# DEEP COMPRESSION: COMPRESSING DEEP NEURAL NETWORKS WITH PRUNING AND QUANTIZATION

Angelo Catalani 1582230

**Abstract**— Neural Network are both computationally intensive and memory intensive, making them difficult to deploy on embedded systems with limited hardware resources.
Compressing a neural network with pruning and quantization with no loss of accuracy is the aim of this project

## 1 Introduction

I have taken into consideration the following papers:

1. [1] : deals with specific pruning issues: regularization terms, threshold choice, and parameter co-adaption
2. [2] : is the sequel of the previous paper : pruning and quantization

I have run my experiment with the following two neural networks on the MNIST classification problem:

1. LeNet300-100 : two dense layers
2. LeNet5 : two convolutional layers followed by a dense layer

The loss function is the categorical cross entropy with L2 regularization and the optimizer algorithm is Adam.

## 2 Pruning

The pruning technique I have implemented as described in the papers consists of three steps :

1. train connectivity : the neural network is trained for a given number of epochs
2. prune connection : remove all the weights below a threshold
3. train the weights : re-train the reduced neural network to learn the final weights and repeat from step 2

It is significant to note that:

1. the first step is conceptually different from the way a neural network is normally trained because in this step we are interested in finding the important connection rather than the final weights
2. retraining the pruned neural network is necessary for the accuracy since after the remotion of some connection (step 2),in general the accuracy will drop
3. pruning works under the hypothesis that the network is overparametrized so that it solves not only memory/complexity issues but also can reduce the risk of overfitting

The regularization terms used in the loss function, tends to lower the magnitude of the weight matrices, so that more weights will be close to zero and good candidates for being pruned.
In particular, I have chosen L2 regularization because gives better results ([1]).
Deep neural network can be affected by the vanishing gradient problem.Even if [1] does not deals directly with that problem it notes that when a neural network struggles to update its weights consistently, it will not be able to recover in terms of accuracy and the training will be ineffective after the pruning of some neurons.
To deals with this issue, as written in [1] I have pruned the convolutional layers and the dense layers on different iterations, so that the error caused by the pruning at each iteration is attenuated.
In [1], the threshold value is obtained as a quality parameter multiplied by the standard deviation of a layer's weights.
This choice is justified by the fact that as it is the case of my experiments, the weights of a dense/convolutional layers are distributed as a gaussian of zero mean so that the weights in the range of the positive and negative standard deviation are 68% of the total.

## 3 Quantization And Weight Sharing

This section is described in [2].
After the network has been pruned, to the network is further compressed by reducing the number of bits to represent a a single weights.
In particular I have applied k-means (1 dimensional) to the weights of each layer so that the new weights of the layer are the centroids to which the original weights belong to.
Crucial to this step is the choice of k and the centroid initialization.
The choice of k is a consequence of the number of bits used in this step : if we want to compress the layers weights to n bits, we can use up to $2^n$ centroids.
The tradeoff between accuracy-compression is due to the number of bits used : the more the bits, the more the accuracy, the more the space required.
The paper describes 3 different technique :

1. BLABLABLABLABLA EXPLAIN WITH FIGURES
2. 
3. BLABLABLABLABLA

I have not implemented the fine tuning of the centroids because:

1. the loss of accuracy is inexistent
2. the latency is eccessive : for each single batch in every epoch, I should have scanned all the gradients (more than 1 minutes for a single batch on my pc)

## 4 Experiment

1) AGGRESSIVE COMPRESSION WITHOUT STD 2) WITH STD

## References

[1] Song Han, Jeff Pool, John Tran, William J. Dally *Learning both Weights and Connections for Efficient Neural Networks*
[2] Song Han, Huizi Mao, John Tran, J. Dally *DEEP COMPRESSION: COMPRESSING DEEP NEURAL NETWORKS WITH PRUNING, TRAINED QUANTIZATION AND HUFFMAN CODING*