Neural Network pruning using hidden layers output's unit vector angles and autoencoder for feature selection

Aman Kawatra

Research School of Computer Science
Australian National University
Acton, ACT
Australia

Abstract. Neural networks can be computationally and storage wise expensive as it may require massive computations and a number of neurons split among multiple layers. Such high computational and storage resources requirements would restrict its use to selective systems. Thus, it's important that a NN can be pruned to make it more computationally and storage efficient.

A lot of techniques have been proposed and worked on, in this direction. This paper follows a approach to pruning neural network by taking into account unit vector angles between the hidden layer neurons outputs and eliminating those with similar or opposite contributions. Feature Selection has also been implemented using autoencoder making the network even more efficient while maintaining the accuracy. The experiment was conducted using a three-layer NN which as trained and tested using a stock market dataset. The experiment managed to reduce the size of the network by almost more than 36% while maintaining an accuracy of 70% in most of the cases which was not better than the referenced paper.

Keywords: Neural Network, Autoencoder, Network pruning

1 Introduction

Neural Networks have become a default standard when it comes to applications in the field of computer vision, speech recognition or Natural Language Processing [1] . Furthermore, the flexibility of neural network models while handling complex data patterns, made it suitable for use in fields of economics and econometrics [2]. However, this increased use of neural networks, also resulted in the drastic increase in the size of networks, from less than 1M parameter to classify handwritten digits in 1998 [3] to 60M parameters used by Krizhevsky to win ImageNet competition in 2012 [4].

While this increase in size was directly proportional to increase in the computational power and accuracy of neural networks, there was a downside to it as well. These large networks consume considerable computational and memory resources, which makes it becomes prohibitive for embedded mobile applications [1]. Also, the flexibility of NN models which made them suitable for financial data, comes with a downside of overfitting. Figure 1, briefly depicts the comparison of energy costs of basic arithmetic and memory operations in a 45nm CMOS process. So, if we take example of Krizhevsky's NN that won him ImageNet in 2012, running a 60M connection neural network, say at 20Hz will cost (20Hz) (0.6G) (640pJ) = 7.6W for DRAM access, which is beyond the capacity of a typical mobile device. So, for running these large NNs on mobile devices, it is important to have faster/smaller networks.

Operation	Energy [pJ]	Relative Cost
32 bit int ADD	0.1	1
32 bit float ADD	0.9	9
32 bit Register File	1	10
32 bit int MULT	3.1	31
32 bit float MULT	3.7	37
32 bit SRAM Cache	5	50
32 bit DRAM Memory	**640**	**6400**

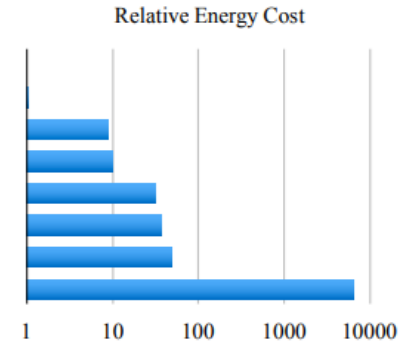

Figure 1: Energy table for 45nm CMOS process [Mark Horowitz. Energy table for 45nm process, Stanford VLSI wiki]. Memory access is 3 orders of magnitude more energy expensive than simple arithmetic. [1]

Pruning neural networks is an old idea going back to 1990 [5] and determining the structures of the NNs has always remained a serious issue [3]. Therefore, number of researcher and organization have worked for pruning the NNs and various methods have been proposed for the same [6] [7]. In this paper, the method of reducing the hidden layer neuron connection on the basis of distinctive property [8] of hidden units have been used. Also, a descriptive technique for reducing the number of input features, have also been considered in the experiment conducted. The Experiment was performed on a set of 751 patterns of Weekly stock data for Dow Jones Index, comprising the data reported by the major stock exchanges [9].

```
1,AA,1/7/2011,$15.82,$16.72,$15.78,$16.42,239655616,3.79267,,,$16.71,$15.97,-4.42849,26,0.182704
1,AA,1/14/2011,$16.71,$16.71,$15.64,$15.97,242963398,-4.42849,1.380223028,239655616,$16.19,$15.79,-2.47066,19,0.187852
1,AA,1/21/2011,$16.19,$16.38,$15.60,$15.79,138428495,-2.47066,-43.02495926,242963398,$15.87,$16.13,1.63831,12,0.189994
1,AA,1/28/2011,$15.87,$16.63,$15.82,$16.13,151379173,1.63831,9.355500109,138428495,$16.18,$17.14,5.93325,5,0.185989
1,AA,2/4/2011,$16.18,$17.39,$16.18,$17.14,154387761,5.93325,1.987451735,151379173,$17.33,$17.37,0.230814,97,0.175029
```

Figure 2: Each record (row) is data for a week. Each record also has the percentage of return that stock has in the following week (last column) (percent_change_next_weeks_price).

The goal of this exercise is to predict the percentage change of the stock price for the next week based on the last week's performance of that stock and maintain the accuracy of the results while applying the above stated network reduction technique. For this experiment, a simple neural network model consisting of three layers, an input layer, a hidden layer, and an output layer have been used. Also, as actual target data (percent_change_next_weeks_price) we use, is in the form of percentage_change/100, it has been converted to percentage_change/10 for the ease of target classification. Furthermore, the name of the stock exchanges has been given numerical values, for the ease of computation.

2 Method

The methods used in this exercise to prune the network involves pruning of the network both on the hidden layer as well as the input layer. Pruning at both the layers involve four stages as shown in figure 3. First, we start with training the neural network via normal training. The second step involves analysing the results to find the units that do not contribute much in the final results. Now, the techniques used to find, which connections and units can be pruned will be explained later in this section. The third step is to re-train the new pruned network, as shown in figure 4 and the last step is to analyse the results. This last step is of great importance, as we want to prune the network but at the same time, we want to maintain the accuracy. The experiment was conducted using a 3-layer artificial NN with the learning rate of 0.25 and initial hidden layer size of 50 neurons. Sigmoid activation function was used for hidden layer units and resilient propagation (Rprop) optimisation is used.

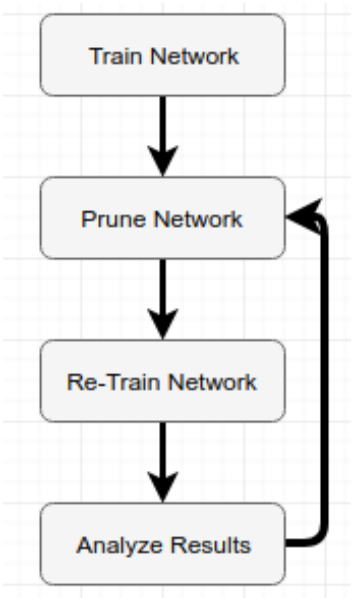

Figure 3: Four Step Training

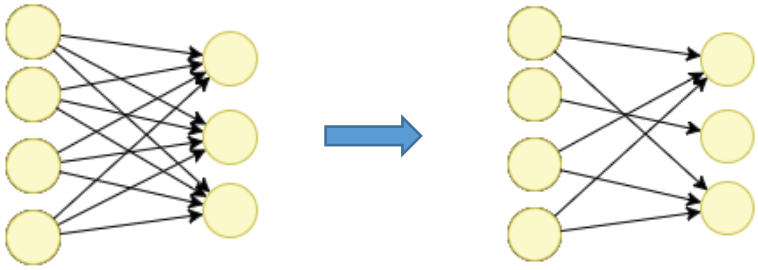

Figure 4: Pruning from Fully Connected to Sparsely Connected

2.1 Pruning Hidden layer connections and units

Hidden layer connections and units are pruned on the basis of distinctiveness of the hidden units illustrated by Gedeon & Harris [8]. Distinctiveness of hidden units is calculated using the unit output activation vector over the pattern presentation set [8]. In this method, for each hidden unit, a unit vector of the output of the activation, of dimensionality equal to the size of the training dataset i.e. number of the patterns or instances provided in the training set is constructed. Here each component of the unit vector represents the functionality of the hidden unit in the input pattern space [8]. Then based on either, the angle between the unit vectors of the units, as shown in figure 5 or, the size of activation vector, decisions were made regarding the relevance of a specific unit in the hidden layer.

```
3 & 48: 7.09395
8 & 17: 14.5598
8 & 48: 9.64074
11 & 33: 12.777
13 & 48: 12.8755
25 & 36: 169.139
27 & 36: 166.11
36 & 38: 165.279
2 & 38: 167.328
```

Figure 5: Pair of hidden layer units and their respective unit vector angles

The decisions regarding the relevance of a unit in hidden layer was made through the three major observations stated below:
1. If a unit have a short activation vector in the pattern space, it can be considered insignificant and can be removed.
2. If the angle between the unit vectors of two units is less than 15º, then the units can be considered having similar functionality and one of them can be removed.
3. If the angle between the unit vectors of two units is more than 165º, then the units can be considered having complimentary functionality and both of them can be removed.

Now, that we know the connections to prune, the weights of the corresponding network connections are updated to make sure that irrelevant hidden units get zero weight. The weight distribution curves in figure 6 and 7, shows the distribution of weights among the connections from input layer to hidden layer before and after the weights have been updated to prune the network.

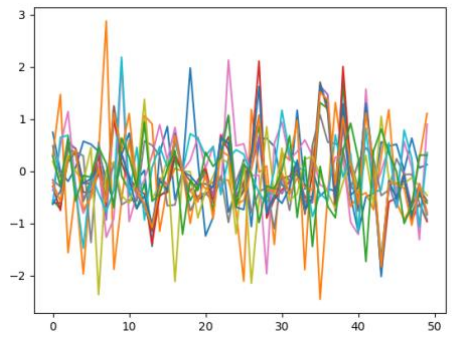

Figure 6: Weight Distribution before pruning

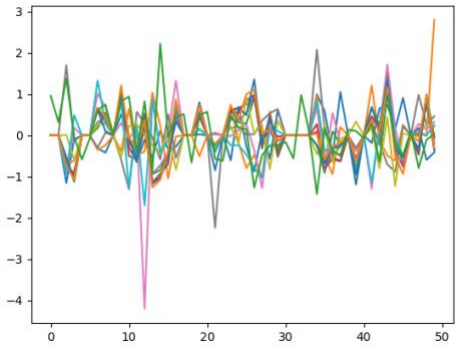

Figure 7: Weight Distribution after pruning

2.2 Pruning Input layer units

Input features are pruned by using autoencoder. An autoencoder is a multi-layer network where the aim of hidden layer is to reconstruct the input data at the output layer which means the hidden layer have to extract the most relevant information. This encoding and then decoding of the input feature forces the autoencoder to engage in dimensionality reduction. For this experiment, a 3-layer autoencoder was used with the hidden layers having variable units which were then fed as input features to the NN for stock market prediction.

For Both pruning at the input layer and hidden layer, the network is re-trained and tested on variable values of learning rate, batch size and number of epochs, to test and compare the accuracy and error rate of the reduced/pruned neural network to the one before pruning. Also, the training and testing set for the experiment was randomly split in ratio of 80% and 20%, respectively.

3 Results and Discussion

3.1 Results and Evaluation of Pruning at hidden Layer

Experiment conducted to prune the Hidden layer inputs on the basis of distinctive property showed some promising results. The trade-off curve between accuracy Loss and percentage of hidden units removed after re-training is shown in Figure 8. The accuracy did not drop much when the incoming and outgoing connections from the hidden layer units, that were found to be irrelevant, were removed from the network. This shows that these units did not contribute much in final results. Also, the results highlighted the importance of re-training, as the accuracy dropped much faster when the model is not re-trained on the reduced NN.

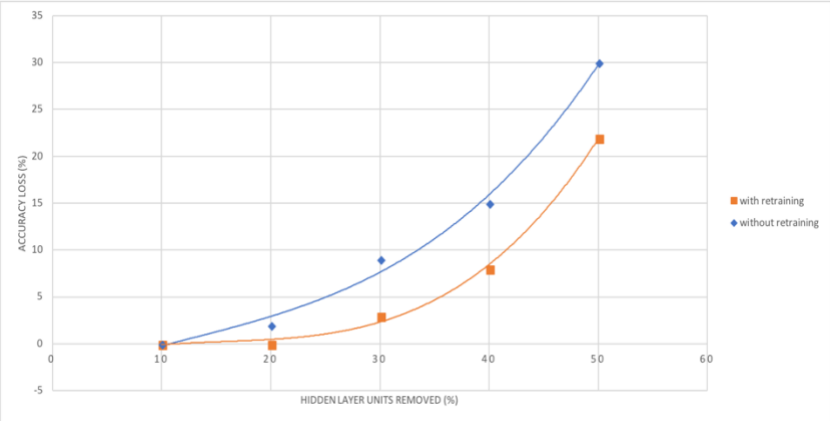

Figure 8: Accuracy Loss and Hidden layer Units Removed Trade-off

The Reduced Network after pruning the hidden layer, left a network that have 28% less connections, with almost no accuracy drop.

3.2 Results and Evaluation of Pruning at Input Layer

Experiment conducted to prune the features that is feed as input in the neural network using the autoencoder technique showed some promising results. The trade-off curve between accuracy Loss and percentage of input features summarised/reduced by the autoencoder after re-training is shown in Figure 9.
The accuracy of the network did not drop much fast when the input features were summarised by the autoencoder to 75% of the total input features, but it dropped faster when the input features were summarised even further by the autoencoder

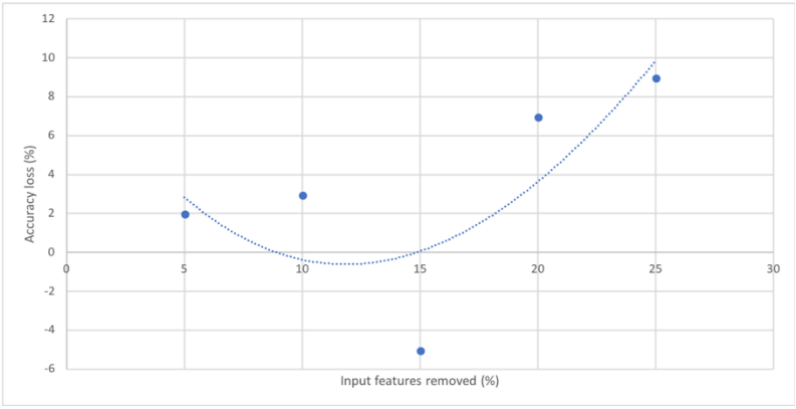

Figure 9: Accuracy Loss and Input Features Removed Trade-off

Surprisingly, the accuracy increased by a small percentage when we summarised the input features by 15%. This shows that using more meaningful features can actually improve the accuracy for classification tasks. The experiments showed that, the input layer pruning using autoencoder technique can further reduce the network by 5% with a small increase in the accuracy and the network can be reduced further by 7% with almost 5% accuracy loss.

The results shown above clearly depicts that the neural networks can be reduced using the above used techniques, but there are other techniques like, Pruning using the concept of incremental contribution of explanatory variable [2] [10] which showed better results on a similar kind of economic dataset. Also, creating a taxonomy of undesirable units [8] and then pruning on the basis of it rather than only relying on the distinctiveness of the hidden units can also give better results.

4 Conclusion and Future Work

A novel approach to reduce/prune a neural network by reduction both at hidden layer level and input layer level have been shown in this paper. The above explained experiment proves that the used techniques give good results for network reduction/pruning. The explained methods, allowed us to reduce 36% of the network connections while maintaining the accuracy of the network on the test set, which was 70% in most of the cases. Note that these values can differ for different datasets and for different network configurations. Also, the experiment also showed that the above used network was pruned by almost 15% with a small increase in the accuracy when using autoencoder technique.

I end with listing some topics for further research. An Extension of finding irrelevant units and then pruning them would be, to be able to rank the hidden units on the basis of their contribution and then performing iterative pruning on them [11]. In this scenario, I would like to work on multi-layer networks in comparison to three-layer network used in this paper, so as to be able to compare the percentage of pruning based on how deep the layer is in the network [12]. Furthermore, I will also like to implement a variational autoencoder and denoising autoencoder, and a combination of these two type to see if that improves the performance even further and to what extent.

References

[1] S. Han, J. Pool, J. Tran and W. Dally, "Learning both Weights and Connections for Efficient Neural Network," in *Advances in Neural InformationProceeding Systems 28*, 2015.

[2] J. F. Kaashoek and H. K. Van Dijk, "Neural Network Pruning Applied to Real Exchange Rate Analysis," *Journal of Forecasting,* vol. 21, pp. 559-577, 2002.

[3] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of IEEE*, 1998.

[4] A. Krizheysky, I. Sutskever and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *In Advances in neural information processing systems,* pp. 1097-1105, 2012.

[5] Y. L. Cun, J. S. Denker and S. A. Solla, "Optimal Brain Damage," AT&T Bell Laboratories, Holmdel,N.J. 07733.

[6] J. Sietsma and R. Dow, "Creating artiċcial neural networks that generalize," *Neural Networks,* vol. 4, no. 1, pp. 67-69, 1991.

[7] K. Kameyama and Y. Kosugi, " Neural network pruning by fusing hidden layer units," *Trans. IEICE,* vol. 74, no. 12, pp. 4198-4204, 1991.

[8] T. D. Gedeon and D. Harris, "Network Reduction Techniques," *Proceedings International Conference on Neural Networks Methodologies and Applications,* vol. 1, pp. 119-126, 1991.

[9] M. S. Brown, M. Peloski and H. Dirska, "Dynamic-radius Species-coserving Genetic Algorithm for the Financial Forecasting of Dow Jones Index Stocks," *Machine Learning and Data Mining in Pattern Recognition,* pp. 27-41, 2013.

[10] H. Theil, "Principles of Econometrics," Wiley, New York, 1971.

[11] J. Gil, "Pruning deep neural networks to make them fast and small," [Online]. Available: https://jacobgil.github.io/deeplearning/pruning-deep-learning. [Accessed April 2018].

[12] P. Molchanov, S. Tyree, T. Karras, T. Aila and J. Kautz, "PRUNING CONVOLUTIONAL NEURAL NETWORKS FOR RESOURCE EFFICIENT INFERENCE," in *ICLR*, 2017.