

Identifying Winning Lottery Tickets in Small Neural Network Architectures

Murat Aksu
STUDENT NUMBER: 2046041

THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN DATA SCIENCE & SOCIETY
DEPARTMENT OF COGNITIVE SCIENCE & ARTIFICIAL INTELLIGENCE
SCHOOL OF HUMANITIES AND DIGITAL SCIENCES
TILBURG UNIVERSITY

Thesis committee:
Prof. Dr. Eric Postma
Dr. Juan Sebastian Olier

Tilburg University
School of Humanities and Digital Sciences
Department of Cognitive Science & Artificial Intelligence
Tilburg, The Netherlands
May 2020

Preface

I wish to thank the European Union's Jean Monnet Scholarship Program (<https://www.jeanmonnet.org.tr/en-us/>) for giving me this opportunity to improve myself and contribute to the strategic priority of more understandable Artificial Intelligence.

Also, I wish to thank Arjun Majumdar, his work (https://github.com/arjun-majumdar/Lottery_Ticket_Hypothesis-TensorFlow_2) was inspirational for this study.

Identifying Winning Lottery Tickets in Small Neural Network Architectures

Murat Aksu

Deep Learning is a breakthrough in the field of Artificial Intelligence (AI) with human-like (or better) performance in various challenging tasks. Deep Neural Networks (DNNs) are large and complex network architectures, which are difficult to interpret and computationally expensive. To tackle these problems, pruning the network is a common practice.

Frankle and Carbin (2018) proposed the lottery ticket hypothesis, which suggests that even 90% pruned sub-networks (winning tickets) can exceed the performance of the original unpruned network if the weights are initialized as in the original network. However, finding these winning tickets is computationally expensive. One possible remedy is to use a smaller network (lesser units in hidden layers) from the start to find these winning tickets. We introduce a new concept as super winning tickets, a winning ticket generated from a smaller version of the original network that can surpass the performance of the original unpruned network. The research question addressed in this thesis reads: to what extent is it possible to generate super winning tickets using the methodology of Frankle and Carbin (2018)? Surprisingly, we found that up to 70% smaller version of the Lenet 300-100 architecture, we can find these super winning tickets on the MNIST dataset. We discuss the implications of these findings and conclude that it is possible to find super winning tickets within even much smaller fully-connected networks.

1. Introduction

Artificial intelligence (AI) is the main topic emphasized under the European Union's *A Europe fit for the digital age* strategic priority (Leyen 2019). Its importance is not just limited to the businesses but also the governmental services as well (European Commission 2020a). To benefit from and improve the AI ecosystem, European-wide considerable investments are taking place.

Deep Neural Networks (DNNs) are at the backbone of this AI revolution. DNNs are achieving state-of-the-art performance (even human-like or better) on various complex tasks, such as image classification, speech recognition, and machine translation. Nevertheless, due to the complexity and the vast number of free parameters in DNNs, the reason for their success is not well understood, preventing its more extensive usage. In particular, explainability, i.e., the ability of clearly comprehending the decision-making mechanism behind the DNNs, is required to reach its full potential and to use it on governmental services or official statistics (European Commission 2020b).

The empirical evidence suggests that larger and deeper DNNs learn better from the data (e.g., He et al. (2015)). However, the increased size of the networks damages the interpretability and increases the computational requirement. The pruning of networks is a common procedure to reduce the size of trained networks (Han, Mao, and Dally 2015; Li et al. 2016). In a standard pruning procedure, DNNs are first trained with

millions of parameters, and then the unnecessary weights¹ are pruned away to reduce the size of the network. These sparsified networks are easier to interpret, require less memory, and are computationally more efficient. Various studies suggest that networks can be pruned up to a reduction of 90% of the weights without losing significant accuracy (Han et al. 2015; Narang et al. 2017).

Despite its broad application, the brute-force reduction of network size by means of pruning seems inefficient, like cracking a nut with a sledgehammer first and then to change to a smaller tool that can crack a nut just as well. Naturally, the question arises: Why not using the more appropriate tool from the start? Frankle and Carbin (2018) posed this question and investigated the possibility of finding smaller and optimal networks before training. They proposed the *lottery ticket hypothesis*, which states that randomly initialized, dense neural networks contain a large number of smaller sub-networks, the performances of which can reach a similar or even better level of performance as the full network when trained in isolation with their initial original weights. These smaller sub-networks are identified as *winning tickets* that have won the initialization lottery (Frankle and Carbin 2018). If the lottery ticket hypothesis applies, we can achieve similar performances with networks that are significantly reduced in the number of parameters more efficiently.

If we can identify these optimal sub-networks (winning tickets) from the start, we can build much computationally efficient and more interpretable DNNs with lesser memory requirements. These sub-networks will be more understandable with fewer parameters and will be energy-efficient so that we can use it with more extensive datasets without needing super-computers.

Still, finding these winning tickets is computationally expensive due to the *iterative magnitude pruning process* (See Section 4.1), which requires 20 or more² consecutive training cycles. The theory supporting the lottery ticket hypothesis suggests that the larger the size of the DNNs, the higher the probability of containing the winning ticket (Frankle and Carbin 2018). The intuitive explanation is that larger DNNs have a larger number of sub-networks and hence a larger probability of having a useful sub-network for solving the task. This would imply that smaller networks are less likely to contain winning tickets. Yet, smaller networks do yield fruitful results. For instance, Frankle and Carbin (2018) conducted experiments with smaller network sizes, and they managed to find winning tickets in smaller versions of the *Lenet 300-100 architecture* (Lecun et al. 1998). These small-network results suggest that the lottery ticket hypothesis applies to small networks as well.

However, experiments of the Frankle and Carbin (2018) with smaller network sizes were in a relatively narrow scope. It was only side research of their work, and the question they particularly investigated was if the bigger networks find better winning tickets compared to the smaller ones. They experimented with bigger and smaller versions of the *Lenet 300-100 architecture* on the MNIST dataset (LeCun and Cortes 2010). They found that the winning tickets generated from the bigger versions achieve higher test accuracy while winning tickets generated from smaller versions reaches minimum validation loss faster (in earlier iteration).

Since the actual goal of Frankle and Carbin (2018) was not to experiment with the smaller networks specifically, they experimented only with five different smaller

1 There are various pruning methods. In this thesis we will refer to pruning to mean *magnitude pruning* which removes the certain percentage of the smallest weights after training.

2 Number of iterative rounds is dependent on the *pruning rate* and the *target sparsity*. Lower pruning rates and lower target sparsity will require more rounds.

versions of the network without experimenting with different configuration. They did not experiment with different methods or strategies with smaller network sizes. In this thesis, we especially experimented with different pruning strategies and methods in order to dig deeper and to find more effective configuration. We also experimented on eleven different network sizes to have more stable results. Besides, they only compared the winning tickets with each other but not with the original network. We think it is crucial to compare the performance of the winning ticket generated from the smaller network with the original unpruned network.

Unique to this work, we introduce a new concept as *super winning tickets*. These are the winning tickets generated from the smaller version of the original network that surpasses the performance of the original unpruned network. As an example, we consider the Lenet 300-100 architecture, that is reduced the number of units in hidden layers by 50% yielding, what we will henceforth refer to as a *Lenet 150-50 architecture*. If a winning ticket generated from Lenet 150-50 (total 200 units in hidden layers) reaches the same or higher test accuracy as the original Lenet 300-100 (total 400 units in hidden layers) at the same or fewer number of epochs of training, it qualifies as a super winning ticket.

These super winning tickets are particularly exciting because it means that the computationally expensive winning ticket generation process can be applied from the start to a much smaller network with quicker computation time and substantially lesser memory requirement!

The research question addressed in this thesis reads as follows.

To what extent is it possible to find super winning tickets in smaller successfully trained fully-connected deep learning architectures using the pruning methodology of [Frankle and Carbin \(2018\)](#)?

For the rest of the thesis, we will refer to Lenet 300-100 architecture (See Section 4) as an *original unpruned network*. Throughout the study, we experimented with the smaller versions of this architecture³ on the MNIST dataset.

2. Related Work

Removing (pruning) the certain weights in the neural networks is not a new idea and can be dated back to the 1990s ([LeCun, Denker, and Solla 1990](#); [Hassibi and Stork 1993](#)). Researchers tried pruning in order to improve generalization, mitigate interpretability, increase the training speed, and reduce the size and the storage requirement of the networks. The idea of pruning the networks for efficiency is relatively longstanding; however, it did not gain popularity until recently. The current surge of the deeper and more complex DNN's created the need for optimization. The unmanageable sizes of modern DNN's urged researchers to search for a solution.

Our work can be directly rooted back to the work of the [Han et al. \(2015\)](#). They presented a new *three-step magnitude pruning* method that trains the networks, prunes the low magnitude weights, and retrains the pruned (sparse) network. The final step mentioned is vital; without it, there is a significant loss in the accuracy. In the original method ([Han et al. 2015](#)), weights are randomly initiated each round. The main contribution of [Frankle and Carbin \(2018\)](#) was that they modified this pruning methodology by *resetting the weights to their initial values* after the pruning and subsequently retrained the network. They have proposed the famous lottery ticket hypothesis, which this

³ Unit sizes in hidden layers are gradually reduced in each round of experiment.

research is directly inspired. After the publication of the original report (Frankle and Carbin 2018), researchers performed follow-up experiments in order to understand the nature of these winning tickets and the reasons for their success.

The methodology of Frankle and Carbin (2018) failed to find winning tickets in deeper neural networks. Frankle et al. (2019) further developed⁴ the iterative magnitude pruning technique to find winning tickets in deeper networks on much extensive and challenging datasets such as ImageNet (Deng et al. 2009).

Morcos et al. (2019) tried to address the computational intensiveness problem by reusing the already generated winning tickets across a variety of datasets and optimizers. According to the findings of the authors, these winning tickets are relatively dataset- and optimizer-independent, which demonstrate their *generic inductive bias*. They also found that winning tickets generated from larger datasets generalize better. Their finding is exciting since it indicates that these predefined winning tickets can be used on different datasets from the start without the lengthy iterative magnitude pruning process. Then again, this approach has its limitations. As an example, the same network architecture⁵ has to be employed to use the winning tickets that generated from other datasets, and they have only evaluated the transfer of these winning tickets in the same domain (natural image classification). Yu et al. (2019) investigated the *transferability* of the winning tickets, they have reported that the winning tickets are not solely successful in *supervised learning* of natural images, but in different tasks as well, such as in linguistic tasks and *reinforcement learning*. These results suggest that the lottery ticket hypothesis is not just limited in a specific domain, but it has potentially more extensive applications—possibly a general phenomenon in deep learning.

Perhaps the most inspirational work that aims to understand the nature of the winning tickets was conducted by Zhou et al. (2019). They contributed to the work by demonstrating the importance of the three critical components: *signs* (signs of the initial weights), *setting weights to zero*⁶, and the effect of *masks*⁷ over the performance of the winning tickets. According to the study, the signs are the only significant factor of the initial weights affecting the performance of the winning tickets. This is a crucial finding for our study since we will be using gradually lesser initial weights; our probability of getting the right sign will remain the same as 50%. The other exciting outcome of the study was the effectiveness of the network architectures over performance. As a result of pruning and blocking certain connections (with masks), the architecture of the network changes. Authors achieved impressive accuracies just by using masks, not even training the network⁸, emphasizing the effect of the network architecture. Gaier and Ha (2019) also focused on network architecture; they tried to optimize the network architecture rather than the weights. They used the same weight (no training and weight optimization) selected from a *uniform distribution* for all the networks and achieved an impressive test accuracy of 92% on the MNIST dataset. These studies have

4 In original work of Frankle and Carbin (2018), in the beginning of every iterative magnitude pruning rounds, weights are reset to their initial values (iteration 0). However, Frankle et al. (2019) updated this method to reset the weights to the values at iteration k early in the training rather than at iteration 0. They called this method *rewinding*.

5 Even though the dataset is different, the exact network architecture has to be employed. E.g., winning ticket generated on Lenet 300-100 architecture can solely be used on the same architecture.

6 Zhou et al. (2019) argue that setting pruned weights to the zero is an important factor of success because these weights would headed up towards to the zero anyways.

7 Masks are used to block certain connections (in this case pruned parameters) in the neural networks and to modify its structure. They play essential role in the lottery ticket hypothesis.

8 Without training means using random weights in the neural network without optimization.

demonstrated the importance of the network architectures over the performance of the neural networks. These are also crucial findings for this research because we can achieve the same architectures - for the fully-connected network architectures- with the smaller versions of the neural networks as the original ones. For instance, 90% pruned Lenet 300-100 network (unit size of the first hidden layer is 300, and the second hidden layer is 100), and 80% pruned Lenet 150-50 network can have the same architecture of 30-10 since all the units are connected.

As a continuation work of the Zhou et al. (2019), and much inspired by the Gaier and Ha (2019), Ramanujan et al. (2019) stated that the stochastic nature of mask selection might be a limiting factor, thus, developed an algorithm that finds more effective sub-networks, which achieves superior performance.

The theory supporting the lottery ticket hypothesis suggests that the *over-parametrized neural networks* have a higher probability of containing the winning ticket (Frankle and Carbin 2018). The reasoning is that larger DNNs have a larger number of sub-networks and hence a larger probability of having a useful sub-network for solving the task. However, based on empirical evidence (Zhou et al. 2019; Gaier and Ha 2019; Ramanujan et al. 2019), we know that the architecture of the neural network is the most important component of performance of the winning tickets, and we can achieve the same architectures in the smaller fully-connected networks. Thus, we constructed our intuition that the *over-parametrization of the neural networks* should not be required (Oymak and Soltanolkotabi 2020) to find optimal sub-networks. Using smaller neural networks from the start could give similar results.

3. Super Winning Tickets

Frankle & Carbin described winning tickets as "a sub-network that is initialized such that—when trained in isolation—it can match the test accuracy of the original network after training for at most the same number of iterations" 2018, p. 2. In other words, sub-networks should reach the same or higher test accuracy as the original network in the same or less amount of training time in order to be identified as a winning ticket.

More formally, consider a fully connected network architecture as f , which is optimized at minimum validation loss at iteration j with test accuracy a . Also, consider f_s as a sub-network (pruned with a mask) of f with fewer parameters and connections. With the same initial weights and with the same dataset, f_s optimizes at minimum validation loss at j_s with a test accuracy of a_s . A sub-network f_s identified as a winning ticket if it optimizes at minimum validation loss at j_s that is lesser or equal to j with a_s that is greater or equal to a (Frankle and Carbin 2018).

We have employed the same methodology to decide whether the sub-network identified as a winning ticket. Unique to this work, we introduced a new concept as *super winning tickets*. These are the winning tickets generated from the smaller version of the original network that surpasses the performance of the unpruned original network. For example, if a winning ticket generated from Lenet 150-50 reaches the same or higher test accuracy as the original unpruned Lenet 300-100 at the same or lesser iteration, it qualifies as a super winning ticket.

Super winning tickets are the main focus of this thesis. Throughout the study, we try to explore different configurations to find more super winning tickets. We also investigate the properties of the winning tickets in order to understand them better.

3.1 (Super) Winning Ticket Percentage

This research aims to find out to what extent smaller neural networks contain (super) winning tickets. In order to contextualize which network contains more winning tickets, we introduce the *winning ticket percentage* metric. During the iterative pruning process -depending on the learning rate and the target sparsity- 20 or more consecutive training rounds are required. On these rounds, we decide if the pruned sub-network is a winning ticket or not. The number of winning tickets that are found divided by the total number of the iterative pruning rounds will give the winning ticket percentage of that network size. For instance., 10 winning tickets found over 20 iterative pruning rounds on Lenet 150-50 architecture will give us %50 winning ticket percentage for that network size. We will use this metric to compare the performances of different neural network sizes, strategies, and pruning rates. The higher winning ticket percentage will indicate more winning tickets contained in the neural network. The same rationality applies for the *super winning ticket percentage*.

4. Experimental Setup

As an *original model*, we used the fully-connected Lenet 300-100 architecture (Lecun et al. 1998). Lenet 300-100 has two fully connected hidden layers; the first layer consists of 300 units, the second layer consists of 100 units and the output layer of 10 units. *Rectified Linear Units* (ReLU) is used as an activation function for the hidden layers, and *Softmax* as a classification function.

The MNIST dataset (LeCun and Cortes 2010) of handwritten digits have a training set of 60.000 examples in 10 classes, and a test set of 10.000 examples. 5.000 examples are randomly selected from the training set to be used as a validation set, and the remaining 55.000 examples in the training set is used to train the network. We selected this validation set at the beginning of the experiment. *Validation loss* for *early stopping criterion* (See Section 4.2) is determined on the validation set, while *test accuracy* is evaluated on the test set. 60 mini-batches are used to train the network, and the whole dataset is shuffled after each epoch.

Frankle and Carbin (2018) used this dataset in the original paper for conducting experiments in fully connected architectures. We choose this dataset in this research to be able to have comparable and consistent results with the original work. The MNIST dataset is publicly available and widely used in scientific research in the field of machine learning.

We used Adam (Kingma and Ba 2015) with a 0.0012 *learning rate* as an *optimizer* following the original work (Frankle and Carbin 2018). They selected Adam (0.0012) because, it was highly performant on the original unpruned network as there was no lottery ticket experiment, and it remained the high validation accuracy even in the severely pruned networks. According to Frankle and Carbin (2018), various initialization distributions with different standard deviations performed well in generating winning tickets. They selected Gaussian (normal) Glorot (Glorot and Bengio 2010) as an *initializer* in order to be generic as possible. In this way, the study would not be biased through hand-picked hyperparameters. We shared their rationale and decided to use Gaussian Glorot initialization throughout the study as well.

This study concentrates on experimenting on the smaller versions of the Lenet 300-100 architecture. Only the unit sizes in hidden layers reduced while keeping the other features of the Lenet 300-100 same. We decided to keep the three to one ratio of the Lenet 300-100. Each of the 11 rounds, the number of units remaining from the previous

round are removed by 20%. It is like an iterative pruning process applied to the units rather than weights. The training process on each round is independent of another. We picked this method to reduce the size of the network because this approach is more concentrated on smaller networks compared to the equal size removing each round. We opted for 11 rounds because after that round, during the iterative pruning process, parameters in the second hidden layer converge to zero; besides, the three to one ratio is deteriorated.

For each of the 11 smaller versions of the original network, we operated the iterative magnitude pruning process in order to find winning tickets. Simple *layer-wise pruning* is employed in this work; each layer is pruned independently. 20% (See Section 4.1.1) of the weights with the lowest magnitudes are removed in each round (Han et al. 2015). The layer connected to the output is pruned half of this rate since it already consists of much fewer parameters (0.38% of the total parameters) compared to the full network, higher pruning rate could affect the performance. We selected 0.036 as a final sparsity for our pruned networks. According to Frankle and Carbin (2018), after that level of sparsity, the performance of the sub-networks starts to regress. With a 0.2 pruning rate and 0.036 level of target sparsity, 15 rounds of iterative magnitude pruning process are conducted (for 0.1 pruning rate (See Section 4.1.1, 32 rounds are required). We computed the winning ticket percentage and super winning ticket percentages over these rounds.

We tested different iterative pruning strategies (See Section 4.1) and pruning rates. We conducted every experiment over five rounds; we took *average*, *minimum*, and *maximum* values to use in the results. For all the figures in this study, lines demonstrate the average value of these rounds, while error bars indicate maximum and minimum values.

In every round, we used a specific *random seed* to make the results reproducible. We have opted for five rounds; firstly, because Frankle and Carbin (2018) used the same methodology. Secondly, each round consists of 600 times training the network, so given the time constraint, it would be inefficient otherwise. The network is trained over 3,400 times during the experiments.

We make our code publicly available at <https://github.com/Mraksu/Winning-Tickets-in-Smaller-Neural-Networks>.

4.1 Iterative Magnitude Pruning

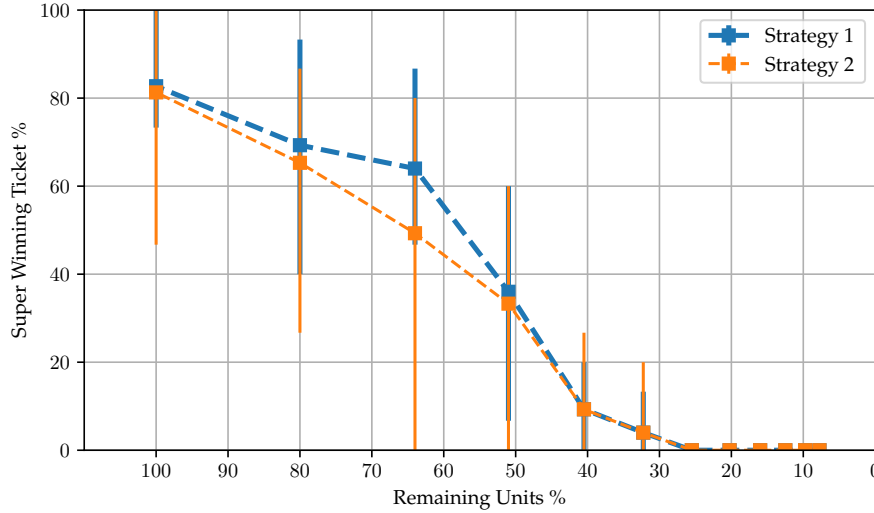
In *one-shot magnitude pruning*⁹, the network is trained till convergence, the smallest $p\%$ of the weights are pruned, and the remaining weights are reset to their initial values. In iterative pruning, this process is repeated over n rounds; each round, remaining weights are pruned, reset, and trained again. Although it is possible to discover winning tickets with the one-shot pruning approach, Frankle and Carbin (2018) suggested that iterative pruning approach is more successful at finding winning tickets in smaller network sizes. Throughout this research, we utilized the iterative pruning method, since our goal is to find winning tickets in smaller network architectures. We did not experiment with any other pruning method than the magnitude pruning in this study since Zhou et al. (2019) already experimented with eight other different methods concluding that the magnitude pruning is superior over others.

In the original paper (Frankle and Carbin 2018), the authors tested two different iterative pruning strategies. The first one is the original version, which is described in

⁹ Not the original version of Han et al. (2015), the modified version by Frankle and Carbin (2018)

Figure 1

Different iterative pruning strategies experimented on the MNIST dataset with smaller sizes of Lenet 300-100. Percentage of units remaining in hidden layers of Lenet 300-100 demonstrated on X-axis, super winning ticket percentage shown on the Y-axis. In smaller sizes up to %40, both strategies performing equally; however, Strategy 1 is superior overall.



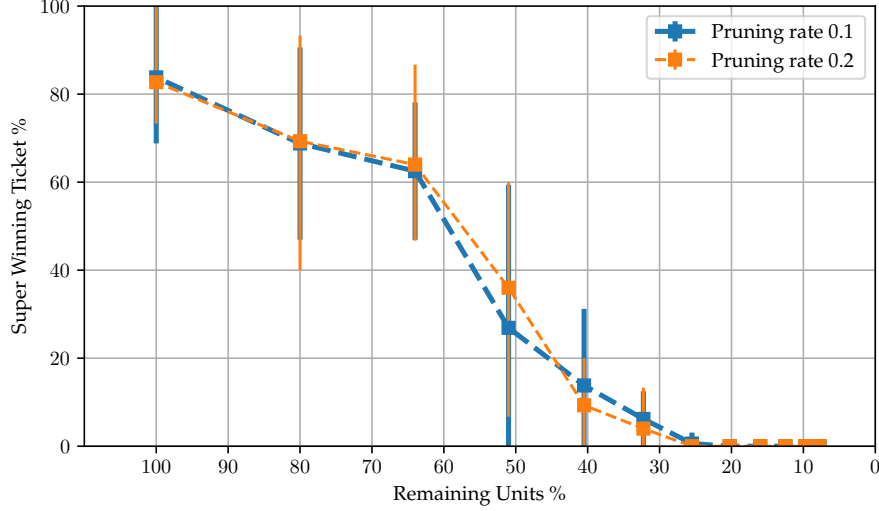
the first paragraph of this section. The only difference in the second strategy is that, after the pruning, already trained weights trained again to reach a certain level of sparsity. In both strategies, weights are reset to their initial values after the pruning and trained again till convergence. In the original paper, the authors proposed *Strategy 1* as superior at finding winning tickets in smaller sizes of the network. We used the same methodology for Strategy 1. However, for *Strategy 2*, we have used the Tensorflow Model Optimization API (https://www.tensorflow.org/model_optimization), which trains and prunes the already trained network to certain sparsity. However, this strategy can be seen as an *iterative one-shot pruning* approach rather than classical iterative pruning because this algorithm uses the already pruned weights (zeros) while training. So, it reaches the same level of sparsity as Strategy 1, but by pruning different weights each round. We have decided to experiment with this novel method as well since it is different from the original one and might give contradicting results.

Figure 1 compares the performance of both strategies on smaller versions of the Lenet 300-100 on the MNIST dataset. The networks pruned with Strategy 1 contain a higher super winning ticket percentage compared to Strategy 2; thus, we choose Strategy 1 for the rest of the study.

4.1.1 Pruning Rate. In the original work, Frankle and Carbin (2018) experimented with various *pruning rates*. They found that the lower pruning rates find better performing winning tickets on smaller network sizes. On the other hand, lower pruning rates require more iterative pruning rounds, which means more computational requirements. So, they decided to use a 0.2 pruning rate, which can achieve similar results with a 0.1 pruning rate within lesser rounds.

Figure 2

Different pruning rates experimented for iterative magnitude pruning process with smaller versions of Lenet 300-100 architecture. Up to 50% smaller network sizes (X-axis), iterative pruning process with pruning rate 0.2 contains more super winning tickets (Y-axis), for smaller network sizes, pruning rate 0.1 has the advantage.



Since our study tries to find winning tickets in smaller networks, we also experimented with the 0.1 pruning rate. Figure 2 and Figure 3 show the performance of both pruning rates on finding winning tickets and super winning tickets. Pruning rate of 0.1 has better performance on smaller network sizes for finding super winning tickets; on the other hand, the pruning rate of 0.2 is better at finding winning tickets overall. Since it takes 32 iterative pruning rounds for 0.1 pruning rate and 15 rounds for the 0.2 pruning rate, we used 0.2 for the rest of the paper to be more computationally efficient.

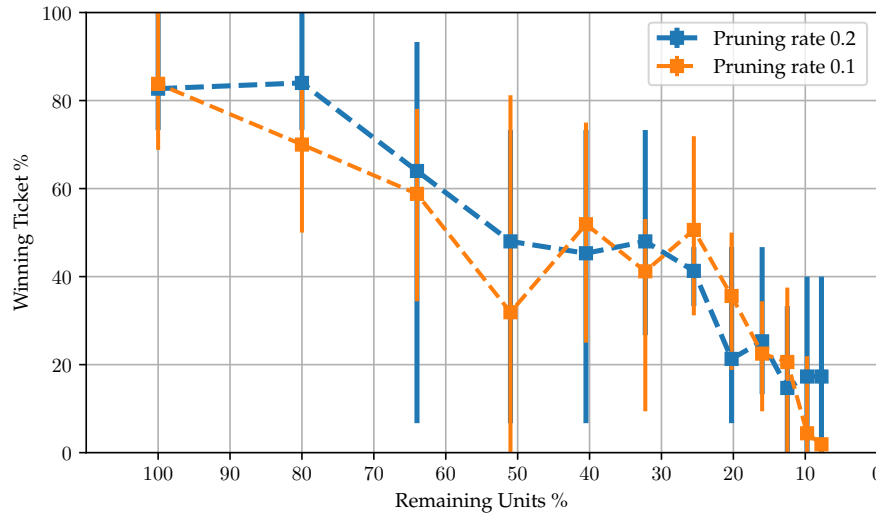
It is important to note that [Frankle and Carbin \(2018\)](#) also opted for Strategy 1 and 0.2 pruning rate setup for their iterative magnitude process as a result of their experiments. We confirm their rationale with our empirical study.

4.2 Early Stopping Criterion

One of the measures of identifying a winning ticket is the speed of learning. In this research, we used the same *early stopping criterion* as the [Frankle and Carbin \(2018\)](#). Validation loss is used as a metric to determine this criterion. The lowest point of validation loss reached, considered as where the training would stop. We consider a sub-network (winning ticket) that reaches this minimum validation loss in an earlier iteration as faster. The test accuracy is evaluated at the corresponding iteration at this early stopping criterion occurs. The validation loss is evaluated in every 100 iterations.

Figure 3

The same experiment conducted as Figure 2, this time winning ticket percentage (Y-axis), is evaluated. Overall, the 0.2 pruning rate performed better in a more stable manner in different sizes of the network.



In the original work, the network is trained for 50,000 iterations¹⁰ (50 epochs), and the early stopping criterion (the minimum validation loss reached) is determined retrospectively. We also determined the early stopping criterion retrospectively after the training is completed. Unique to our case, training the network for so long is not computationally feasible due to time and hardware constraints. Throughout this work, we stopped the training after five consecutive epochs if the validation loss is not mitigated by more than 0.001.

4.3 Software

We used Python 3.7.7 (<https://www.python.org/>) as the main programming language throughout the study. All the other software used is built upon Python. We built models with the Sequential library of the Keras 2.3.1 (<https://keras.io/>) with Tensorflow 2.1.0 (<https://www.tensorflow.org/>) backend. We used Gradient Tape API (https://www.tensorflow.org/api_docs/python/tf/GradientTape) to calculate the gradients of the neural networks and multiplied the gradients with the mask. For most of the calculations, we used NumPy 1.18.1 (<https://numpy.org/>). For storing the raw outputs, JSON 2.0.9 (<https://www.json.org/json-en.html>) is used. We used Pandas 1.0.3 (<https://pandas.pydata.org/>) for analyzing the results, and Matplotlib 3.2.1 (<https://matplotlib.org/>) for visualizations.

¹⁰ One iteration means training of one mini-batches conducted with 60 training example, one epoch means thousand iterations over all the dataset.

Table 1

15 rounds of iterative pruning process with Strategy 1 and Pruning Rate 0.2 applied on the MNIST dataset with smaller sizes of Lenet 300-100. Values shown are averaged over 5 runs. "First Layer" and "Second Layer" columns demonstrates the number of the units in the hidden layers. In third column "WT" stands for winning tickets.

First Layer	Second Layer	Super WT %
300	100	83
240	80	69
192	64	64
153	51	36
122	40	9
97	32	4
77	25	0
61	20	0
48	16	0
38	12	0
30	9	0
24	7	0

5. Results

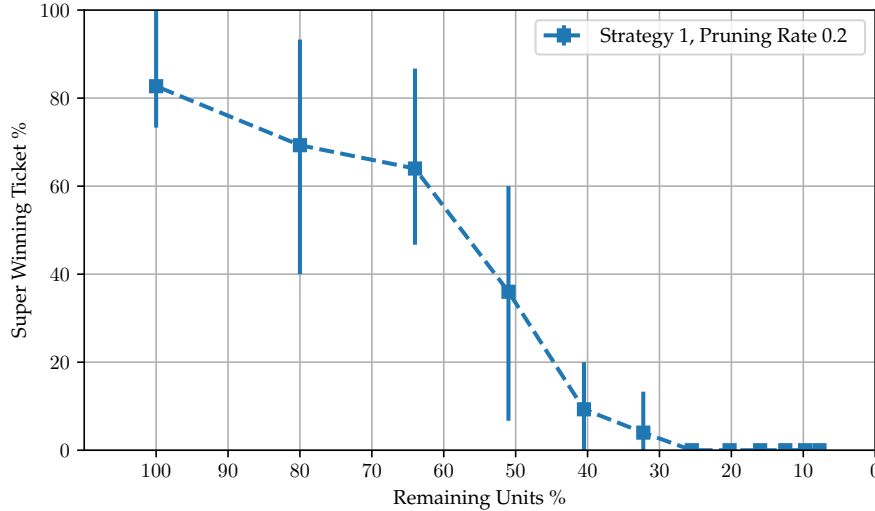
We identified super winning tickets up to 70% smaller version of the Lenet 300-100 architecture on the MNIST dataset. Table 1 shows the levels of super winning ticket percentage in various sizes of the Lenet 300-100 architecture. We found that 69% of the iterative pruning rounds applied on Lenet 240-80 contain super winning ticket. The rate drops to 4% for the Lenet 97-32. We can see a clear trend (See Figure 4) that the larger networks contain more super winning tickets compared to the smaller versions. After the 70% smaller networks (Lenet 97-32), we could not find any more super winning tickets.

As for the winning tickets, we found them in every smaller version of the Lenet 300-100 architecture that we experimented with, yet, with different ratios (See Table 2). We found that 84% of the iterative pruning rounds applied on Lenet 240-80 contain winning ticket. The rate drops to 17% for the Lenet 24-7. As shown in Figure 5, following the trend in Figure 4, larger networks contain more winning tickets. We can see that up to 75% smaller network sizes, the level of winning ticket percentage is above 40%. Also, there is no significant difference in the average performance of the sizes of the networks between 50%-25%.

We investigated the main properties of the winning tickets, such as the speed of learning and the test accuracies. In order to measure these metrics, we took the average values (minimum validation loss iteration and corresponding test accuracy) of the winning tickets generated for each size of the network (See Table 2). Figure 6 shows the learning speed behavior of the winning tickets. Higher iteration means slower learning (convergence). We can see that smaller network sizes reach minimum validation loss at higher iteration, indicating slower convergence. However, performance stays stable, and there is no radical change up to 75% smaller networks. After that point, winning tickets reach minimum validation loss in a much higher iteration.

Figure 4

15 rounds of iterative pruning process with Strategy 1 and Pruning Rate 0.2 applied on the MNIST dataset with smaller sizes of Lenet 300-100. Percentage of units remaining in hidden layers of Lenet 300-100 demonstrated on X-axis, super winning ticket percentage shown on the Y-axis.

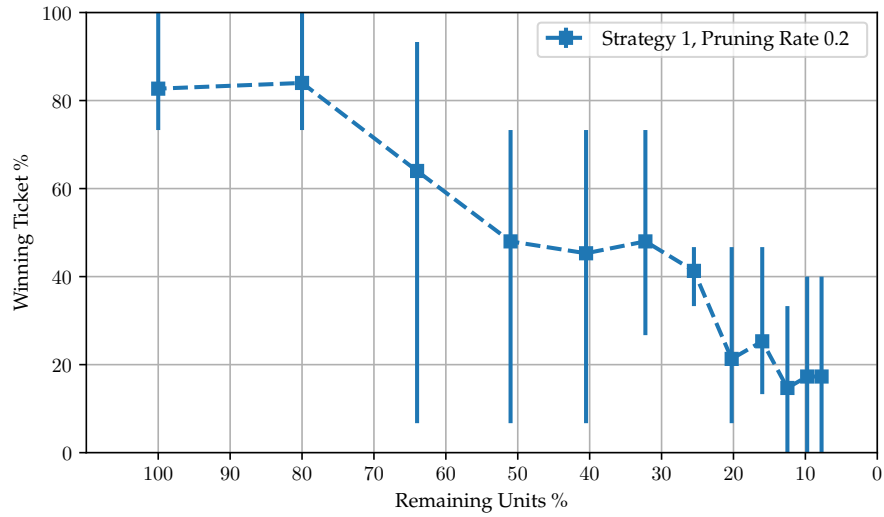
**Table 2**

15 rounds of iterative pruning process with Strategy 1 and Pruning Rate 0.2 applied on the MNIST dataset with smaller sizes of Lenet 300-100. Values shown are averaged over 5 runs. "First Layer" and "Second Layer" columns demonstrate the number of the units in the hidden layers. In the third column "Iteration" stands for the average iteration of the minimum validation loss reached in the winning tickets found on the corresponding network size. "Test Accuracy" in the fourth column stands for the average test accuracy of the winning tickets found on the corresponding network size. In the final column "WT" stands for winning tickets.

First Layer	Second Layer	Iteration	Test Accuracy	WT %
300	100	2493	97.60	83
240	80	2748	97.60	84
192	64	3067	97.50	64
153	51	3061	97.30	48
122	40	3699	97.20	45
97	32	3736	97.10	48
77	25	3799	96.90	41
61	20	4999	96.80	21
48	16	4724	96.40	25
38	12	5300	96.30	15
30	9	7103	95.70	17
24	7	7640	95.00	17

Figure 5

15 rounds of iterative pruning process with Strategy 1 and Pruning Rate 0.2 applied on the MNIST dataset with smaller sizes of Lenet 300-100. Percentage of units remaining in hidden layers of Lenet 300-100 demonstrated on X-axis, winning ticket percentage shown on the Y-axis.

**Figure 6**

15 rounds of iterative pruning process with Strategy 1 and Pruning Rate 0.2 applied on the MNIST dataset with smaller sizes of Lenet 300-100. Percentage of units remaining in hidden layers of Lenet 300-100 demonstrated on X-axis. *Average iteration* shown on the Y-axis is the average iteration of the minimum validation loss reached in the winning tickets found on the corresponding network size.

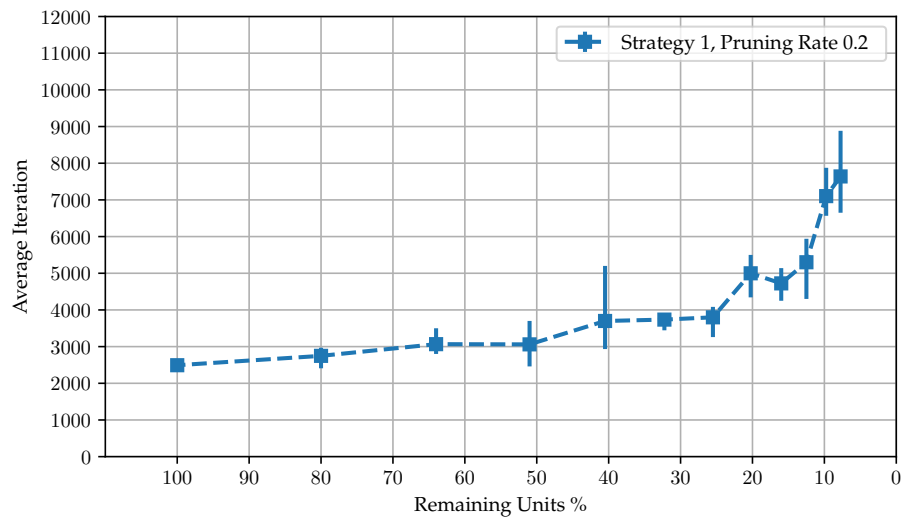


Figure 7

15 rounds of iterative pruning process with Strategy 1 and Pruning Rate 0.2 applied on the MNIST dataset with smaller sizes of Lenet 300-100. Percentage of units remaining in hidden layers of Lenet 300-100 demonstrated on X-axis. *Average test accuracy* shown on the Y-axis is the average test accuracy of the winning tickets found on the corresponding network size.

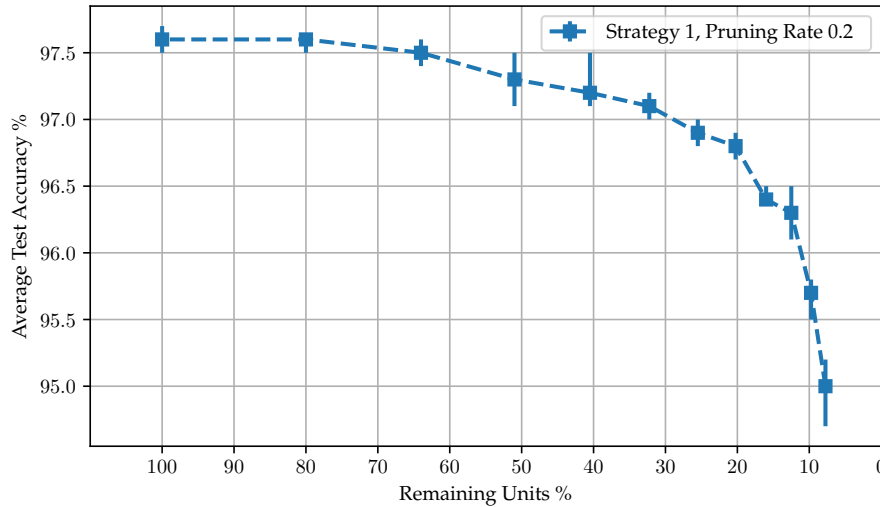


Table 2 provides the information for the test accuracies of the winning tickets. Up to 70% smaller network sizes, average test accuracies of the winning tickets stays above the 97%, which is the average performance of the Lenet 300-100 on the MNIST dataset (Lecun et al. 1998). After 70%, there is a significant performance loss, and we can see the general trend here as well (See Figure 7); smaller network sizes generate winning tickets with lower test accuracies.

6. Discussion

Frankle and Carbin (2018) came up with the lottery ticket hypothesis, which proposes that the networks contain much smaller sub-networks that can perform better than the original network. The authors suggested that we can find these winning tickets in significantly pruned networks with iterative magnitude pruning process. However, due to the exhaustive iterative pruning process, finding these winning tickets are computationally expensive.

Inspired by this exciting finding, to offer a solution to the computational efficiency problem; we wondered about the possibility of finding these winning tickets from the start with smaller network sizes. We introduced a new concept as a super winning ticket, a winning ticket generated from the smaller version of the original network that surpasses the performance of the original unpruned network. We aimed to answer our research question: to what extent is it possible to generate super winning tickets using the methodology of Frankle and Carbin (2018)?.

Frankle and Carbin (2018) suggested that the *over-parametrized neural networks* –which requires more computational power- have a higher probability of containing

the winning ticket. The logic behind is that larger DNNs have a larger number of sub-networks and hence a larger probability of having a useful sub-network for solving the task. However, empirical evidence (Zhou et al. 2019; Gaier and Ha 2019; Ramanujan et al. 2019) demonstrates to us that the most critical component of the performance of the winning tickets is the architecture of the neural network. For fully-connected neural networks, we can achieve the same architectures in the smaller networks. In addition, only important performance indicator about the initial weights are their signs (Zhou et al. 2019), so our probability remains the same independent of the network size. Therefore, we have speculated that over-parametrization should not be a necessity for the fully connected network architectures in order to find a useful sub-network, and using smaller neural networks from the start could give similar results.

In order to test our intuition, we experimented with the smaller versions of the Lenet 300-100 architecture on the MNIST dataset. After experimentation with different iterative pruning strategies and pruning rates, we found that up to 70% smaller versions of Lenet 300-100 architecture contain super winning tickets on the MNIST dataset with our optimal configuration: Strategy 1 and Pruning Rate 0.2¹¹.

Thus, according to our empirical study, it is possible to generate these super winning tickets with iterative pruning process from the start with smaller network sizes. As long as there is one, the number of super winning tickets is not that important. Thus, applying the iterative pruning process to the smaller networks from the start gives substantial computational performance advantage.

Based on our findings, we clearly see that the larger networks contain more (super) winning tickets. So, our study supports the theory that the over-parametrized neural networks have a higher probability of containing the winning ticket (Frankle and Carbin 2018). However, we can conclude that over-parametrization is not strictly required, because it is still possible to find useful sub-networks from much smaller networks. Our intuition about the necessity for over-parametrization in order to find winning tickets proved to be true to a certain extent.

This finding also emphasizes the potential overuse of the parameters in the DNN's, which causes computational (energy and time) inefficiency. Once again, the importance of understanding the DNN's is evident. Not only for efficiency purposes but also the potential broader use. Our study contributes towards to goal of the more understandable and efficient deep neural networks. With improved explainability, we might use deep learning more extensively in governmental services and official statistics in the future (European Commission 2020b).

6.1 Winning Ticket Properties

Frankle and Carbin (2018) also experimented with the different network sizes with Lenet 300-100 on the MNIST dataset. They reported that the winning tickets from the larger networks reach higher test accuracy while winning tickets from smaller networks reach minimum validation loss faster (in earlier iteration). We partially confirm their finding. According to our empirical study (Table 2), we also found that the larger networks reach higher test accuracies overall. However, we also found that larger networks tend to reach minimum validation loss faster than the smaller networks which differs from the findings of Frankle and Carbin (2018). But, it is important to mention that Frankle and Carbin (2018) experimented with 10 different versions of the Lenet 300-

¹¹ This is the same setup as the Frankle and Carbin (2018)

100 ranging from 900-300 to 25-9. Since our study focuses on smaller network sizes, we experimented with 11 different versions of the Lenet 300-100 ranging from 240-80 to 24-7. Hence, the network sizes used are not exactly the same.

As shown in Figure 5, larger networks also tend to contain more winning tickets on average. So, although over-parametrization is not a necessity to generate winning tickets, larger networks contain more and faster-learning winning tickets with higher test accuracies.

7. Conclusion

In this study, we have experimented with different strategies and methods in order to answer our research question: to what extent is it possible to generate super winning tickets using the methodology of [Frankle and Carbin \(2018\)](#)?. During the experimentation process, we also looked into different properties of winning tickets generated from smaller networks.

We found that up to 70% smaller versions of Lenet 300-100 architecture, it is possible to generate super winning tickets on the MNIST dataset. Our empirical study also partially confirms the findings of the [Frankle and Carbin \(2018\)](#): larger networks contain more and faster-learning winning tickets with higher test accuracy compared to the smaller networks.

[Han et al. \(2015\)](#) proposed the magnitude pruning methodology to remove the unnecessary weights from the networks up to 90% while not losing significant accuracy. [Frankle and Carbin \(2018\)](#) improved the pruning technique by introducing the iterative magnitude pruning process, and they found out that even significantly pruned sub-networks can reach similar or better performance as the original unpruned network when trained in isolation with their initial original weights. By using the same pruning technique as [Frankle and Carbin \(2018\)](#), we found that we can find sub-networks in the smaller network architectures that can surpass the performance of the original unpruned network.

7.1 Limitations and Future Work

We only experimented with fully-connected Lenet 300-100 architecture; thus, further experimentation with other types of networks (i.e., convolutional architectures) are required to observe the general behavior of super winning tickets in smaller network sizes. Also, we only used the MNIST dataset, which is a relatively small dataset compared to the real-life datasets.

We could only experiment with the limited number of configurations due to the extensive computational requirement of the iterative pruning process and time constraints. Different strategies or methods can give better results in the future. Although much faster with the smaller network sizes, the iterative pruning process is still computationally exhaustive. We used 5 rounds of experiments for every setup, but more rounds can be used in order to have more stable results.

We kept the ratio of the original network while removing the units in the hidden layers. However, it would be interesting to see how super winning tickets will behave in different ratios. Perhaps using a more severe removing percentage for larger hidden layers or randomly constructing smaller networks can give interesting results.

References

- Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.
- European Commission. 2020a. A european strategy for data. https://ec.europa.eu/info/sites/info/files/communication-european-strategy-data-19feb2020_en.pdf.
- European Commission. 2020b. White paper on artificial intelligence - a european approach to excellence and trust.
- Frankle, Jonathan and Michael Carbin. 2018. The lottery ticket hypothesis: Training pruned neural networks. *CoRR*, abs/1803.03635.
- Frankle, Jonathan, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. 2019. The lottery ticket hypothesis at scale. *CoRR*, abs/1903.01611.
- Gaier, Adam and David Ha. 2019. Weight agnostic neural networks. *CoRR*, abs/1906.04358.
- Glorot, Xavier and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, PMLR, Chia Laguna Resort, Sardinia, Italy.
- Han, Song, Huizi Mao, and William J Dally. 2015. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149*.
- Han, Song, Jeff Pool, John Tran, and William J. Dally. 2015. Learning both weights and connections for efficient neural networks. *CoRR*, abs/1506.02626.
- Hassibi, Babak and David G. Stork. 1993. Second order derivatives for network pruning: Optimal brain surgeon. In S. J. Hanson, J. D. Cowan, and C. L. Giles, editors, *Advances in Neural Information Processing Systems 5*. Morgan-Kaufmann, pages 164–171.
- He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *CoRR*, abs/1512.03385.
- Kingma, Diederik P. and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.
- Lecun, Yann, Leon Bottou, Y. Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324.
- LeCun, Yann and Corinna Cortes. 2010. MNIST handwritten digit database.
- LeCun, Yann, John S. Denker, and Sara A. Solla. 1990. Optimal brain damage. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*. Morgan-Kaufmann, pages 598–605.
- Leyen, Ursula von der. 2019. A union that strives for more - my agenda for europe. https://ec.europa.eu/info/sites/info/files/political-guidelines-next-commission_en_0.pdf.
- Li, Hao, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2016. Pruning filters for efficient convnets. *CoRR*, abs/1608.08710.
- Morcos, Ari S., Haonan Yu, Michela Paganini, and Yuandong Tian. 2019. One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers. *arXiv e-prints*, page arXiv:1906.02773.
- Narang, Sharan, Gregory F. Diamos, Shubho Sengupta, and Erich Elsen. 2017. Exploring sparsity in recurrent neural networks. *CoRR*, abs/1704.05119.
- Oymak, Samet and Mahdi Soltanolkotabi. 2020. Towards moderate overparameterization: global convergence guarantees for training shallow neural networks. *IEEE Journal on Selected Areas in Information Theory*.
- Ramanujan, Vivek, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. 2019. What’s Hidden in a Randomly Weighted Neural Network? *arXiv e-prints*, page arXiv:1911.13299.
- Yu, Haonan, Sergey Edunov, Yuandong Tian, and Ari S Morcos. 2019. Playing the lottery with rewards and multiple languages: lottery tickets in rl and nlp. *arXiv preprint arXiv:1906.02768*.
- Zhou, Hattie, Janice Lan, Rosanne Liu, and Jason Yosinski. 2019. Deconstructing lottery tickets: Zeros, signs, and the supermask. *CoRR*, abs/1905.01067.