Project report

# Weight pruning method:

1. Start with the Layer , input and vector .
2. Define hyperparameter .
3. Define , .
4. Define .
5. Pick , such that is minimal.
6. Save and in a list.
7. If , set and return to step 4.
8. Calculate and return (Rethink this entire algorithm).

Time complexity: , where is the number of weights in , is the maximal pruning we try, and using no parallelization.

Problems:

1. Taking the best step in any iteration and excluding all other steps.
2. The search might not lead to optimal results.
3. Relies on a hyperparameter which we don’t know it’s optimal value.
4. Using an approximation of the recoverability (Not necessarily a bad thing).
5. The argmax function might be a bad fit.
6. Only using 1 X. Possible solution: Choose n random vectors and calculate ethe distance on all of them, then aggregate (Using average, worst case or voting).

# Questions:

1. Do we need to prune individual weights, layers or both?
2. Ask about the 1000 layer network.
3. Ask for explanation about the kernel method and RKHS. (Previously said not required)
4. Ask for practical help with recoverability. (Show what we have already)
5. Maybe help define the project better.
6. Ask for intuition when checking recoverability of complete layers, do we want to check the recoverability between two constant spots (Like start and end of the model) or on changing spots?

# Sources:

ResNet implementation - <https://towardsdatascience.com/residual-network-implementing-resnet-a7da63c7b278>.

CIFAR-10 turorial - <https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html>.