Project Ideas for CS242

Matheus S. Farias, Matthew J. Adiletta, Nestor D. C. Sandoval, Samuel Hsia

School of Engineering and Applied Sciences, Harvard University

Fall, 2021

Abstract

This document presents ideas for CS242's final project.

Contents

§1 Stem Cell Pruning

Stem cells, sometimes called the "master cells", are body components that can imitate the function of every cell in the human body. They have potential to repair, restore, replace, and regenerate cells, being used to treat many medical conditions and diseases.

The procedure consists on randomly initializing a network with a fraction α of active neurons, and $1 - \alpha$ non-active (stem) neurons. An active neuron is a neuron that participate both during training (backpropagation) and inference, while an non-active is just the ones that contribute during inference.

After every e epochs, we run a standard pruning (prune the less weight-valued neurons). For each pruned neuron, a number of s stem neurons are activated in the same layer. Activating a stem neuron allows the organ (layer) to recover the "dead" (pruned) neuron.

- The stop condition is when the number of stem neurons is zero.
- As we are doing pruning during training, we are converging to find a network that is meaningful to solve the problem while also being compressed.
- This procedure can find winning lottery tickets.

§2 Winning the Lottery: The Divine Comedy Approach

The Divine Comedy is an italian narrative poem written by Dante Alighieri in the fourteenth century. The story tells about Dante's soul journey towards God, beginning with the recognition and rejection (Hell), followed by the penitent Christian life (Purgatory), and finally the soul's ascent to God (Heaven).

We first randomly initialize the purgatory model, train this model until e epochs, and then save. Now, we do a search on every neuron of the purgatory, calculating the average weight of every connection to each neuron. Recall that in the standard pruning, we prune connections that are less than a threshold, so connections that have values close to zero are considered not useful. Here we assume that the opposite is also true, that is, connections with large weights are meaningful. We then copy the neuron that has the largest average and all its connections to a model called Heaven, and we assign to the neuron that has the lowest average a label of Hell, so that this neuron is not considered to go Heaven in the future.

This procedure is called the Last Judgment, it happens after every l epochs and we always copy the weight values from the purgatory model.

- After every Last Judgment, we prune two neurons during training, the stop condition is when we ran out of neurons to judge.
- We make the assumption that after each Last Judgment, the most valuable and the less valuable neurons are respectively meaningful and useless for the purgatory model.
- This procedure can find winning lottery tickets.

§3 Negative-Dijkstra Post-Pruning

One of the most well-known problems in graph theory is how to find the shortest path between two nodes of a weighted graph. This problem was solved by Edsger Dijkstra in 1956 and published in 1959. Another version of the problem is the exact opposite. How can we find the largest path? Actually, this problem is NP-Hard, except if we consider a directed acyclic graph.

Neural networks can be interpreted as directed acyclic graphs where each neuron is a vertex and each connection is an edge. The MLP architecture can be considered as a graph directed from the input to the output without cycles.

The pruning strategy consists on finding the largest path between one neuron from the first hidden layer to another neuron located in the last hidden layer. Given a neural network as a graph G, if we want to find the largest path in G, the problem is equivalent to find the shortest path in $-G$, where $-G$ is derived by G with all edges multiplied by -1. And this problem can be easily solved by Dijkstra's algorithm.

After identifying the largest path, we copy this path to another model. Note that it is possible to do inference with just this path as it takes the inputs and flows towards the output, but probably this single path is not sufficient to solve the problem. Then we can define our model by a hyperparameter N that controls the number of largest paths that we are going to copy from our pre-trained neural network without repetition.