MP-A: Model Compression

Ajinkya Mukund Nande, Karanveer Singh, Hemant Kumar Sharma

Methodology

For model compression, we used the following three methods:

i) **Magnitude Pruning**:

- This method of pruning works by removing the connections with absolute weight lower than a particular value of threshold. To implement this pruning method, first the complete model is trained. After this, pruning is performed by using the absolute value of the model weights based on a threshold, and iteratively the model is trained and pruned.

- To obtain the best model we chose the threshold of 0.05 for pruning. We iteratively trained the model for a few epochs followed by pruning.

ii) **L1-Norm Based Pruning**:

- This method works by pruning the filters based on the sum of its absolute weights. For each filter, the sum of its absolute weights are calculated, which are then sorted in ascending order. After this, m filters with smallest sums are selected and set to zero.

- To obtain the best model we chose m=3 for pruning. We iteratively trained the model for a few epochs followed by pruning.

iii) **Network slimming**:

- This method first introduces a scaling factor for each output channel of the convolution layers and then is jointly trained with the network. Finally, the channels with scaling factor less than a threshold are dropped and then the network is again fine-tuned.

- The original paper uses the affine scale factor (γ) of the BatchNorm layer applied just after convolution layer. However, the model which we were asked to abide by didn't use the BatchNorm layer. Thus, in order to implement the Network Slimming method, we first trained a new network, which is the original network with BatchNorm layers added after each convolution layer. Training this new model gave us the BatchNorm affine scaling factors. It was assumed that the “unimportant” dropped channels remain the same with or without BatchNorm layer. Finally, the originally defined model was trained without BatchNorm layers such that, for each epoch, the convolution layer weights corresponding to the “unimportant” layers were set to zero.

Comparison

For the L1-norm based pruning and network slimming methods, we only pruned the weights for convolutional layers which are only 11% of the total model parameters. As most of the parameters are in the dense layer, higher sparsity values cannot be obtained. Due to low sparsity obtained by these methods, these methods could not perform well in terms of score. As sparsity is equally weighted in score, the best model score was obtained by using magnitude pruning method. The magnitude pruning method was the easiest to implement but miraculously performed much better than the other methods.

For magnitude pruning method we were able to get the highest sparsity around 71% with a sparsity of 94%, for L1-norm based pruning method we were able to obtain accuracy around 73% with a sparsity of 0.5%, and for network slimming we could obtain accuracy around 63% and sparsity of 0.6%.

For L1-norm based pruning method, the sparsity of the model will be fixed for a particular value of *m*, as the *m* smallest filters are pruned. So, if a particular target of sparsity has to be achieved, one can use *m* to enforce that, but in magnitude pruning method, although with increase in threshold value, sparsity increases, but one cannot enforce a particular value of sparsity.

The main limitation of the network slimming method arises from the assumption that the "unimportant" channels in the network remain the same before and after removing the BatchNorm layer. This could be a major problem because the BatchNorm layer scales the convolution output channels from each layer individually. Since there is no relation established in the relative "importance" of output channels of individual convolution layers, by removing the BatchNorm layer and then dropping "unimportant" channels may cause the network to lose valuable information.

Thus, the network slimming method is not able to achieve good sparsity and accuracy compared to magnitude based pruning since its process might potentially be dropping important information.

For the magnitude pruning method, we varied the threshold between 0 to 0.12 in steps of 0.01 to obtain the plot. For L1-norm based pruning, we varied the number of filters to set 0, and obtained the sparsity vs accuracy plot. For network slimming, we varied the threshold value for scaling factor, and obtained the plot.

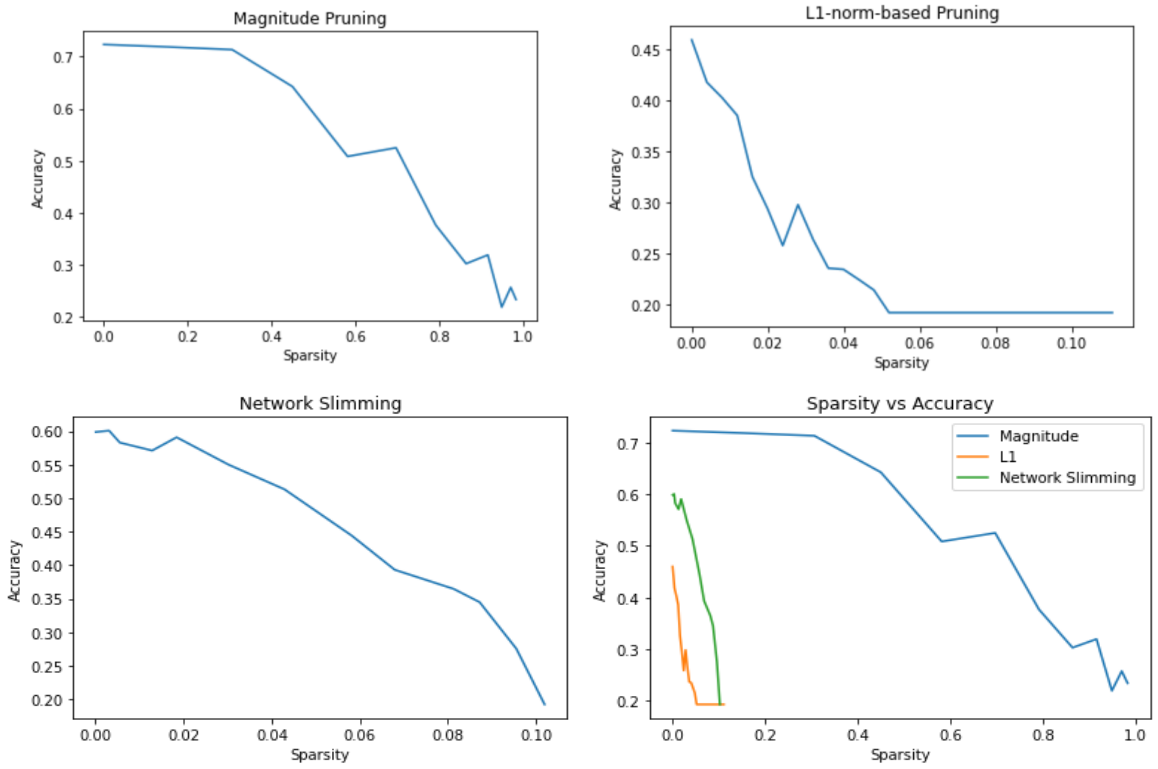

Figure: Accuracy vs. Sparsity Graphs of i) Magnitude Pruning, ii) L1-Norm Pruning, iii) Network Slimming and iv) All Methods Combined

Reflection

We definitely did not expect the results we got with Magnitude Based Pruning. Even with a sparsity of 96%, we were able to achieve more than 70% accuracy.

We learned that there is a tradeoff between sparsity and accuracy that has to be balanced and that many of the weights in the best trained model are not useful. Also, with a better pruning method one can get similar accuracy with pruned weights as obtained from complete weights. The magnitude pruning in the first glance looked very easy, but after seeing the results of the method have proved that sometimes simpler models can work way better compared to complex models.