REPRODUCIBILITY REPORT: THE LOTTERY TICKET HYPOTHESIS: FINDING SPARSE, TRAINABLE NEURAL NETWORKS

Fengxia Shu & Xuan Qi & Liang Liang
{fs1a19,xq3g19,ll3a19}@soton.ac.uk

ABSTRACT

The feature of lottery tickets is that a stack of tickets in which only a handful of tickets actually win. This article uses the lottery ticket to metaphor of the neural network structure. Subnetworks(winning tickets), which have a major role to play in the general prediction results, may have only a small part, and the rest (no winning tickets) can be removed in order to accelerate model training and decrease complexity.

1 INTRODUCTION

Generally speaking, a large amount of data sets and expensive calculation costs are required for initial training of a Neural Network. As a result, a huge neural network structure filled with complex connections between hidden layers is obtained. This structure often requires optimization techniques to remove certain connections to adjust the model size.

A question that has puzzled researchers for decades is whether we really need such a huge neural network structure. Obviously, if we connect each neuron within the network, a particular problem can be solved, but due to the high cost we may have to stop. Can we not start a smaller, more streamlined network training? This is the essence of the lottery assumption. Based on the above, the author proposes a lottery hypothesis theory [1]. The formal definition of the lottery hypothesis in the article is: a randomly initialized dense neural network contains an initialized subnetwork. When trained separately, it can pass the same number of iterations at most and can achieve the same test accuracy as the original network. All parameters of a complex network are considered to be a prize pool. The above parameters in the subset are lottery tickets.

This paper experimented on the author's lottery hypothesis: the huge neural network structure that was obtained through the training process is a big bag of lottery tickets for machine learning models. The model must be optimized following initial training, like pruning, removing unnecessary weights in the network, thus reducing the model's size without sacrificing its performance. This ensures that the winning lottery ticket will be found in the bag and all the remaining lottery tickets are discarded. Normally, the network structure after pruning is about 90 percent smaller than the original one.

The author also found that this trainable subnet from fully connected and convolutional feedforward network is automatically discovered by traditional pruning techniques. However, they cannot be effectively trained until the weights of the subnets are initialized. The structure and weight characteristics of the other networks are also interesting in addition to these striking results. Reinitialization with new weights will result in poor training. In addition, It is also pointed out in the paper that in convolutional neural networks at small learning rates, the original parameters are better than random initialization. However, when at large learning rates, the original parameters and random initialization are not significantly different.

Our report is divided into 5 parts, the second part is an brief introduction to the experimental method, the third step is to focus on describing the experimental details and analysis of the results, the fourth step is discussed in future can continue to study direction. Finally, we summarized this report.

2 EXPERIMENTAL METHODOLOGY

In this article, the author proposes a simple method for generating a sparse high-performance network: after training the network, set all weights less than a certain threshold to 0, and the rest reset back to its initial configuration, and then retrain the network from this initial configuration with the pruned weights in a frozen state. The author proposes a method to find the winning lottery by iteration.
1. Create an initial neural network $f(x;\theta_0)$ randomly, while the parameters $\theta_0 \sim D_\theta$.
2. Train the network by iterating j times, then the parameter θ_j could be obtained.
3. Prune p$\%$ of the θ_j parameters, then there would be a mask m.
4. To process the parameters remained, set their values to 0, so the winning ticket $f(x;\theta_0)$ could be obtained.
Here is another pruning method,random reinitialization pruning, where step 1,2,3 are same as before and then process the remaining parameters by assigning them a random value $\theta_0 \sim D_\theta$, so the winning ticket $f(x; m \odot \theta_0^{'})$ could be obtained.

3 EXPERIMENT DETAILS AND RESULT ANALYSIS

3.1 THE EXPERIMENT OF OBTAINING LOTTERY SUBNETWORK ON MNIST

In this step, we pruned using the winning ticket method in fully-connected networks (fc1) and convolutional networks (Lenet5) on MNIST. It provides detailed information about the configuration of networks architecture. As shown in Figure 1, we totally run 35 times in both two networks. Each time 10% of FC layers are pruned in fully-connected networks and 10% of FC layers and 10% of Conventional layers are cut off in convolutional networks.

Network	fc1	Lenet5
Convolutions layers	-	64，64，pool
FC Layers	300，100，10	256,256,10
All/Conv Weights	266K	4.3M / 38K
Mini-batch size	60	60
Iterations	35	35
Prune percent for FC layers	10	10
Prune percent for Conv layers	-	10
Learning rate	1.2e-3	1.2e-3
Optimizer	Adam	Adam
Dataset for evaluation	MNIST	MNIST

Figure 1: Network configuration for fc1 and Lenet5 model

3.1.1 THE EXPERIMENT OF OBTAINING LOTTERY SUBNETWORK IN FC1 ON MNIST

First of all, We plot the remaining weight as the X-axis and the corresponding test accuracy as the Y-axis,as shown Figure 2. We can observe that such lottery subnetwork does exist. It is clearly shown that there is no substantial drop in model performance relative to the initial model after a large number in pruning (removing 85% - 90% of weights).

3.1.2 THE EXPERIMENT OF OBTAINING LOTTERY SUBNETWORK IN LENET5 ON MNIST

We can also see the same trend in Figure 3. After a large number of pruning (removing 85% - 90% of the weights), there is no significant decrease in model performance compared with the original model, that is, the 100% of the weights. From this experiment can also prove that the lottery hypothesis is true.

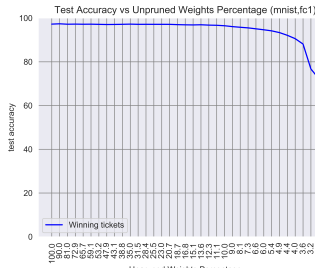

Figure 2: Test Accuracy vs Weights% (MNIST,fc1)

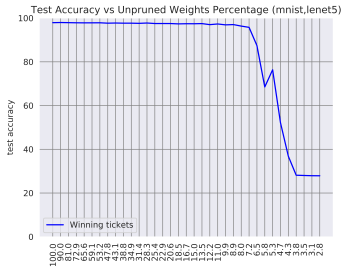

Figure 3: Test Accuracy vs Weights %(MNIST,Lenet5)

3.2 THE COMPARISON OF TWO DIFFERENT WEIGHT INITIALIZATIONS FOR PRUNING ON MNIST

In this step, we compare the trainability of winning tickets' initialization and random initialization in convolutional networks (Lenet5) and fully-connected networks (fc1) on MNIST. In the method of winning tickets' initialization, we retrain the remaining weights according to the value they are first initialized. The random initialization method is to randomly initialize the remaining weights and then train them.

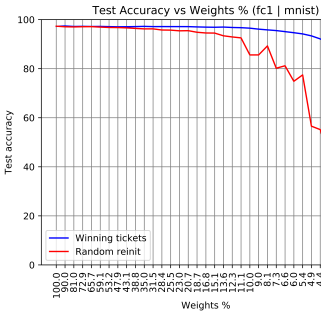

Figure 4: fc1, lr:$1.2e-3$

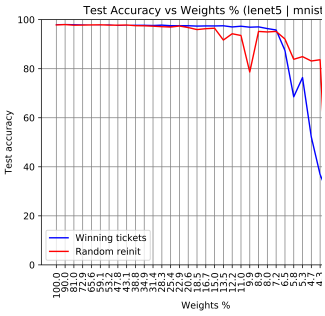

Figure 5: Lenet5, lr:$1.2e-3$

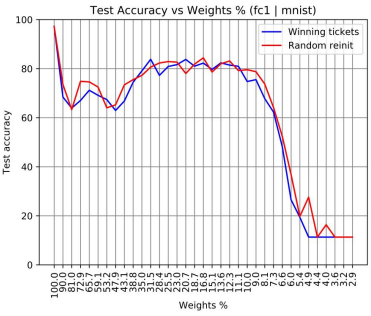

Figure 6: fc1, lr:0.1

3.2.1 THE COMPARISON OF TWO DIFFERENT WEIGHT INITIALIZATIONS FOR PRUNING IN FC1 ON MNIST

Firstly, we do experiment in fully-connected networks (fc1) on MNIST. Details of the experiment are shown in figure and the steps were described in last subsection. We also totally run 35 times and each time 10% of FC layers are cut off. The result shown in Figure 4, we can see that after pruning 10 times, namely the remaining weights is 38.8% of the total weight, the performance of random initialization is worse than winning tickets' initialization since then. Besides, when the weight is only 10%, the performance of random initialization is about 10% lower than the winning tickets' initialization.

3.2.2 THE COMPARISON OF TWO DIFFERENT WEIGHT INITIALIZATIONS FOR PRUNING IN LENET5 ON MNIST

Secondly, we do experiment in convolutional networks (Lenet5) on MNIST.The Experimental results are shown in Figure 5. In general, the performance of random initialization is inferior to winning tickets' initialization in this data and network structure. We can see that after pruning 10 times, namely, the remaining weights 38.8% of the total weight, the performance random initialization is worse than winning tickets' initialization since then. Besides, when the weight is only 10%, We can clearly see that the accuracy of winning tickets' initialization may be 20% higher than that of

random initialization. It can also be seen from the figure that random initialization fluctuates greatly in this network structure.

3.3 RESULTS OF WINNING TICKETS' INITIALIZATION AND RANDOM INITIALIZATION PRUNING AT LARGE LEARNING RATE IN FC1 NETWORK ON MNIST

In this step, we mainly explore whether the effect of high learning rate and low learning rate corresponding to two different ways of initialization after pruning is different in fully-connected networks. We found that the author mentioned in the paper if the learning rate is too high, there is little difference between the two pruning methods for convolutional network. However, the author did not mention about the result of the results in fully-connected layer. So, we do the related experiment to discuss learning rate's influence on the fully-connected layer structure.

In the section 3.2, we already have obtained the results about two different ways of pruning initialization in fc1 on MNIST. It is clearly shown that the learning rate is 1.2e-3 in the left column of Table 1. Now we change the learning rate to 0.1 with other parameters unchanged under this structure to train it. We can obtain the results in Figure 6. When the learning rate is large, the difference in performance between two methods of pruning is not big in the fully connected network structure. Besides, we can find the effect of the two methods of pruning is not so good. This experiment also can prove that winning ticket strategy based on pruning is sensitive to learning rate. Iteration pruning will not find the winning ticket when the learning rate is large, and the performance is no better than random initialization pruning.

4 DISCUSSION

As the author mentioned in the paper, this experiment iterative calculation of pruning is too big, which needs 15 consecutive training or more than 15 times training on a network. Our experiments are 35 times of iteration, which makes the whole training process taking a long time. Each model need to run more than 10 hours. In the future, we can explore more effective method of looking for winning tickets.

5 CONCLUSION

In this report, we mainly reproduce three views of authors. Firstly, it is the lottery ticket assumption that a sub-network exists in the neural network that can achieve the similar test accuracy within a similar number of iterations as the original network. We find a subset which is 10% weight of the original has the similar performance in fully-connected layer, as well as a convolution network. The second reproducing point is the importance of validating initialization. As we mentioned in the previous experiment, winning ticket‘ initialization performs better than random initialization in both the fully-connected layer and the convolutional neural network, that is, it is better to keep the remaining parameters as they are rather than to reinitialize them. The point of view of the third reproduction experiment is that the winning ticket is sensitive to the learning rate. Its shows that when the learning rate is quite large, the winning ticket sub-network will not be found.

ACKNOWLEDGMENTS

Github repository https://github.com/rahulvigneswaran/Lottery-Ticket-Hypothesis-in-Pytorch greatly helped our work. Their codes are useful to reproduce the paper.

REFERENCES

[1] Frankle, Jonathan, and Michael Carbin. "The lottery ticket hypothesis: Finding sparse, trainable neural networks." arXiv preprint arXiv:1803.03635 (2018). DOI:https://arxiv.org/abs/1803.03635