The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks

ICLR 2019

Jonathan Frankle    Michael Carbin

6 December 2019

Presented by: Kevin Liang
Neural networks perform well for many machine learning tasks, but can have a lot of parameters.

Can we reduce model size while maintaining model accuracy?

Benefits:
- Faster
- Less memory
- Lower power consumption
Network Pruning

- Neural network pruning: Reduce parameters (sometimes by 90+%) without sacrificing (much) accuracy
- Old idea: e.g. Optimal Brain Damage (LeCun et al. 1990)
- Many types:
  - Unstructured (by weight) vs structured (by neuron/channel)
  - Pre-defined vs automatic
  - By loss vs by activation vs by magnitude vs ...
Network Pruning Stages

- Training
- Pruning
- Fine-tuning

Was commonly held that all three of these stages are essential.

- Contemporary works suggest that pruned networks trained from scratch do not do as well.
- It was thought that fine-tuning the pruned network is necessary to recover “lost” performance, even when keeping trained weights.
The Lottery Ticket Hypothesis

A randomly-initialized, dense neural network contains a subnetwork 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.
More Formal Statement

For a dense feed-forward neural network $f(x; \theta)$:
- Initialize with parameters $\theta = \theta_0 \sim \mathcal{D}_\theta$
- Train for $j$ iterations of SGD
- Achieve accuracy $a$

For the same neural network, but masked with $m \in \{0, 1\}^{||\theta||}$, resulting in $f(x; m \odot \theta)$:
- Initialize with *same* parameters $\theta_0$
- Train for $j'$ iterations of SGD
- Achieve accuracy $a'$

The lottery ticket hypothesis predicts that $\exists m$ for which $j' \leq j$, $a' \geq a$, and $||m||_0 \ll ||\theta||$. 
Winning Tickets

- Standard pruning techniques can find \( m \), *i.e.* the trainable subnetworks \( f(x; m \odot \theta_0) \), for both fully-connected and convolutional networks.
- The authors find that using a different random initialization \( \theta'_0 \) instead of \( \theta_0 \) in conjunction with \( m \) does *not* recover the same performance.
- These combinations of connections (\( m \)) and weight initializations (\( \theta_0 \)) are denoted *winning tickets*.
Finding Winning Tickets

One-shot pruning\(^1\):

1. Randomly initialize a neural network \( f(x; \theta_0) \) (where \( \theta_0 \sim D_\theta \)).
2. Train the network for \( j \) iterations, arriving at parameters \( \theta_j \).
3. Prune \( p\% \) of the parameters in \( \theta_j \), creating a mask \( m \).
4. Reset the remaining parameters to their values in \( \theta_0 \), creating the winning ticket \( f(x; m \odot \theta_0) \).

\(^1\)Note: The authors find iterative pruning to be more effective, which repeats the above steps for \( n \) rounds, each time pruning \( p \frac{1}{n} % \) of the weights.
# Architectures/Settings Tested

<table>
<thead>
<tr>
<th>Network</th>
<th>Lenet</th>
<th>Conv-2</th>
<th>Conv-4</th>
<th>Conv-6</th>
<th>Resnet-18</th>
<th>VGG-19</th>
</tr>
</thead>
<tbody>
<tr>
<td>Convolutions</td>
<td></td>
<td>64, 64, pool</td>
<td>64, 64, pool</td>
<td>64, 64, pool</td>
<td>16, 3x[16, 16]</td>
<td>2x64 pool 2x128 pool, 4x256, pool</td>
</tr>
<tr>
<td>FC Layers</td>
<td></td>
<td>256, 256, 10</td>
<td>256, 256, 10</td>
<td>256, 256, 10</td>
<td>avg-pool, 10</td>
<td>avg-pool, 10</td>
</tr>
<tr>
<td>All/Conv Weights</td>
<td></td>
<td>266K</td>
<td>4.3M / 38K</td>
<td>2.4M / 260K</td>
<td>1.7M / 1.1M</td>
<td>274K / 270K</td>
</tr>
<tr>
<td>Iterations/Batch</td>
<td></td>
<td>50K / 60</td>
<td>20K / 60</td>
<td>25K / 60</td>
<td>30K / 60</td>
<td>30K / 128</td>
</tr>
<tr>
<td>Optimizer</td>
<td></td>
<td>Adam 1.2e-3</td>
<td>Adam 2e-4</td>
<td>Adam 3e-4</td>
<td>Adam 3e-4</td>
<td>← SGD 0.1-0.01-0.001 Momentum 0.9 →</td>
</tr>
<tr>
<td>Pruning Rate</td>
<td></td>
<td>fc20%</td>
<td>conv10% fc20%</td>
<td>conv10% fc20%</td>
<td>conv15% fc20%</td>
<td>conv20% fc0%</td>
</tr>
</tbody>
</table>

Define $P_m = \frac{||m||_0}{||\theta||}$ as the sparsity of the mask. $P_m = 25\%$ means 75\% of the weights have been pruned.
Fully-Connected Networks for MNIST

Lenet (300-100-10)

(a) Early-stopping iteration and accuracy for all pruning methods.

(b) Accuracy at end of training.

(c) Early-stopping iteration and accuracy for one-shot pruning.
Analysis highlights

- **Training speed (measured by early stopping on validation):**
  - Winning tickets learn faster than the full network as $P_m$ decreases from 100% to 21%.
  - Randomly initialized pruned networks monotonically learn slower as $P_m$ decreases.
  - Winning tickets consistently learn faster than randomly initialized counterparts.

- **Test accuracy:**
  - Winning tickets achieve better test accuracy than the full network, up to a maximum 0.3% improvement when $P_m = 13.5%$.
  - Randomly initialized pruned networks monotonically achieve lower test accuracies as $P_m$ decreases.
  - Winning tickets consistently achieve better test accuracy than randomly initialized counterparts.

- **Iterative vs One-shot pruning:**
  - For random initialization, iterative and one-shot pruning perform similarly.
  - While more expensive, iterative pruning finds smaller and better performing winning tickets than one-shot. Both types of pruning do better than random initializations.
Convolutional Networks for CIFAR10

Conv-2, Conv-4, Conv-6

Similar pattern to results as Lenet on MNIST.
Dropout is a popular technique for neural networks, which randomly disables a proportion of the neurons; training with dropout has been characterized as training the ensemble of subnetworks.

Lottery ticket suggests that one or more of these subnetworks may be winning tickets.

- How does dropout affect winning tickets?
Dropout Results

Dropout rate of 50%

Dropout slows down training (as expected), but overall, many of the patterns are preserved; dropout and winning tickets seem to have synergy.

- A perhaps intuitive result, as dropout reduces neuron interaction representations, which may lead to easier pruning.
VGG and ResNet for CIFAR10

Figure 7: Test accuracy (at 30K, 60K, and 112K iterations) of VGG-19 when iteratively pruned.

Figure 8: Test accuracy (at 10K, 20K, and 30K iterations) of Resnet-18 when iteratively pruned.
Discussion

The importance of winning ticket initialization:
- Winning tickets learn slower and achieve lower test accuracies when randomly reinitialized, so the initial values are important.
- A possible explanation is these initial weights are close to their final trained values, but actually, winning ticket weights move further than other weights. Instead, they may be due to starting in a more favorable region of the loss landscape.

The importance of winning ticket structure:
- Perhaps the winning ticket architecture encodes an inductive bias for the learning task.

The improved generalization of winning tickets:
- Test accuracy forms an Occam’s Hill: original network may have too many parameters and too much complexity, while extremely pruned models have not enough.

Implications for neural network optimization:
- Is a winning ticket necessary or sufficient for SGD to optimize a neural network?
- Perhaps overparameterized networks are easier to train because they have more “lottery tickets” that can potentially be winning tickets.
Selected Other Works

  - Scales up the Lottery Ticket to ResNet50 on ImageNet.

  - Magnitude increase also an effective mask criteria, signs of the initializations more important than exact value, zeroing pruned weights is important, existence of “Supermasks.”

  - Lottery tickets occur in NLP and RL as well.

- **Z. Liu, M. Sun, T. Zhou, G. Huang, T. Darrell. Rethinking the Value of Network Pruning. ICLR 2019.**
  - Concurrent work with somewhat contradictory result implying that pruned networks can learn from random in some settings.