region for a pruned network, justifying the importance of a good sub-network (i.e., the winning ticket).

4. **Characterization of the improved generalization of winning tickets**: We show that gradient-descent methods converge faster to the oracle model when the neural network is properly pruned, or equivalently, learning on a pruned network returns a model closer to the oracle model with the same number of iterations, indicating the improved generalization of winning tickets.

Notations. Vectors are bold lowercase, matrices and tensors are bold uppercase. Scalars are in normal font, and sets are in calligraphy and blackboard bold font. $\boldsymbol{I}$ denote the identity matrix. $\mathbb{N}$ and $\mathbb{R}$ denote the sets of nature number and real number, respectively. $\|\boldsymbol{z}\|$ denotes the ℓ_2-norm of a vector $\boldsymbol{z}$, and $\|\boldsymbol{Z}\|_2$, $\|\boldsymbol{Z}\|_F$ and $\|\boldsymbol{Z}\|_\infty$ denote the spectral norm, Frobenius norm and the maximum value of matrix $\boldsymbol{Z}$, respectively. $[Z]$ stands for the set of $\{1, 2, \cdots, Z\}$ for any number $Z \in \mathbb{N}$. In addition, $f(r) = \mathcal{O}(g(r))$ (or $f(r) = \Omega(g(r))$) if $f \leq C \cdot g$ (or $f \geq C \cdot g$) for some constant $C > 0$ when r is large enough. $f(r) = \Theta(g(r))$ if both $f(r) = \mathcal{O}(g(r))$ and $f(r) = \Omega(g(r))$ holds, where $c \cdot g \leq f \leq C \cdot g$ for some constant $0 \leq c \leq C$ when r is large enough.

1.1 Related Work

Network pruning. Network pruning methods seek a compressed model while maintaining the expressive power. Numerical experiments have shown that over 90% of the parameters can be pruned without a significant performance loss [10]. Examples of pruning methods include irregular weight pruning [25], structured weight pruning [57], neuron-based pruning [28], and projecting the weights to a low-rank subspace [13].

Winning tickets. [20] employs an *Iterative Magnitude Pruning* (IMP) algorithm to obtain the proper sub-network and initialization. IMP and its variations [22, 46] succeed in deeper networks like Residual Networks (Resnet)-50 and Bidirectional Encoder Representations from Transformers (BERT) network [11]. [21] shows that IMP succeeds in finding the "winning ticket" if the ticket is stable to stochastic gradient descent noise. In parallel, [36] shows numerically that the "winning ticket" initialization does not improve over a random initialization once the correct sub-networks are found, suggesting that the benefit of "winning ticket" mainly comes from the sub-network structures. [18] analyzes the sample complexity of IMP from the perspective of recovering a sparse vector in a linear model rather than learning neural networks.

Feature sparsity. High-dimensional data often contains redundant features, and only a subset of the features is used in training [6, 14, 27, 60, 68]. Conventional approaches like wrapper and filter methods score the importance of each feature in a certain way and select the ones with highest scores [24]. Optimization-based methods add variants of the ℓ_0 norm as a regularization to promote feature sparsity [68]. Different from network pruning where the feature dimension still remains high during training, the feature dimension is significantly reduced in training when promoting feature sparsity.

Over-parameterized model. When the number of weights in a neural network is much larger than the number of training samples, the landscape of the objective function of the learning problem has no spurious local minima, and first-order algorithms converge to one of the global optima [37, 44, 64, 50, 9, 49, 38]. However, the global optima is not guaranteed to generalize well on testing data [62, 64].

Generalization analyses. The existing generalization analyses mostly fall within three categories. One line of research employs the Mean Field approach to model the training process by a differential equation assuming infinite network width and infinitesimal training step size [12, 40, 56]. Another approach is the neural tangent kernel (NTK) [30], which requires strong and probably unpractical over-parameterization such that the nonlinear neural network model behaves as its linearization around the initialization [1, 17, 72, 73]. The third line of works follow the oracle-learner setup, where the data are generated by an unknown oracle model, and the learning objective is to estimate the oracle model, which has a generalization guarantee on testing data. However, the objective function has intractably many spurious local minima even for one-hidden-layer neural networks [48, 47, 64]. Assuming an infinite number of training samples, [8, 16, 52] develop learning methods to estimate the oracle model. [23, 69, 66, 67] extend to the practical case of a finite number of samples and characterize the sample complexity for recovering the oracle model. Because the analysis complexity explodes when the number of hidden layers increases, all the analytical results about estimating the oracle model are limited to one-hidden-layer neural networks, and the input distribution is often assumed to be the standard Gaussian distribution.

2 Problem Formulation

In an oracle-learner model, given any input $\boldsymbol{x} \in \mathbb{R}^d$, the corresponding output y is generated by a pruned one-hidden-layer neural network, called ***oracle***, as shown in Figure 1. The oracle network is equipped with K neurons where the j-th neuron is connected to any arbitrary r_j^* ($r_j^* \le d$) input features. Let $\boldsymbol{W}^* = [\boldsymbol{w}_1^*, \cdots, \boldsymbol{w}_K^*] \in \mathbb{R}^{d\times K}$ denotes all the weights (pruned ones are represented by zero). The number of non-zero entries in $\boldsymbol{w}_j^*$ is at most r_j^*. The oracle network is not unique because permuting neurons together with the corresponding weights does not change the output. Therefore, the output label y obtained by the oracle network satisfies [1]

$$y = \frac{1}{K}\sum_{j=1}^{K}\phi(\boldsymbol{w}_j^{*T}\boldsymbol{x}) + \xi := g(\boldsymbol{x}; \boldsymbol{W}^*) + \xi = g(\boldsymbol{x}; \boldsymbol{W}^*\boldsymbol{P}) + \xi, \tag{1}$$

where ξ is arbitrary unknown additive noise bounded by some constant $|\xi|$, ϕ is the rectified linear unit (ReLU) activation function with $\phi(z) = \max\{z, 0\}$, and $\boldsymbol{P} \in \{0,1\}^{K\times K}$ is any permutation matrix. $\boldsymbol{M}^*$ is a ***mask matrix*** for the oracle network, such that $M_{j,i}^*$ equals to 1 if the weight $\boldsymbol{w}_{j,i}^*$ is not pruned, and 0 otherwise. Then, $\boldsymbol{M}^*$ is an indicator matrix for the non-zero entries of $\boldsymbol{W}^*$ with $\boldsymbol{M}^* \odot \boldsymbol{W}^* = \boldsymbol{W}^*$, where $\odot$ is entry-wise multiplication.

Based on N pairs of training samples $\mathcal{D} = \{\boldsymbol{x}_n, y_n\}_{n=1}^N$ generated by the oracle, we train on a ***learner*** network equipped with the same number of neurons in the oracle network. However, the j-th neuron in the learner network is connected to r_j input features rather than r_j^*. Let $r_{\min}$, $r_{\max}$, and r_{ave} denote the minimum, maximum, and average value of $\{r_j\}_{j=1}^K$, respectively. Let $\boldsymbol{M}$ denote the

[1]It is extendable to binary classification, and the output is generated by $\text{Prob}\big(y_n = 1|\boldsymbol{x}_n\big) = g(\boldsymbol{x}_n; \boldsymbol{W}^*)$.

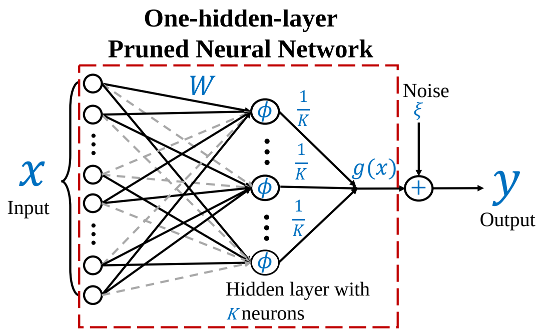

Figure 1: Illustration of the model

mask matrix with respect to the learner network, and $\boldsymbol{w}_j$ is the j-th column of $\boldsymbol{W}$. The empirical risk function is defined as

$$\hat{f}_{\mathcal{D}}(\boldsymbol{W}) = \frac{1}{2N}\sum_{n=1}^{N}\Big(\frac{1}{K}\sum_{j=1}^{K}\phi(\boldsymbol{w}_j^T\boldsymbol{x}_n) - y_n\Big)^2. \tag{2}$$

When the mask $\boldsymbol{M}$ is given, the learning objective is to estimate a proper weight matrix $\boldsymbol{W}$ for the *learner* network from the training samples $\mathcal{D}$ via solving

$$\min_{\boldsymbol{W}\in\mathbb{R}^{d\times K}} \quad \hat{f}_{\mathcal{D}}(\boldsymbol{W}) \qquad \text{s.t.} \quad \boldsymbol{M}\odot\boldsymbol{W} = \boldsymbol{W}. \tag{3}$$

$\boldsymbol{M}$ is called an ***accurate mask*** if the support of $\boldsymbol{M}$ covers the support of a permutation of $\boldsymbol{M}^*$, i.e., there exists a permutation matrix $\boldsymbol{P}$ such that $(\boldsymbol{M}^*\boldsymbol{P})\odot\boldsymbol{M} = \boldsymbol{M}^*$. When $\boldsymbol{M}$ is accurate, and $\xi = 0$, there exists a permutation matrix $\boldsymbol{P}$ such that $\boldsymbol{W}^*\boldsymbol{P}$ is a global optimizer to (3). Hence, if $\boldsymbol{W}^*\boldsymbol{P}$ can be estimated by solving (3), one can learn the oracle network accurately, which has guaranteed generalization performance on the testing data.

We assume $\boldsymbol{x}_n$ is independent and identically distributed from the standard Gaussian distribution $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{d\times d})$. The Gaussian assumption is motivated by the data whitening [34] and batch normalization techniques [29] that are commonly used in practice to improve learning performance. Moreover, training one-hidden-layer neural network with multiple neurons has intractable many fake minima [47] without any input distribution assumption. In addition, the theoretical results in Section 3 assume an accurate mask, and inaccurate mask is evaluated empirically in Section 4.

The questions that this paper addresses include: 1. **what algorithm** to solve (3)? 2. what is the **sample complexity** for the accurate estimate of the weights in the oracle network? 3. what is the **impact of the network pruning** on the difficulty of the learning problem and the performance of the learned model?

3 Algorithm and Theoretical Results

Section 3.1 studies the geometric structure of (3), and the main results are in Section 3.2. Section 3.3 briefly introduces the proof sketch and technical novelty, and the limitations are in Section 3.4.

3.1 Local Geometric Structure

Theorem 1 characterizes the local convexity of $\hat{f}_{\mathcal{D}}$ in (3). It has two important implications.

1. **Strictly locally convex near ground truth**: $\hat{f}_{\mathcal{D}}$ is strictly convex near $\boldsymbol{W}^*\boldsymbol{P}$ for some permutation matrix $\boldsymbol{P}$, and the radius of the convex ball is negatively correlated with $\sqrt{\widetilde{r}}$, where $\widetilde{r}$ is in the order of r_{ave}. Thus, the convex ball enlarges as any r_j decreases.

2. **Importance of the winning ticket architecture**: Compared with training on the dense network directly, training on a properly pruned sub-network has a larger local convex region near $\boldsymbol{W}^*\boldsymbol{P}$, which may lead to easier estimation of $\boldsymbol{W}^*\boldsymbol{P}$. To some extent, this result can be viewed as a theoretical validation of the importance of the winning architecture (a good sub-network) in [20]. Formally, we have

Theorem 1 (Local Convexity)**.** *Assume the mask $\boldsymbol{M}$ of the learner network is accurate. Suppose constants ε_0, $\varepsilon_1 \in (0,1)$ and the number of samples satisfies*

$$N = \Omega\big(\varepsilon_1^{-2} K^4 \widetilde{r} \log q\big), \tag{4}$$

for some large constant $q > 0$, where

$$\widetilde{r} = \frac{1}{8K^4}\Big(\textstyle\sum_{k=1}^{K}\sum_{j=1}^{K}(1+\delta_{j,k})(r_j + r_k)^{\frac{1}{2}}\Big)^2, \tag{5}$$

$\delta_{j,k}$ is 1 if the indices of non-pruned weights in the j-th and k-th neurons overlap and 0 otherwise. Then, there exists a permutation matrix $\boldsymbol{P}$ such that for any $\boldsymbol{W}$ that satisfies

$$\|\boldsymbol{W} - \boldsymbol{W}^*\boldsymbol{P}\|_F = \mathcal{O}\big(\tfrac{\varepsilon_0}{K^2}\big), \quad \textit{and } \boldsymbol{M}\odot\boldsymbol{W} = \boldsymbol{W}, \tag{6}$$

its Hessian of $\hat{f}_{\mathcal{D}}$, with probability at least $1 - K\cdot q^{-r_{\min}}$, is bounded as:

$$\Theta\Big(\frac{1-\varepsilon_0-\varepsilon_1}{K^2}\Big)\boldsymbol{I} \preceq \nabla^2 \hat{f}_{\mathcal{D}}(\boldsymbol{W}) \preceq \Theta\Big(\frac{1}{K}\Big)\boldsymbol{I}. \tag{7}$$

Remark 1.1 (Parameter $\widetilde{r}$): Clearly $\widetilde{r}$ is a monotonically increasing function of any r_j from (5). Moreover, one can check that $\frac{1}{8}r_{\text{ave}} \le \widetilde{r} \le r_{\text{ave}}$. Hence, $\widetilde{r}$ is in the order of r_{ave}.

Remark 1.2 (Local landscape): Theorem 1 shows that with enough samples as shown in (4), in a local region of $\boldsymbol{W}^*\boldsymbol{P}$ as shown in (6), all the eigenvalues of the Hessian matrix of the empirical risk function are lower and upper bounded by two positive constants. This property is useful in designing efficient algorithms to recover $\boldsymbol{W}^*\boldsymbol{P}$, as shown in Section 3.2.

Remark 1.3 (Size of the convex region): When the number of samples N is fixed and r changes, ε_1 can be $\Theta(\sqrt{\widetilde{r}/N})$ while (4) is still met. ε_0 in (7) can be arbitrarily close to but smaller than $1-\varepsilon_1$ so that the Hessian matrix is still positive definite. Then from (6), the radius of the convex ball is $\Theta(1) - \Theta(\sqrt{\widetilde{r}/N})$, indicating an enlarged region when $\widetilde{r}$ decreases. The enlarged convex region serves as an important component in proving the faster convergence rate, summarized in Theorem 2. Besides this, as Figure 1 shown in [20], the authors claim that the learning is stable if the linear interpolation of the learned models with SGD noises still remain similar in performance, which is summarized as the concept "linearly connected region." Intuitively, we conjecture that the winning ticket shows a better performance in the stability analysis because it has a larger convex region. In the other words, a larger convex region indicates that the learning is more likely to be stable in the linearly connected region.

3.2 Convergence Analysis with Accelerated Gradient Descent

We propose to solve the non-convex problem (3) via the accelerated gradient descent (AGD) algorithm, summarized in Algorithm 1. Compared with the vanilla gradient descent (GD) algorithm, AGD has an additional momentum term, denoted by $\beta(\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t-1)})$, in each iteration. AGD enjoys a faster convergence rate than vanilla GD in solving optimization problems, including learning neural networks [65]. Vanilla GD can be viewed as a special case of AGD by letting $\beta = 0$. The initial point $\boldsymbol{W}^{(0)}$ can be obtained through a tensor method, and the details are provided in Appendix B.

Algorithm 1 Accelerated Gradient Descent (AGD) Algorithm

```
1: Input: training data $\mathcal{D} = \{(\boldsymbol{x}_n, y_n)\}_{n=1}^N$, gradient step size $\eta$, momentum parameter $\beta$, and an
   initialization $\boldsymbol{W}^{(0)}$ by the tensor initialization method;
2: Partition $\mathcal{D}$ into $T = \log(1/\varepsilon)$ disjoint subsets, denoted as $\{\mathcal{D}_t\}_{t=1}^T$;
3: for $t = 1, 2, \cdots, T$ do
4:     $\boldsymbol{W}^{(t+1)} = \boldsymbol{W}^{(t)} - \eta\cdot\boldsymbol{M}\odot\nabla\hat{f}_{\mathcal{D}_t}(\boldsymbol{W}^{(t)}) + \beta(\boldsymbol{W}^{(t)} - \boldsymbol{W}^{(t-1)})$
5: end for
6: Return: $\boldsymbol{W}^{(T)}$
```

The theoretical analyses of our algorithm are summarized in Theorem 2 (convergence) and Lemma 1 (Initialization). The significance of these results can be interpreted from the following aspects.

1. **Linear convergence to the oracle model**: Theorem 2 implies that if initialized in the local convex region, the iterates generated by AGD converge linearly to $\boldsymbol{W}^*\boldsymbol{P}$ for some $\boldsymbol{P}$ when noiseless. When there is noise, they converge to a point $\boldsymbol{W}^{(T)}$. The distance between $\boldsymbol{W}^{(T)}$ and $\boldsymbol{W}^*\boldsymbol{P}$ is proportional to the noise level and scales in terms of $\mathcal{O}(\sqrt{\widetilde{r}/N})$. Moreover, when N is fixed, the convergence rate of AGD is $\Theta(\sqrt{\widetilde{r}/K})$. Recall that Algorithm 1 reduces to the vanilla GD by setting $\beta = 0$. The rate for the vanilla GD algorithm here is $\Theta(\sqrt{\widetilde{r}}/K)$ by setting $\beta = 0$ by Theorem 2, indicating a slower convergence than AGD. Lemma 1 shows the tensor initialization method indeed returns an initial point in the convex region.

2. **Sample complexity for accurate estimation**: We show that the required number of samples for successful estimation of the oracle model is $\Theta\big(\widetilde{r}\log q \log(1/\varepsilon)\big)$ for some large constant q and estimation accuracy ε. Our sample complexity is much less than the conventional bound of $\Theta(d\log q\log(1/\varepsilon))$ for one-hidden-layer networks [69, 66, 67]. This is the first theoretical characterization of learning a pruned network from the perspective of sample complexity.

3. **Improved generalization of winning tickets**: We prove that with a fixed number of training samples, training on a properly pruned sub-network converges faster to $\boldsymbol{W}^*\boldsymbol{P}$ than training on the original dense network. Our theoretical analysis justifies that training on the winning ticket can meet or exceed the same test accuracy within the same number of iterations. To the best of our knowledge, our result here provides the first theoretical justification for this intriguing empirical finding of "improved generalization of winning tickets" by [20].

Theorem 2 (Convergence)**.** *Assume the mask $\boldsymbol{M}$ of the learner network is accurate. Suppose $\boldsymbol{W}^{(0)}$ satisfies* (6) *and the number of samples satisfies*

$$N = \Omega\big(\varepsilon_0^{-2}K^6\widetilde{r}\log q\log(1/\varepsilon)\big) \tag{8}$$

for some $\varepsilon_0 \in (0, 1/2)$. Let $\eta = K/14$ in Algorithm 1. Then, the iterates $\{\boldsymbol{W}^{(t)}\}_{t=1}^{T}$ returned by Algorithm 1 converges linearly to $\boldsymbol{W}^*$ up to the noise level with probability at least $1 - K^2T\cdot q^{-r_{\min}}$

$$\|\boldsymbol{W}^{(t)} - \boldsymbol{W}^*\boldsymbol{P}\|_F \leq \nu(\beta)^t\|\boldsymbol{W}^{(0)} - \boldsymbol{W}^*\boldsymbol{P}\|_F + \mathcal{O}\Big(\sum_j\sqrt{\frac{r_j\log q}{N}}\Big)\cdot|\xi|, \tag{9}$$

$$\textit{and}\quad \|\boldsymbol{W}^{(T)} - \boldsymbol{W}^*\boldsymbol{P}\|_F \leq \varepsilon\|\boldsymbol{W}^*\|_F + \mathcal{O}\Big(\sum_j\sqrt{\frac{r_j\log q}{N}}\Big)\cdot|\xi|, \tag{10}$$

for a fixed permutation matrix $\boldsymbol{P}$, where $\nu(\beta)$ is the rate of convergence that depends on β with $\nu(\beta^*) = 1 - \Theta\big(\frac{1-\varepsilon_0}{\sqrt{K}}\big)$ for some non-zero β^* and $\nu(0) = 1 - \Theta\big(\frac{1-\varepsilon_0}{K}\big)$.

Lemma 1 (Initialization)**.** *Assume the noise $|\xi| \leq \|\boldsymbol{W}^*\|_2$ and the number of samples $N = \Omega\big(\varepsilon_0^{-2}K^5 r_{\max}\log q\big)$ for $\varepsilon_0 > 0$ and large constant q, the tensor initialization method outputs $\boldsymbol{W}^{(0)}$ such that (6) holds, i.e., $\|\boldsymbol{W}^{(0)} - \boldsymbol{W}^*\|_F = \mathcal{O}\big(\frac{\varepsilon_0\sigma_K}{K^2}\big)$ with probability at least $1 - q^{-r_{\max}}$.*

Remark 2.1 (Faster convergence on pruned network): With a fixed number of samples, when $\widetilde{r}$ decreases, ε_0 can increase as $\Theta(\sqrt{\widetilde{r}})$ while (8) is still met. Then $\nu(0) = \Theta(\sqrt{\widetilde{r}}/K)$ and $\nu(\beta^*) = \Theta(\sqrt{\widetilde{r}/K})$. Therefore, when $\widetilde{r}$ decreases, both the stochastic and accelerated gradient descent converge faster. Note that as long as $\boldsymbol{W}^{(0)}$ is initialized in the local convex region, not necessarily by the tensor method, Theorem 2 guarantees the accurate recovery. [66, 67] analyze AGD on convolutional neural networks, while this paper focuses on network pruning.

Remark 2.2 (**Sample complexity of initialization**): From Lemma 1, the required number of samples for a proper initialization is $\Omega\big(\varepsilon_0^{-2}K^5 r_{\max}\log q\big)$. Because $r_{\max} \leq K r_{\text{ave}}$ and $\widetilde{r} = \Omega(r_{\text{ave}})$, this number is no greater than the sample complexity in (8). Thus, provided that (8) is met, Algorithm 1 can estimate the oracle network model accurately.

Remark 2.3 (Inaccurate mask): The above analyses are based on the assumption that the mask of the learner network is accurate. In practice, a mask can be obtained by an iterative pruning method such as [20] or a one-shot pruning method such as [55]. In Appendix E, we prove that the magnitude pruning method can obtain an accurate mask with enough training samples. Moreover, empirical experiments in Section 4.2 and 4.3 suggest that even if the mask is not accurate, the three properties (linear convergence, sample complexity with respect to the network size, and improved generalization of winning tickets) can still hold. Therefore, our theoretical results provide some insight into the empirical success of network pruning.

3.3 The Sketch of Proofs and Technical Novelty

Our proof outline is inspired by [69] on fully connected neural networks, however, major technical changes are made in this paper to generalize the analysis to an arbitrarily pruned network. To characterize the local convex region of $\hat{f}_{\mathcal{D}}$ (Theorem 1), the idea is to bound the Hessian matrix of the population risk function, which is the expectation of the empirical risk function, locally and then characterize the distance between the empirical and population risk functions through the concentration bounds. Then, the convergence of AGD (Theorem 2) is established based on the desired local curvature, which in turn determines the sample complexity. Finally, to initialize in the local convex region (Lemma 1), we construct tensors that contain the weights information and apply a decomposition method to estimate the weights.

Our technical novelties are as follows. First, a direct application of the results in [69] leads to a sample complexity bound that is linear in the feature dimension d. We develop new techniques to tighten the sample complexity bound to be linear in $\tilde{r}$, which can be significantly smaller than d for a sufficiently pruned network. Specifically, we develop new concentration bounds (Lemmas 4 and 5 in Appendix) to bound the distance between the population and empirical risk functions rather than using the bound in [69]. Second, instead of restricting the acitivation to be smooth for convergence analysis, we study the case of ReLU function which is non-smooth. Third, new tensors are constructed for pruned networks (see (21)-(23) in Appendix) in computing the initialization, and our new concentration bounds are employed to reduce the required number of samples for a proper initialization. Last, Algorithm 1 employs AGD and is proved to converge faster than the GD algorithm in [69].

3.4 Limitations

Like most theoretical works based on the oracle-learner setup, limitations of this work include (1) one hidden layer only; and (2) the input follows the Gaussian distribution. Extension to multi-layer might be possible if the following technical challenges are addressed. First, when characterizing the local convex region, one needs to show that the Hessian matrix is positive definite. In multi-layer networks, the Hessian matrix is more complicated to compute. Second, new concentration bounds need to be developed because the input feature distributions to the second and third layers depend on the weights in previous layers. Third, the initialization approach needs to be revised. The team is also investigating the other input distributions such as Gaussian mixture models.

4 Numerical Experiments

The theoretical results are first verified on synthetic data, and we then analyze the pruning performance on both synthetic and real datasets. In Section 4.1, Algorithm 1 is implemented with minor modification, such that, the initial point is randomly selected as $\|\boldsymbol{W}^{(0)} - \boldsymbol{W}^*\|_F / \|\boldsymbol{W}^*\|_F < \lambda$ for some $\lambda > 0$ to reduce the computation. Algorithm 1 terminates when $\|\boldsymbol{W}^{(t+1)} - \boldsymbol{W}^{(t)}\|_F / \|\boldsymbol{W}^{(t)}\|_F$ is smaller than 10^{-8} or reaching 10000 iterations. In Sections 4.2 and 4.3, the Gradient Signal Preservation (GraSP) algorithm [55] and IMP algorithm [10, 20][2] are implemented to prune the neural networks. As many works like [11, 10, 20] have already verified the faster convergence and better generalization accuracy of the winning tickets empirically, we only include the results of some representative experiments, such as training MNIST and CIFAR-10 on Lenet-5 [32] and Resnet-50 [27] networks, to verify our theoretical findings.

The synthetic data are generated using a oracle model in Figure 1. The input $\boldsymbol{x}_n$'s are randomly generated from Gaussian distribution $\mathcal{N}(0, \boldsymbol{I}_{d\times d})$ independently, and indices of non-pruned weights of the j-th neuron are obtained by randomly selecting r_j numbers without replacement from $[d]$. For the convenience of generating specific $\widetilde{r}$, the indices of non-pruned weights are almost overlapped ($\sum_j \sum_k \delta_j \delta_k > 0.95K^2$) except for Figure 5. In Figures 2 and 4, r_j is selected uniformly from $[0.9\widetilde{r}, 1.1\widetilde{r}]$ for a given $\widetilde{r}$, and r_j are the same in value for all j in other figures. Each non-zero entry of $\boldsymbol{W}^*$ is randomly selected from $[-0.5, 0.5]$ independently. The noise ξ_n's are i.i.d. from $\mathcal{N}(0, \sigma^2)$, and the noise level is measured by σ/E_y, where E_y is the root mean square of the noiseless outputs.

[2]The source codes used are downloaded from https://github.com/VITA-Group/CV_LTH_Pre-training.

4.1 Evaluation of theoretical findings on synthetic data

Local convexity near $\boldsymbol{W}^*$. We set the number of neurons $K = 10$, the dimension of the data $d = 500$ and the sample size $N = 5000$. Figure 2 illustrates the success rate of Algorithm 1 when $\widetilde{r}$ changes. The y-axis is the relative distance of the initialization $\boldsymbol{W}^{(0)}$ to the ground-truth. For each pair of $\widetilde{r}$ and the initial distance, 100 trails are constructed with the network weights, training data and the initialization $\boldsymbol{W}^{(0)}$ are all generated independently in each trail. Each trail is called successful if the relative error of the solution $\boldsymbol{W}$ returned by Algorithm 1, measured by $\|\boldsymbol{W} - \boldsymbol{W}^*\|_F / \|\boldsymbol{W}^*\|_F$, is less than 10^{-4}. A black block means Algorithm 1 fails in estimating $\boldsymbol{W}^*$ in all trails while a white block indicates all success. As Algorithm 1 succeeds if $\boldsymbol{W}^{(0)}$ is in the local convex region near $\boldsymbol{W}^*$, we can see that the radius of convex region is indeed linear in $-\widetilde{r}^{\frac{1}{2}}$, as predicted by Theorem 1.

Convergence rate. Figure 3 shows the convergence rate of Algorithm 1 when $\widetilde{r}$ changes. $N = 5000$, $d = 300$, $K = 10$, $\eta = 0.5$, and $\beta = 0.2$. Figure 3(a) shows that the relative error decreases exponentially as the number of iterations increases, indicating the linear convergence of Algorithm 1. As shown in Figure 3(b), the results are averaged over 20 trials with different initial points, and the areas in low transparency represent the standard deviation errors. We can see that the convergence rate is almost linear in $\sqrt{\widetilde{r}}$, as predicted by Theorem 2. We also compare with GD by setting β as 0. One can see that AGD has a smaller convergence rate than GD, indicating faster convergence.

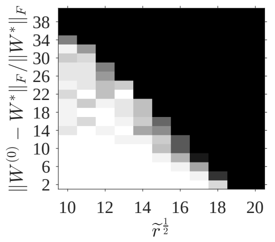

Figure 2: The radius of the local convex region against $\widetilde{r}^{\frac{1}{2}}$

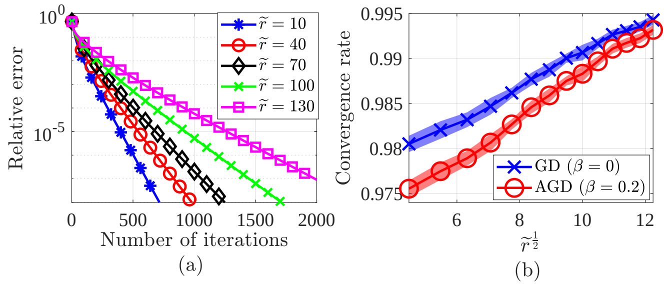

Figure 3: Convergence rate when $\widetilde{r}$ changes

Sample complexity. Figures 4 and 5 show the success rate of Algorithm 1 when varying N and $\widetilde{r}$. d is fixed as 100. In Figure 4, we construct 100 independent trails for each pair of N and $\widetilde{r}$, where the ground-truth model and training data are generated independently in each trail. One can see that the required number of samples for successful estimation is linear in $\widetilde{r}$, as predicted by (8). In Figure 5, r_j is fixed as 20 for all neurons, but different network architectures after pruning are considered. One can see that although the number of remaining weights is the same, $\widetilde{r}$ can be different in different architectures, and the sample complexity increases as $\widetilde{r}$ increases, as predicted by (8).

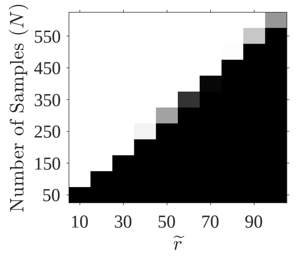

Figure 4: Sample complexity when $\widetilde{r}$ changes

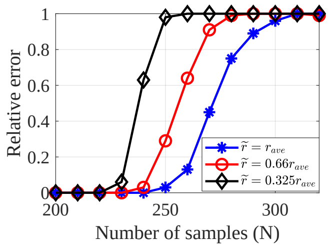

Figure 5: Relative error against $\widetilde{r}$

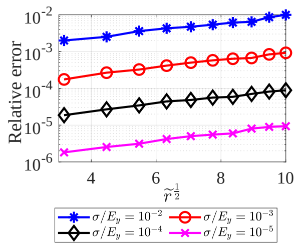

Figure 6: Relative error against $\widetilde{r}^{\frac{1}{2}}$ at different noise level

Performance in noisy case. Figure 6 shows the relative error of the learned model by Algorithm 1 from noisy measurements when $\widetilde{r}$ changes. $N = 1000$, $K = 10$, and $d = 300$. The results are averaged over 100 independent trials, and standard deviation is around 2% to 8% of the corresponding relative errors. The relative error is linear in $\widetilde{r}^{\frac{1}{2}}$, as predicted by (9). Moreover, the relative error is proportional to the noise level $|\xi|$.

4.2 Performance with inaccurate mask on synthetic data

The performance of Algorithm 1 is evaluated when the mask $\boldsymbol{M}$ of the learner network is inaccurate. The number of neurons K is 5. The dimension of inputs d is 100. r_j^* of the oracle model is 20 for all $j \in [K]$. GraSP algorithm [55] is employed to find masks based only on early-trained weights in 20 iterations of AGD. The mask accuracy is measured by $\|\boldsymbol{M}^* \odot \boldsymbol{M}\|_0/\|\boldsymbol{M}^*\|_0$, where $\boldsymbol{M}^*$ is the mask of the oracle model. The pruning ratio is defined as $(1 - r_{\text{ave}}/d) \times 100\%$. The number of training samples N is 200. The model returned by Algorithm 1 is evaluated on $N_{\text{test}} = 10^5$ samples, and the test error is measured by $\sqrt{\sum_n |y_n - \hat{y}_n|^2/N_{\text{test}}}$, where $\hat{y}_n$ is the output of the learned model with the input $\boldsymbol{x}_n$, and $(\boldsymbol{x}_n, y_n)$ is the n-th testing sample generated by the oracle network.

Improved generalization by GraSP. Figure 7 shows the test error with different pruning ratios. For each pruning ratio, we randomly generate 1000 independent trials. Because the mask of the learner network in each trail is generated independently, we compute the average test error of the learned models in all the trails with same mask accuracy. If there are less than 10 trails for certain mask accuracy, the result of that mask accuracy is not reported as it is statistically meaningless. The test error decreases as the mask accuracy increases. More importantly, at fixed mask accuracy, the test error decreases as the pruning ratio increases. That means the generalization performance improves when $\widetilde{r}$ deceases, even if the mask is not accurate.

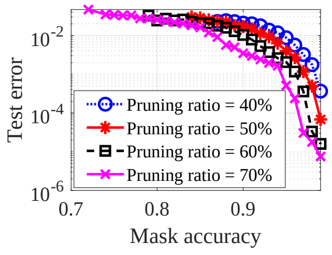

Figure 7: Test error against mask accuracy with different pruning ratios

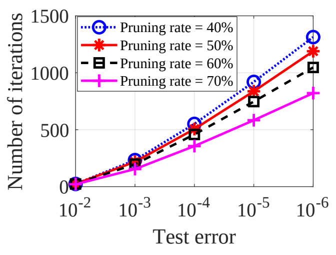

Figure 8: Convergence rate with mask accuracy in $[0.85, 0.9]$

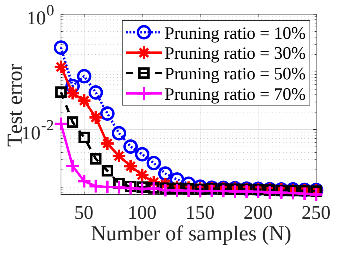

Figure 9: Test error against the number of samples with mask accuracy in $[0.85, 0.9]$

Linear convergence. Figure 8 shows the convergence rate of Algorithm 1 with different pruning ratios. We show the smallest number of iterations required to achieve a certain test error of the learned model, and the results are averaged over the independent trials with mask accuracy between 0.85 and 0.90. Even with inaccurate mask, the test error converges linearly. Moreover, as the pruning ratio increases, Algorithm 1 converges faster.

Sample complexity with respect to the pruning ratio. Figure 9 shows the test error when the number of training samples N changes. All the other parameters except N remain the same. The results are averaged over the trials with mask accuracy between 0.85 and 0.90. We can see the test error decreases when N increases. More importantly, as the pruning ratio increases, the required number of samples to achieve the same test error (no less than 10^{-3}) decreases dramatically. That means the sample complexity decreases as $\widetilde{r}$ decreases even if the mask is inaccurate.

4.3 Performance of IMP on synthetic, MNIST and CIFAR-10 datasets

We implement the IMP algorithm to obtain pruned networks on synthetic, MNIST and CIFAR-10 datasets. Figure 10 shows the test performance of a pruned network on synthetic data with different sample sizes. Here in the oracle network model, $K = 5, d = 100$, and $r_j^* = 20$ for all $j \in [K]$. The noise level $\sigma/E_y = 10^{-3}$. One observation is that for a fixed sample size N greater than 100, the test error decreases as the pruning ratio increases. This verifies that the IMP algorithm indeed prunes the network properly. It also shows that the learned model improves as the pruning progresses, verifying our theoretical result in Theorem 2 that the difference of the learned model from the oracle model decreases as r_j decreases. The second observation is that the test error decreases as N increases for any fixed pruning ratio. This verifies our result in Theorem 2 that the difference of the learned model from the oracle model decreases as the number of training samples increases. When the pruning ratio is too large (greater than 80%), the pruned network cannot explain the data properly, and thus the test error is large for all N. When the number of samples is too small, like $N = 100$, the test error is always large, because it does not meet the sample complexity requirement for estimating the oracle model even though the network is properly pruned.

Figures 11 and 12 show the test performance of learned models by implementing the IMP algorithm on MNIST and CIFAR-10 using Lenet-5 [32] and Resnet-50 [27] architecture, respectively. The experiments follow the standard setup in [10] except for the size of the training sets. To demonstrate the effect of sample complexity, we randomly selected N samples from the original training set without replacement. As we can see, a properly pruned network (i.e., winning ticket) helps reduce the sample complexity required to reach the test accuracy of the original dense model. For example, training on a pruned network returns a model (e.g., P_1 and P_3 in Figures 11 and 12) that has better testing performance than a dense model (e.g., P_2 and P_4 in Figures 11 and 12) trained on a larger data set. Given the number of samples, we consistently find the characteristic behavior of winning tickets: That is, the test accuracy could increase when the pruning ratio increases, indicating the effectiveness of pruning. The test accuracy then drops when the network is overly pruned. The results show that our theoretical characterization of sample complexity is well aligned with the empirical performance of pruned neural networks and explains the improved generalization observed in LTH.

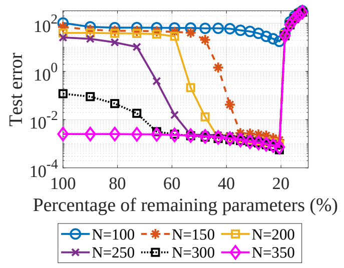

Figure 10: Test error of pruned models on the synthetic dataset

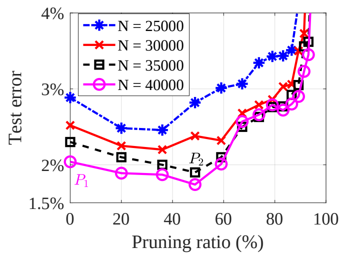

Figure 11: Test accuracy of pruned LeNet-5 on Mnist

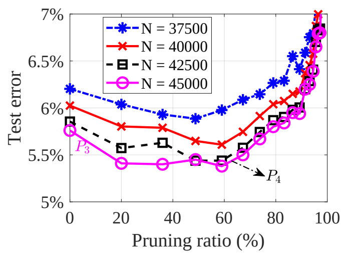

Figure 12: Test accuracy of pruned Resnet-50 on Cifar-10

5 Conclusions

This paper provides the first theoretical analysis of learning one-hidden-layer pruned neural networks, which offers formal justification of the improved generalization of winning ticket observed from empirical findings in LTH. We characterize analytically the impact of the number of remaining weights in a pruned network on the required number of samples for training, the convergence rate of the learning algorithm, and the accuracy of the learned model. We also provide extensive numerical validations of our theoretical findings.

Broader impacts

We see no ethical or immediate societal consequence of our work. This paper contributes to the theoretical foundation of both network pruning and generalization guarantee. The former encourages the development of learning method to reduce the computational cost. The latter increases the public trust in incorporating AI technology in critical domains.

Acknowledgement

This work was supported by AFOSR FA9550-20-1-0122, ARO W911NF-21-1-0255, NSF 1932196 and the Rensselaer-IBM AI Research Collaboration (http://airc.rpi.edu), part of the IBM AI Horizons Network (http://ibm.biz/AIHorizons). We thank Tianlong Chen at University of Texas at Austin, Haolin Xiong at Rensselaer Polytechnic Institute and Yihua Zhang at Michigan State University for the help in formulating numerical experiments. We thank all anonymous reviewers for their constructive comments.

References

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

[2] S. Arora, R. Ge, B. Neyshabur, and Y. Zhang. Stronger generalization bounds for deep nets via a compression approach. In J. Dy and A. Krause, editors, *Proceedings of the 35th International*

Conference on Machine Learning, volume 80 of *Proceedings of Machine Learning Research*, pages 254–263, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[3] C. Baykal, L. Liebenwein, I. Gilitschenski, D. Feldman, and D. Rus. Data-dependent coresets for compressing neural networks with applications to generalization bounds. In *International Conference on Learning Representations*, 2018.

[4] C. Baykal, L. Liebenwein, I. Gilitschenski, D. Feldman, and D. Rus. Sipping neural networks: Sensitivity-informed provable pruning of neural networks. *arXiv preprint arXiv:1910.05422*, 2019.

[5] M. Ben, M. Osadchy, V. Braverman, S. Zhou, and D. Feldman. Data-independent neural pruning via coresets. In *International Conference on Learning Representations (ICLR)*, 2020.

[6] M. L. Bermingham, R. Pong-Wong, A. Spiliopoulou, C. Hayward, I. Rudan, H. Campbell, A. F. Wright, J. F. Wilson, F. Agakov, P. Navarro, et al. Application of high-dimensional feature selection: evaluation for genomic prediction in man. *Scientific reports*, 5(1):1–12, 2015.

[7] R. Bhatia. *Matrix analysis*, volume 169. Springer Science & Business Media, 2013.

[8] A. Brutzkus and A. Globerson. Globally optimal gradient descent for a convnet with gaussian inputs. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 605–614. JMLR. org, 2017.

[9] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud. Neural ordinary differential equations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 6572–6583, 2018.

[10] T. Chen, J. Frankle, S. Chang, S. Liu, Y. Zhang, M. Carbin, and Z. Wang. The lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16306–16316, 2021.

[11] T. Chen, J. Frankle, S. Chang, S. Liu, Y. Zhang, Z. Wang, and M. Carbin. The lottery ticket hypothesis for pre-trained bert networks. *arXiv preprint arXiv:2007.12223*, 2020.

[12] L. Chizat and F. Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 3040–3050, 2018.

[13] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, and N. de Freitas. Predicting parameters in deep learning. In *Proceedings of the 26th International Conference on Neural Information Processing Systems-Volume 2*, pages 2148–2156, 2013.

[14] V. C. Dinh and L. S. Ho. Consistent feature selection for analytic deep neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 2420–2431. Curran Associates, Inc., 2020.

[15] X. Dong, S. Chen, and S. Pan. Learning to prune deep neural networks via layer-wise optimal brain surgeon. In *Advances in Neural Information Processing Systems*, pages 4857–4867, 2017.

[16] S. S. Du, J. D. Lee, Y. Tian, A. Singh, and B. Poczos. Gradient descent learns one-hidden-layer cnn: Don't be afraid of spurious local minima. In *International Conference on Machine Learning*, pages 1338–1347, 2018.

[17] S. S. Du, X. Zhai, B. Poczos, and A. Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International Conference on Learning Representations*, 2019.

[18] B. Elesedy, V. Kanade, and Y. W. Teh. Lottery tickets in linear models: An analysis of iterative magnitude pruning. *arXiv preprint arXiv:2007.08243*, 2020.

[19] U. Evci, T. Gale, J. Menick, P. S. Castro, and E. Elsen. Rigging the lottery: Making all tickets winners. *International Conference on Machine Learning*, 2020.

[20] J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

[21] J. Frankle, G. K. Dziugaite, D. Roy, and M. Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pages 3259–3269. PMLR, 2020.

[22] J. Frankle, G. K. Dziugaite, D. M. Roy, and M. Carbin. Stabilizing the lottery ticket hypothesis. *arXiv preprint arXiv:1903.01611*, 2019.

[23] H. Fu, Y. Chi, and Y. Liang. Guaranteed recovery of one-hidden-layer neural networks via cross entropy. *IEEE transactions on signal processing*, 68:3225–3235, 2020.

[24] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of machine learning research*, 3(Mar):1157–1182, 2003.

[25] S. Han, J. Pool, J. Tran, and W. Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015.

[26] B. Hassibi and D. G. Stork. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in neural information processing systems*, pages 164–171, 1993.

[27] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[28] H. Hu, R. Peng, Y.-W. Tai, and C.-K. Tang. Network trimming: A data-driven neuron pruning approach towards efficient deep architectures. *arXiv preprint arXiv:1607.03250*, 2016.

[29] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

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

[31] V. Kuleshov, A. Chaganty, and P. Liang. Tensor factorization via matrix factorization. In *Artificial Intelligence and Statistics*, pages 507–516, 2015.

[32] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[33] Y. LeCun, J. S. Denker, and S. A. Solla. Optimal brain damage. In *Advances in neural information processing systems*, pages 598–605, 1990.

[34] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012.

[35] L. Liebenwein, C. Baykal, H. Lang, D. Feldman, and D. Rus. Provable filter pruning for efficient neural networks. In *International Conference on Learning Representations*, 2019.

[36] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2018.

[37] R. Livni, S. Shalev-Shwartz, and O. Shamir. On the computational efficiency of training neural networks. In *Advances in neural information processing systems*, pages 855–863, 2014.

[38] Y. Lu, C. Ma, Y. Lu, J. Lu, and L. Ying. A mean field analysis of deep resnet and beyond: Towards provably optimization via overparameterization from depth. In *International Conference on Machine Learning*, pages 6426–6436. PMLR, 2020.

[39] E. Malach, G. Yehudai, S. Shalev-Schwartz, and O. Shamir. Proving the lottery ticket hypothesis: Pruning is all you need. In *International Conference on Machine Learning*, pages 6682–6691. PMLR, 2020.

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

[41] D. Molchanov, A. Ashukha, and D. Vetrov. Variational dropout sparsifies deep neural networks. In *International Conference on Machine Learning*, pages 2498–2507, 2017.

[42] B. Neyshabur. Towards learning convolutions from scratch. *Advances in Neural Information Processing Systems*, 33, 2020.

[43] L. Orseau, M. Hutter, and O. Rivasplata. Logarithmic pruning is all you need. *Advances in Neural Information Processing Systems*, 33, 2020.

[44] S. Oymak and M. Soltanolkotabi. Toward moderate overparameterization: Global convergence guarantees for training shallow neural networks. *IEEE Journal on Selected Areas in Information Theory*, 1(1), 2020.

[45] V. Ramanujan, M. Wortsman, A. Kembhavi, A. Farhadi, and M. Rastegari. What's hidden in a randomly weighted neural network? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11893–11902, 2020.

[46] A. Renda, J. Frankle, and M. Carbin. Comparing rewinding and fine-tuning in neural network pruning. In *International Conference on Learning Representations*, 2019.

[47] I. Safran and O. Shamir. Spurious local minima are common in two-layer relu neural networks. In *International Conference on Machine Learning*, pages 4430–4438, 2018.

[48] O. Shamir. Distribution-specific hardness of learning neural networks. *The Journal of Machine Learning Research*, 19(1):1135–1163, 2018.

[49] S. Singh and A. Majumdar. Deep sparse coding for non–intrusive load monitoring. 9(5):4669–4678, Feb. 2017.

[50] M. Soltanolkotabi, A. Javanmard, and J. D. Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 65(2):742–769, 2018.

[51] S. Srinivas and R. V. Babu. Data-free parameter pruning for deep neural networks. *arXiv preprint arXiv:1507.06149*, 2015.

[52] Y. Tian. An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3404–3413. JMLR. org, 2017.

[53] J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics*, 12(4):389–434, 2012.

[54] R. Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[55] C. Wang, G. Zhang, and R. Grosse. Picking winning tickets before training by preserving gradient flow. In *International Conference on Learning Representations*, 2019.

[56] F. Wang, K. Li, C. Liu, Z. Mi, M. Shafie-Khah, and J. P. S. Catalão. Synchronous pattern matching principle-based residential demand response baseline estimation: Mechanism analysis and approach description. *IEEE Transactions on Smart Grid*, 9(6):6972–6985, 2018.

[57] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li. Learning structured sparsity in deep neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 2082–2090, 2016.

[58] T.-J. Yang, Y.-H. Chen, and V. Sze. Designing energy-efficient convolutional neural networks using energy-aware pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5687–5695, 2017.

[59] M. Ye, C. Gong, L. Nie, D. Zhou, A. Klivans, and Q. Liu. Good subnetworks provably exist: Pruning via greedy forward selection. In *International Conference on Machine Learning*, pages 10820–10830. PMLR, 2020.

[60] M. Ye and Y. Sun. Variable selection via penalized neural network: a drop-out-one loss approach. In *International Conference on Machine Learning*, pages 5620–5629. PMLR, 2018.

[61] M. Ye, L. Wu, and Q. Liu. Greedy optimization provably wins the lottery: Logarithmic number of winning tickets is enough. *Advances in Neural Information Processing Systems*, 33, 2020.

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

[63] H. You, C. Li, P. Xu, Y. Fu, Y. Wang, X. Chen, R. G. Baraniuk, Z. Wang, and Y. Lin. Drawing early-bird tickets: Toward more efficient training of deep networks. In *International Conference on Learning Representations*, 2019.

[64] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

[65] S. Zhang, M. Wang, S. Liu, P.-Y. Chen, and J. Xiong. Fast learning of graph neural networks with guaranteed generalizability:one-hidden-layer case. In *2020 International Conference on Machine Learning (ICML)*, 2020.

[66] S. Zhang, M. Wang, S. Liu, P.-Y. Chen, and J. Xiong. Guaranteed convergence of training convolutional neural networks via accelerated gradient descent. In *2020 54th Annual Conference on Information Sciences and Systems (CISS)*, 2020.

[67] S. Zhang, M. Wang, J. Xiong, S. Liu, and P.-Y. Chen. Improved linear convergence of training cnns with generalizability guarantees: A one-hidden-layer case. *IEEE Transactions on Neural Networks and Learning Systems*, 32(6):2622–2635, 2020.

[68] L. Zhao, Q. Hu, and W. Wang. Heterogeneous feature selection with multi-modal deep neural networks and sparse group lasso. *IEEE Transactions on Multimedia*, 17(11):1936–1948, 2015.

[69] K. Zhong, Z. Song, P. Jain, P. L. Bartlett, and I. S. Dhillon. Recovery guarantees for one-hidden-layer neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 4140–4149. JMLR. org, https://arxiv.org/abs/1706.03175, 2017.

[70] H. Zhou, J. Lan, R. Liu, and J. Yosinski. Deconstructing lottery tickets: Zeros, signs, and the supermask. In *Advances in Neural Information Processing Systems 32*, pages 3597–3607. 2019.

[71] W. Zhou, V. Veitch, M. Austern, R. P. Adams, and P. Orbanz. Non-vacuous generalization bounds at the imagenet scale: a pac-bayesian compression approach. In *International Conference on Learning Representations*, 2018.

[72] D. Zou, Y. Cao, D. Zhou, and Q. Gu. Gradient descent optimizes over-parameterized deep relu networks. *Machine Learning*, 109(3):467–492, 2020.

[73] D. Zou and Q. Gu. An improved analysis of training over-parameterized deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2055–2064, 2019.