Testing the Lottery Ticket Hypothesis

Artur Back de Luca backdeluca.1900870@studenti.uniroma1.it

Abstract

This work investigates the phenomenon first brought forward by Frankle and Carbin (2018), known as the *Lottery Ticket Hypothesis*. Furthermore, details of the implementation[1] replicating the original findings, and some of the experiments are also disclosed.

1. Motivation

The archetype of neural network that became popular in the last decades is known to be deep. The ability to train an extensive stack of computational layers enabled researchers to tackle more challenging problems and to reach unprecedented performance. Although these models usually process considerably complex tasks, their overall number of parameters far exceeds what could be in fact necessary. Nevertheless, despite potential redundancies, deep networks are usually employed for the lack of more effective solutions.

However, in spite of their undisputed performance, training such bulky models comes at a price: with far too many coexisting parameters to oversee, the overall training time for these models can be burdensome, lasting insofar as days or even weeks. The drawbacks are not only practical, but also environmental: Strubell et al. (2019) shows that training deep learning models can produce as much carbon as nearly what five average cars generate over their whole life span.

Remedying these problems naturally demands for more efficient solutions that share same levels of accuracy. However, this requires understanding the underlying mechanisms when training these complex models, much of which remains unexplained. To this effect, identifying properties of deep networks is crucial, something that could provide evidence to help explain deep networks from a theoretical perspective. One of the most recent findings gets the name of the *Lottery Ticket Hypothesis* and sheds light onto overlooked aspects during training.

2. Pruning and the Lottery Ticket Hypothesis

Several research streams dwelt on finding ways to reduce the complexity of deep learning models without hurting performance. One of these solutions comes in the way of pruning. As our general notion suggests, pruning essentially removes redundant connections within the network based on given criteria, much like a gardener would prune the excessive leaves of a tree.

1. Code available at: https://github.com/artur-deluca/lth

Typically, pruning is carried out after several steps of training or even after the network is fully trained. However, having an excessive number of parameters prompts the question: couldn't smaller networks achieve similar performance in the first place?

As Li et al. (2016) suggests, smaller models have less capacity for fitting data, which can be detrimental when it comes to training. Largely over parametrized models, on the other hand, have many degrees of freedom, and thus are able to fit the data in many ways, since there are several sub-network combinations inside its structure that could capture the structure of the task in hand. This idea is supported by experiments both in Li et al. (2016) and in Frankle and Carbin (2018) where networks once pruned and trained from scratch would generally yield results of no match to their original structure.

However, the experiments brought forward by Frankle and Carbin (2018), Frankle et al. (2019), and subsequent studies shed more light on the training dynamics between these types of models. It turns out that some pruned networks can indeed yield as good or even better results and even faster as their original formulations as long as they are initialized with the exact same parameters as the original network.

These pruned sub-networks found inside larger models were given the name of *winning tickets*, that is, their initialization has proven to be particularly effective for the training procedure ahead of them. So far, these results have been more noticeable for dense-shallow networks, but they also extend to more complex architectures such as VGGs and ResNet. This empirical observation motivates the proposition that all such networks have these winning tickets, which became famously known as the *Lottery Ticket Hypothesis*.

3. Research objective

This project intends to revisit this hypothesis by replicating the framework as described in Frankle and Carbin (2018) and also partially in Frankle et al. (2020)[2]. However, due to the unavailability to such computational resources as the ones enjoyed by the authors, the settings will be limited to the simplest of the architectures, excluding experiments with more complex architectures such as VGGs and ResNets, resorting to the single fully-connected architecture in the work using the MNIST dataset.

4. Implementation details

In this section, we go through technical details involving the implementation of the experimental set-up, as well as the pruning techniques employed.

Programming language and versions

The whole framework as built using Python 3.7.2, compatible with all versions equal to or greater than 3.7. All the models, training procedures, and pruning were carried out using Pytorch's 1.6.0 version, as well as torchvision 0.7.0.

2. The authors have released the framework used to execute their experiments. It is maintained by Facebook Research at: github.com/facebookresearch/open_lth

Iterative magnitude pruning

Pruning is commonly thought of as a single process. However, to achieve great levels of sparsity within a network, a single process can be troublesome. Alternatively, in Frankle and Carbin (2018), the authors propose what is called *iterative pruning*, a pruning step that progressively achieves the desired level of sparsity. Initially, once the training procedure is finished, the network undergoes a pruning step, which removes a percentage $p\%$ of connections based on some criteria. As the authors employ, pruning removes the smallest $p\%$ of connections based on the L1-norm. Pruning can be employed globally, or layer-wise, varying the pruning rate according to the layer type and position.

Once pruning is completed, the network gets the remaining weights reset to their original initialization values. Then the whole process is repeated until the network reaches the desired level of sparsity.

As observed in Frankle and Carbin (2018) and later mitigated in Frankle et al. (2020), iterative pruning does not achieve the desired effects in more parametrized models, such as VGGs and ResNets. In order to remedy this, the authors devised an extension of the iterative pruning process called iterative magnitude pruning (IMP) with rewinding, resetting the network not to their initial weights but to a solution some iterations thereafter. For instance, instead of rewinding to the very beginning, the network is re-winded to the k^{th} iteration. This addition has shown to stabilize the lottery tickets and yield the desired effects. Since IMP can also replicate the original formulation as long as k is set to 0, this will be the implemented version.

Experiments

In this section, we discuss the experiments carried out in Frankle and Carbin (2018) and the exact extent to which they were replicated. In the following table we see all the models tested.

Table 1: Models analyzed in Frankle and Carbin (2018)

Model	Lenet	Conv-2	Conv-4	Conv-6	ResNet18	VGG-19
Convolutions		64, 64, pool	64, 64, pool 128, 128, pool	64, 64, pool 128, 128, pool 256, 256, pool	16, 3×[16, 16] 3×[32, 32] 3×[64, 64]	2×64, *pool* 2×128, *pool* 4×256, *pool* 4×512, *pool* 4×512
FC Layers	300,100,10	256,256,10	256,256,10	256,256,10	avg-pool,10	avg-pool,10
All weights	266K	4.3M	2.4M	1.7M	274K	20.0M
Iterations/Batch	50K/60	20K/60	25K/60	30K/60	30K/128	112K/64
Optimizer	Adam 1.2e-3	Adam 2e-4	Adam 3e-4	Adam 3e-4	← 0.1-0.01-0.001 Momentum 0.9 →	
Pruning rate	fc20%	conv10%fc20%	conv10%fc20%	conv15%fc20%	conv20%fc0%	conv20%fc0%

The first four models are pruned layer-wise while the remaining is prune *globally*, that is, removing the given pruning percentage of the structure altogether. Additionally, all the

last layers were pruned to half of the given pruning rate. Additionally, all models but Lenet were trained under CIFAR10 while the latter under MNIST.

Due to the aforementioned limitations, the experiments will be restrained to the first setting, wherein contrast with the original implementation, the validation and the test set will be evaluated at each epoch rather than at each training iteration.

5. Results

Considering the aforementioned settings, we come to the following results.

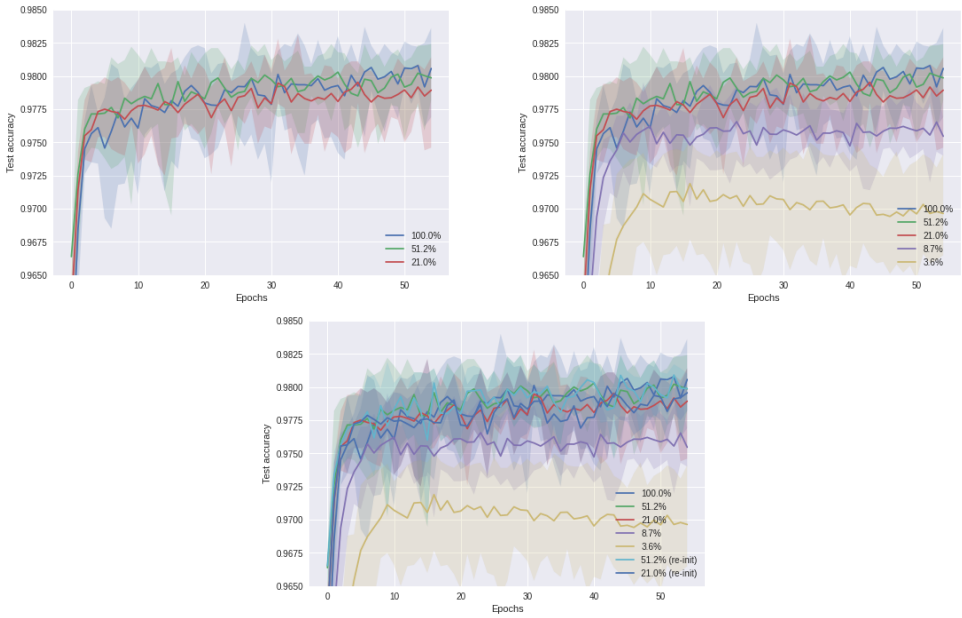

Figure 1: Test accuracy on Lenet as training proceeds. Each curve is the average of five trials with envelopes surrounding the maximum and the minimum

Surprisingly, contradicting the expected behavior describe by previous works, we observe no sign of improvement upon subsequent pruning nor any discrepancies w.r.t. to the original and random initialization, which is also further supported by observing the test accuracy as pruning increases.

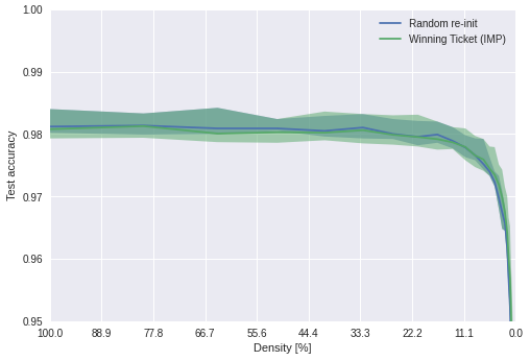

Figure 2: Test accuracy along pruning. Each curve is the average of five trials with envelopes surrounding the maximum and the minimum

Alternatively, as the authors propose, a proxy for assessing performance is investigating the training dynamics in early stopping. Since one of the claims is that lottery tickets achieve better results faster, a proxy for assessing performance improvement is analyzing where in training we obtain the model that best performs in the validation set, what can be virtually seen as a perfect early stopping.

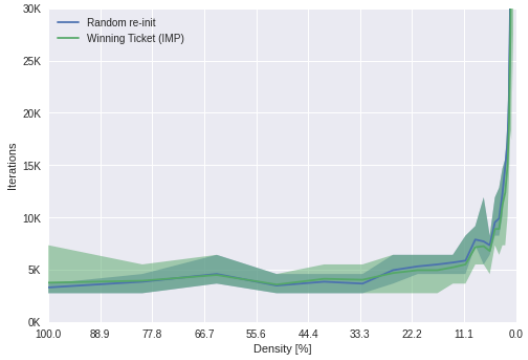

Figure 3: Early stop iteration along pruning. Each curve is the average of five trials with envelopes surrounding the maximum and the minimum

The noticeable peaks in the prior graphs are explained by complete degradation of the original structure of the network, as sparsity reaches near 100% and test accuracy plummets while the best iteration shoots up.

6. Discussion

Considering the aforementioned results, and the amount of peer-reviewed evidence available, it is likely that these results are a product of some unforeseen mistake, despite several re-runs of analysis and corrections. Even more so when such results manifests in one of the

settings where such phenomenon becomes most prominent. As further work, the author proposes to revise the code in search for any potential mistakes, as well as to test the framework in additional variations, such as changing pruning criteria and other fully connected model given said limitations. Despite the adversities, the project has been a great exercise of autonomy, technical craftsmanship and critical thinking when analyzing disagreements between theory and practice.

References

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks, 2018. URL https://openreview.net/forum?id=rJl-b3RcF7.

Jonathan Frankle, David J. Schwab, and Ari S. Morcos. The early phase of neural network training. 2019. URL https://openreview.net/forum?id=Hkl1iRNFwS.

Jonathan Frankle, Gintare Karolina Dziugaite, Daniel M. Roy, and Michael Carbin. Stabilizing the lottery ticket hypothesis. 2020. URL http://arxiv.org/abs/1903.01611.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets, 2016.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp, 2019.