**Pruning Techniques for Convolutional Neural Networks based on Similarity of Hidden Units**

Author Name (Paper 215 assignment 1)

Research School Of Computer Science, Australia National Universiry

**Abstract.** When we solve a variety of problems using Convolutional Neural Networks (CNNs), we usually need to choose a moderate size of each layer in the net. Especially when there are complicated problems with large data, an appropriate size of network helps us reduce the running time and improve the efficiency of training process. In this situation, pruning techniques are essential as it can remove the excess units which perform no real function in the final output of the neural network. In this paper, I made a study of these hidden units and performed an easy but powerful pruning technique to remove one kind of them in CNNs with different size. After pruning, the network has a similar good performance as before. However, it performs worse when comparing to other modern pruning techniques these days.

# Introduction

It is the generally assumption that a feed-forward CNN has two convolutional layers, two max-pooling layers and two fully connected layers in this paper. No lateral, backward or multilayer connections in the CNN. The CNN uses back-propagation for error measurement. General properties of excess units include low relevance, poor contribution, less sensitive and large badness. (T.D. Gedeon and D. Harris, 1991) When two units are too similar, one of them can be considered as an excess unit.

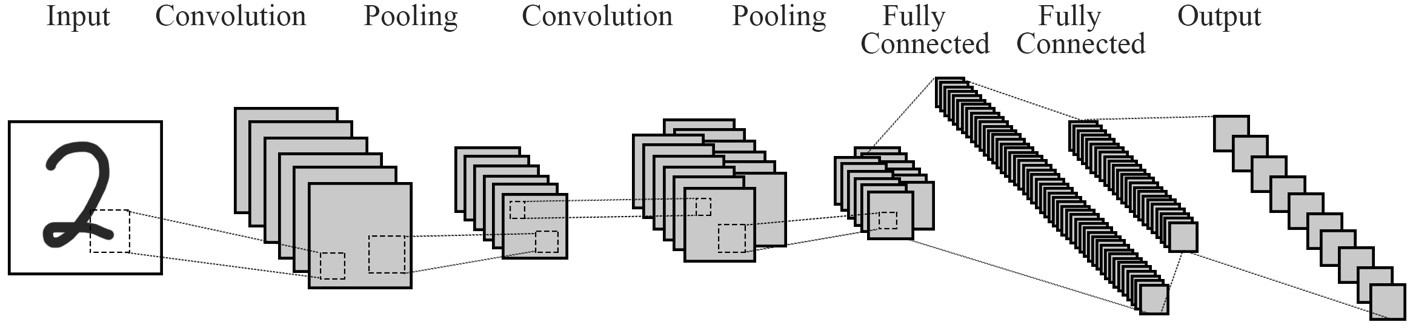
According to Gedeon’s article, vector angle is suggested as a measurement of similarity between units.

In this paper, I calculated the vector angle between pairs of hidden units in CNN’s first fully connected layer and removed one of them when the angle is beyond some certain bounds. Size of the fully connected layer is considered and three networks with different size of hidden units are chosen in the experiment. After pruning on each network, the outcome is compared with other techniques from other published paper.

# Method

Firstly, CNN basic structure with input layer, output layer, two convolutional layer, two pooling layer and two fully connected layer is built as in Figure1. MNIST is used as the dataset choice. Vector angle calculation is carried out on the first fully

connected layer of each CNN. The parameter of the first CNN I chose is shown in Table1.



**Figure 1.** CNN Structure

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Net1 | Convolutional layer1 | Pooling layer1 | Convolutional layer2 | Pooling layer2 | Fully Connected layer1 | Fully Connected layer2 |
| input | 1 | 10 | 10 | 20 | 320 | 50 |
| output | 10 | 10 | 20 | 320 | 50 | 10 |
| Kernel | 5\*5 | 2\*2 | 5\*5 | 2\*2 |  |  |

**Table 1.** 1st CNN Parameters

Without any pruning, this network performed well and get 98% accuracy on test set after 5 epoch of training sessions. After the model finished its training, I calculated the vector angle on every pair of two hidden units on fully connected layer1. Then put the output in a sigmoid function to bound it between 0 and 1. After normalising the outcome from sigmoid function to 0.5, a bound is set. If the vector angle between two hidden units is less than the lower bound or larger than the upper bound, one of the two units will be deleted. The weight of the deleted unit was set to zero and its original weight will be added to the other unit which is preserved. The vector angle bounds I used in the pruning procedure are [60o, 150 o], [65 o, 150 o], [70 o, 150 o], [75

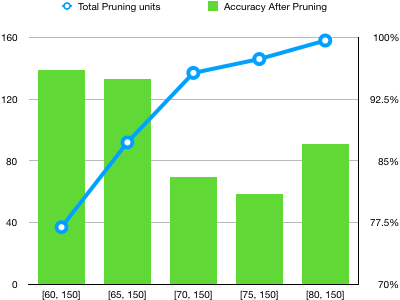
o, 150 o], [80 o, 150 o].

# Results and Discussions

The pruning result of the first network is shown in Table 2 and Figure 2.

From the results in Table 2 and Figure 2 below, it is easy to see that with appropriate pruning, the test accuracy can still stay high. And as every time at most one unit was removed from every pair if the vector angle reaches the bound, total pruning units will not exceed half of the total hidden units in fully connected layer 1. After removing almost half of the units, the network still performed well because I chose a large number of hidden units at initialization. So even half of the units are removed, it is sufficiently large number of units for CNN to predict the outcome. To

prove my guess, I implemented the same pruning techniques on the other three networks. The parameters of these three networks are shown in Table 3 and Table 4.



|  |  |  |
| --- | --- | --- |
| Bound | Pruning units (Total: 320) | Accuracy After Pruning |
| [60o, 150 o] | 37 | 96% |
| [65o, 150 o] | 92 | 95% |
| [70o, 150 o] | 137 | 83% |
| [75o, 150 o] | 146 | 81% |
| [80o, 150 o] | 158 | 87% |

**Table2.** Pruning statistics on 1st CNN **Figure2.** Relationship between pruning number

and test accuracy on 1st CNN

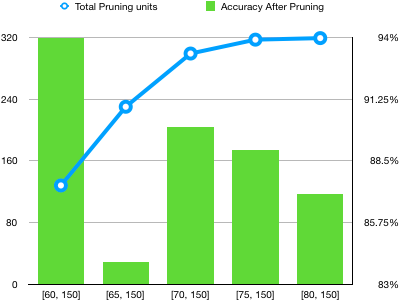
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Net2 | Convolutional layer1 | Pooling layer1 | Convolutional layer2 | Pooling layer2 | Fully Connected  layer1 | Fully Connected  layer2 |
| input | 1 | 10 | 10 | 40 | 640 | 50 |
| output | 10 | 10 | 40 | 640 | 50 | 10 |
| Kernel | 5\*5 | 2\*2 | 5\*5 | 2\*2 |  |  |

**Table3.** 2nd CNN Parameters

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Net3 | Convolutional layer1 | Pooling layer1 | Convolutional layer2 | Pooling layer2 | Fully Connected  layer1 | Fully Connected  layer2 |
| input | 1 | 10 | 10 | 10 | 160 | 50 |
| output | 10 | 10 | 10 | 160 | 50 | 10 |
| Kernel | 5\*5 | 2\*2 | 5\*5 | 2\*2 |  |  |

**Table4.** 3rd CNN Parameters

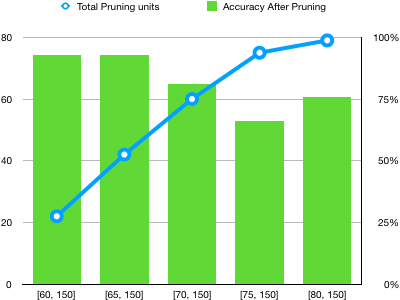
By applying 5 epoch of training and choosing the same pruning technique with same parameters as the first neural network, the pruning results of these two networks are illustrated in Table 5, Figure 3 and Table 6, Figure 4. The test accuracy of 2nd and 3rd network without any pruning are both 97%.



|  |  |  |
| --- | --- | --- |
| Bound | Pruning units (Total:640) | Accuracy After  Pruning |
| [60o, 150 o] | 128 | 94% |
| [65o, 150 o] | 230 | 84% |
| [70o, 150 o] | 299 | 90% |
| [75o, 150 o] | 317 | 89% |
| [80o, 150 o] | 319 | 87% |

**Table5.** Pruning statistics on 2nd CNN **Figure3.** Relationship between pruning number

and test accuracy on 2nd CNN

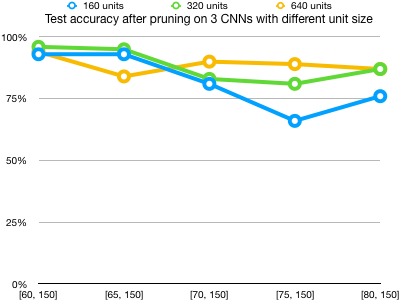


|  |  |  |
| --- | --- | --- |
| Bound | Pruning units (Total : 160) | Accuracy After Pruning |
| [60o, 150 o] | 22 | 93% |
| [65o, 150 o] | 42 | 93% |
| [70o, 150 o] | 60 | 81% |
| [75o, 150 o] | 75 | 66% |
| [80o, 150 o] | 79 | 76% |

**Table6.** Pruning statistics on 3rd CNN **Figure4.** Relationship between pruning number

and test accuracy on 3rd CNN

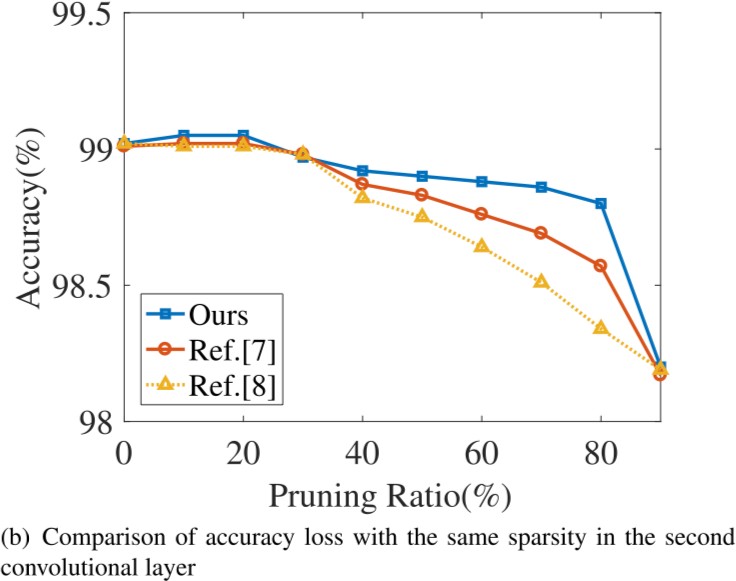
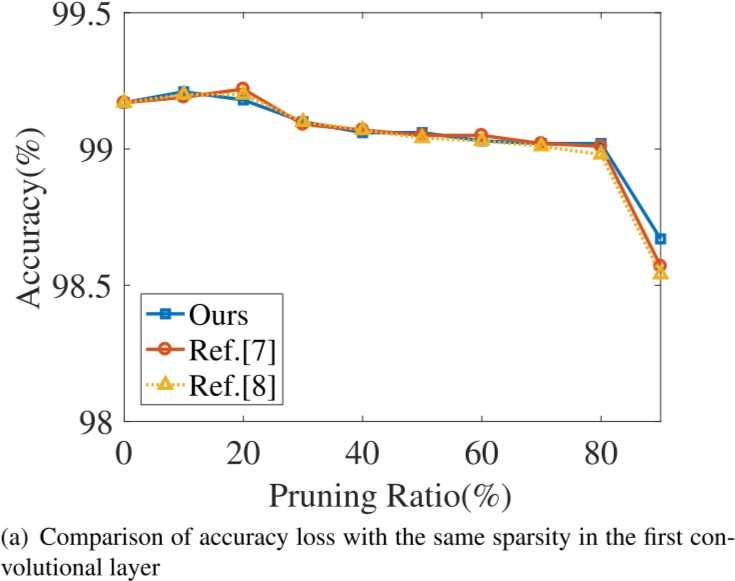
To make it more clear, the test accuracy after same pruning technique implemented on 3 CNNs with different unit size is shown as below.



**Figure5.** Comparison of the pruning influences between 3 networks with different unit size

From the diagram above, it is clear that my guess mentioned before is correct. When the CNN has sufficient number of hidden units, the pruning operation does little bad influence on the final prediction result. However, when the number of hidden units is relatively small, the pruning implemented on it may decrease the final prediction accuracy. In other words, it is better to use the pruning technique on CNNs with larger number of hidden units.

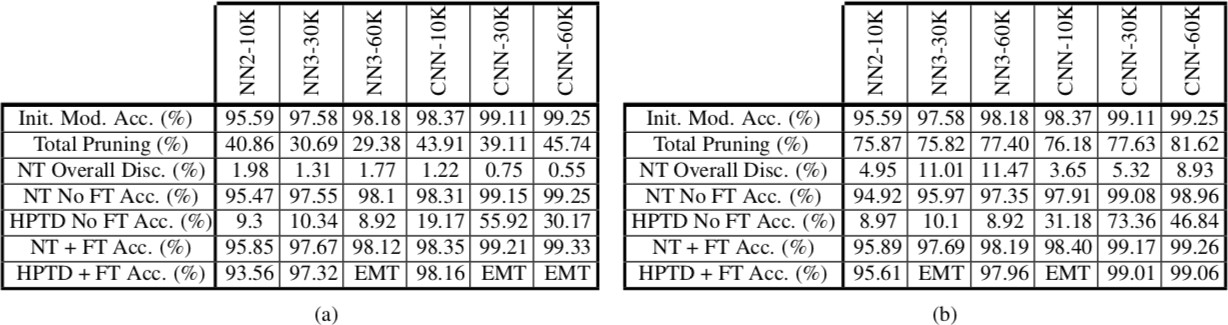
Pruning the similar hidden units using vector angle calculation is an easy but very efficient way. There are also many other ways discussed and implemented by other people. For example, with the same MNIST dataset I used in this paper, filter level pruning based on similar feature extraction using k-means++ algorithm is carried out and performed really well. (Lianqiang LI and Yuhui XU, 2018) According to Lianqiang’s paper, similar feature extraction are pruned on two convolutional layers and the test accuracy is shown below:



**Figure6.** Outcome of similar feature extraction pruning technique

From the trend graph above, we can see that similar feature extraction method using k-means++ algorithm perform better than the vector angle calculation method used in this paper. As the pruning ratio reaches 80% in Lianqiang’s paper, the accuracy of network prediction still holds a high level around 99%.

And in another paper, convex optimization program is solved at each layer of CNNs to achieve pruning. (Alireza Aghasi and Afshin Abdi) According to that study, they use the MNIST dataset too and by doing the convex optimization on different- structured CNNs, they successfully finish the pruning work with an accuracy guarantee. The tables below show their works. On each CNN with distinctive complicated structure, the prediction accuracy remains pretty high after doing a 30%~82% pruning.



**Figure7.** Outcome of convex optimization pruning technique