The Lottery Ticket Hypothesis

Coco Zhang and Noah Andrew

May 8th, 2024

The Theorem (Frankle & Carbin, 2019)

"Dense, randomly-initialized, feed-forward networks contain subnetworks (winning tickets) 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."

Neural network pruning

- **Objective**
  - Improve performance & accuracy and decrease storage requirements by reducing the parameter counts of trained networks
- **One-shot Pruning**

  1. Randomly initialize a neural network $f(x; \theta_0)$ (where $\theta_0 \sim \mathcal{D}_\theta$).
  2. Train the network for j iterations, arriving at parameters θ_j.
  3. Prune $p\%$ of the parameters in θ_j, creating a mask m.
  4. Reset the remaining parameters to their values in θ_0, creating the winning ticket $f(x; m \odot \theta_0)$.

- **Iterative Pruning**
  - Repeats the process of one-shot pruning (prune, reset, retrain) over n rounds, where
  - Each round prunes $p^{1/n}$ % of weights that survived the previous round
  - More computationally and time intensive, since the network must be pruned and re-trained many times

Experiments (Frankle & Carbin, 2019)

- Establishing the lottery ticket hypothesis for various neural network architectures.
- Findings:
    - **Iterative pruning**: extracts smaller winning tickets.
    - **One-shot pruning**: identifies winning tickets without repeated training.

Network	Lenet	Conv-2	Conv-4	Conv-6	Resnet-18	VGG-19
Convolutions		64, 64, pool	64, 64, pool 128, 128, pool	64, 64, pool 128, 128, pool 256, 256, pool	16, 3x[16, 16] 3x[32, 32] 3x[64, 64]	2x64 pool 2x128 pool, 4x256, pool 4x512, pool, 4x512
FC Layers	300, 100, 10	256, 256, 10	256, 256, 10	256, 256, 10	avg-pool, 10	avg-pool, 10
All/Conv Weights	266K	4.3M / 38K	2.4M / 260K	1.7M / 1.1M	274K / 270K	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	← SGD 0.1-0.01-0.001 Momentum 0.9 →	
Pruning Rate	fc20%	conv10% fc20%	conv10% fc20%	conv15% fc20%	conv20% fc0%	conv20% fc0%

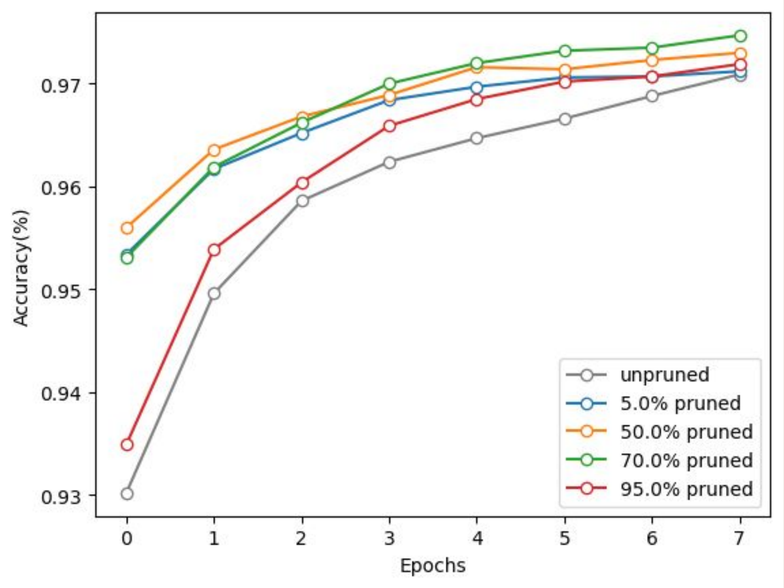
Our Deliverable

- We extended the `training` module from the digitize demo to include the following:

```python
def winning_ticket(trained_network: NeuralNetwork, copy_network: NeuralNetwork, amount: float):

    '''
    Function that takes a trained network and a previously made copy of that network
    and prunes away the lowest valued parameter weights by l1 norm. The copy network is
    modified in place to be a "winning ticket," i.e. it retains the random initialization
    values that later became highest magnitude in the trained network.
    '''
```

Identifying a “winning ticket” with one-shot pruning



- We applied one-shot pruning to a feedforward neural network on the MNIST dataset.
  - **`torch.nn.prune`** package
  - **`prune.l1_unstructured`**: removes a variable percentage of the smallest- magnitude weights in each parameter

Discussions

- Experiments demonstrate that the function learned by a neural network can often be represented with fewer parameters – a winning ticket exists.
- Winning ticket initialization is key to achieving better test accuracy than the original unpruned neural network, but initialization is also sparsity dependent.
- Limitations:
  - only finding winning tickets through sparse pruning
  - may not be as effective for larger datasets like Imagenet

Questions or Comments?

Work Cited: Frankle, J. & Carbin, M. (2019). *The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks.* ICLR 2019. arXiv:1803.03635