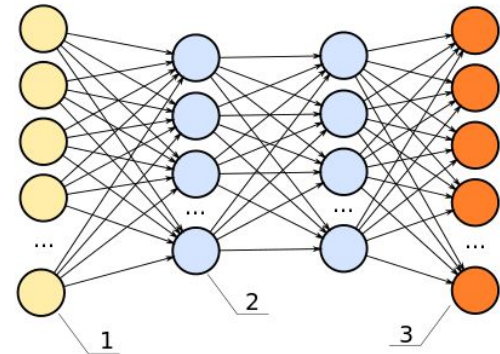


The Lottery Ticket Hypothesis

paper by Jonathan Frankle and Michael Carbin
@ MIT CSAIL [\[link\]](#)

presentation by Jack Morris
12/1/19



Background: **Pruning**

- To reduce the size of a neural network by removing unwanted parts
- People have been trying to **prune** neural networks for awhile
 - Idea originated into 1990s

The Pruning Process

1. Train the network
2. Remove **superfluous structure**
3. Fine-tune the network
4. [optional] iteratively repeat steps 2 and 3

What structure?

Weights? Neurons? Filters? Channels?
Layers?

What does “superfluous” mean?

Magnitudes? Gradients? Activations?

Motivation

As you may imagine, lots of groups have tried something like this before

“Training a pruned model from scratch performs worse than retraining a pruned model, ..., which may be due to the difficulty of training small networks from scratch” – Pruning Filters for Efficient ConvNets



Motivating Questions

Can we train sparsely pruned networks from scratch? **Yes**

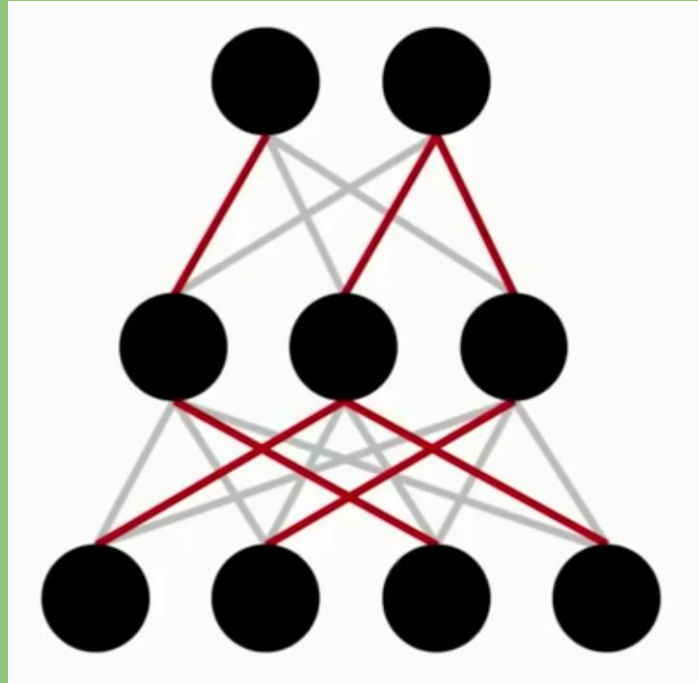
Corollary: Do networks have to be overparameterized to learn? **No**

There's a catch: **Need to reuse the weight initializations from the original training process.**

Training Pruned Networks

1. Randomly initialize the network's weights
2. Train it and prune superfluous structure
3. Reset each remaining weight to its value from 1
4. Repeat and  **PROFIT** 

Training Pruned Networks



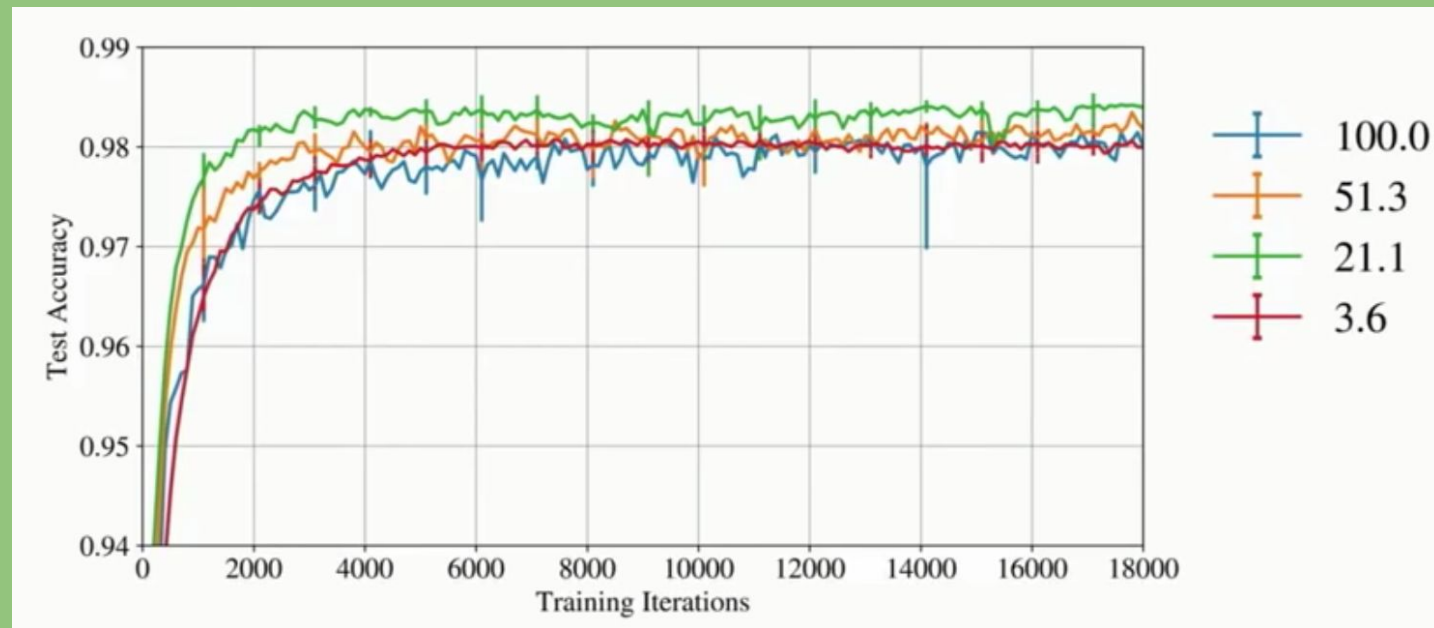
Training Pruned Networks

- Giant appendix shows that this works with batch normalization, dropout, convolutional layers, weight decay, residual connections, optimizers, etc. for any hyperparameter choices
- ***Caveats:***
 - 1. If you randomly reinitialize the network, this won't work
 - 2. You still have to train the network first (so it's not a particularly efficient process)

Training Pruned Networks

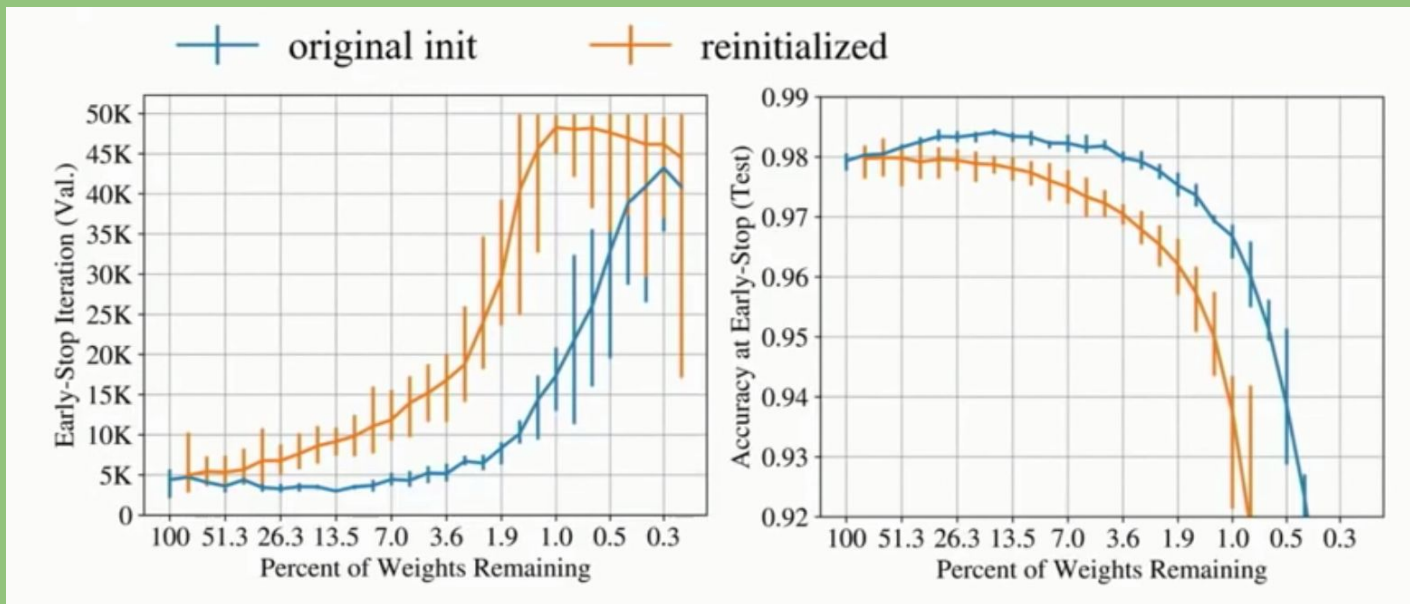
- So to recap, these small subnetworks:
 - 1. Are between 15% and 1% of the original size
 - 2. Learn faster than the original network
 - 3. Reach the same or higher test accuracy

Results



LeNet 300-100-10 for MNIST / fully-connected / 300k parameters

Results



LeNet 300-100-10 for MNIST / fully-connected / 300k parameters

The Lottery Ticket Hypothesis

Plain English:

Dense, trainable networks have sparse trainable subnetworks that are equally capable

Formally:

$f(x; W)$ reaches accuracy \mathbf{a} in \mathbf{t} iterations

$f(x; m \circ W)$ reaches accuracy \mathbf{a}' in \mathbf{t}' iterations

$$\exists m \mid \sum m \ll w \quad \mathbf{a}' \geq \mathbf{a} \quad \mathbf{t}' \leq \mathbf{t}$$

Possible future work

- Finding a way to prune networks early in training
- Examining these subnetworks to see what works– use this info to develop better architectures and initializations
- Make good subnetworks and reuse them on tasks
 - (A good test for overfitting, too)
- *Stabilizing the Lottery Ticket Hypothesis* (Frankle, 2019)
 - Prune after a few iterations, not at $t=0$
 - Compress ResNet-50, Inception-v3 in one shot by over 50%!

Questions?