Deep Neural Network Reduction Based on Layer Activation Vector Angles

Jonathan Ventigan
College of Engineering and Computer Science
Australian National University
Canberra, ACT 2600, Australia
u7099070@anu.edu.au

Abstract. Pruning of network nodes was done on a fully connected, feed-forward, deep neural network trained on a small geographic dataset. Removal of nodes was done on the input and penultimate layers, based on activation vectors over input space. The network was unable to learn from the limited dataset and any impact on network performance of the removal of neurons could not be determined. Interestingly, angles between activation vectors showed behaviour associated with a network continuing to learn from data.

Keywords: deep neural network, network pruning, activation vector, classification

1 Introduction

Interest in reducing network complexity comes from a need to improve efficiency and decrease training time.[1] These considerations become more important for deep neural networks which have multiple nodes and layers. Methods of pruning neural networks include Optimal Brain Damage (OBD), which removes parameters from a network through the concept of "saliency" of parameters, based on the second derivatives of the error function with respect to the network weights along the diagonal of the Hessian matrix[2], and Optimal Brain Surgeon, an extension of OBD that includes all parameter second derivatives (not just the diagonals) in the Hessian matrix to determine saliency.[3] Another approach would be to combine machine learning approaches, such as using a genetic algorithm to determine the optimum number of nodes and hidden layers, with a fitness function based on both accuracy and network complexity.[4]

The most difficult hyperparameter to determine is the number of nodes in the processing layers. Network nodes may be removed either because they are redundant or produce little to no output. The former can be identified based on their activation vectors, which are the output pattern of nodes over a network's input.[1] A previous work by Gedeon applied this technique to pruning a single hidden layer image autoencoder in order to achieve compression.[5] The present work applies this technique to a deep neural

network on a classification problem involving a small geographic dataset. As an initial attempt, network pruning is attempted on the input layer and the last hidden layer.

2 Methods

2.1 Dataset

The dataset is the same as the one used in a previous study by Milne et al.[6] It contains 190 datapoints and each datapoint corresponds to a 30 m^2 patch of land in Nullica State Forest in New South Wales. The data was taken from soil maps, aerial photographs and satellite images. The features of the data include altitude, aspect, slope, geology, topographic position, rainfall, temperature, and Landsat TM bands 1 to 7. Each datapoint is labeled with the one of five forrest categories: scrub, dry sclerophyll, wet/dry sclerophyll, wet sclerophyll and rainforest.

For the purposes of this paper, the labels were simplified into two categories: dry sclerophyll and other types. Dry sclerophyll comprises 50.5% of the data so using this binary label makes the dataset nearly balanced. The continuous variables, which are altitutde, slope, topographic position, rainfall, temperature and Landsat TM bands, were scaled to a range of 0 to 1. Aspect was encoded into four values according to figure 1 below. This is the same as in Milne et al.[6]

As	Dir.	A1	A2	A3	A4	Σ activ
0	Flat	0	0	0	0	0
10	N	1	0.5	0	0.5	2
20	NE	1	1	0	0	2
30	E	0.5	1	0.5	0	2
40	SE	0	1	1	0	2
50	S	0	0.5	1	0.5	2
60	SW	0	0	1	1	2
70	W	0.5	0	0.5	1	2
80	NW	1	0	0	1	2

Fig. 1. Table of encoding for the aspect feature. Reproduced from Milne et al. [6]

The geology feature was one-hot encoded into four categories: three of the most common categories and a bucket category for the rest of the values. The three most common categories comprise 82.6% of the dataset, which makes the aformentioned encoding reasonable. This is also the same as in Milne et al.[6]

The end result of this preprocessing was that the input vectors contained 20 features of mixed continuous and categorical variables and the labels were single values of either 0 or 1 for non-dry sclerophyll and dry sclerophyll forest types respectively. The dataset was

randomly shuffled before being divided into training, validation and test sets. Seventy percent (n = 133) of the dataset was set aside for training and the remaining datapoints were split into the validation (n = 28) and test sets (n = 29).

2.2 Neural Network

A fully-connected, feed-forward, deep neural network with six hidden layers was used for testing. This network was implemented using the PyTorch machine learning library in Python. The input layer contained 20 nodes with a leaky rectified linear unit (LReLU) activation function. The subsequent six layers contained 25 nodes each and each layer also used the LReLU activation function. Batch normalization and dropout were implemented after each hidden layer with the dropout rate set at 80%. The final layer had a single node with a sigmoid activation function. The error criterion used was binary cross entropy(BCE) loss. Network weights were initialized using He initialization and were updated using the Adam algorithm with a learning rate of 0.01.

2.3 Test Setup

Five-fold cross validation was first performed to estimate the number of epochs needed to train the network. Training was then done for 70 epochs, in mini-batches of ten datapoints with random shuffling. During training, Gaussian noise was added to the inputs to try to improve network generalization. Network performance was evaluated after training using the test set.

Activation vectors were measured for both the input layer and the last hidden layer. The training set was used as input to get these activation vectors. Angles between activation vectors were calculated in order to identify similar or complementary nodes. Two nodes are considered similar if the angle between their activation vectors was less than 15° and complementary if the angle was greater than 165°, as described in Milne et al.[6] When two nodes are found to be similar or complementary, the weights of one are added to the other before the former node is deleted. Nodes were deleted in the input and last hidden layers separately, with deletions continuing as long as there were similar or complementary nodes. Network loss was reevaluated after each deletion.

Lastly, to observe the behaviour of the network during training, activation vectors of the input and last hidden layers were measured for every epoch of training up to 4,000 epochs. Five percent of these angles were chosen for evaluation.

3 Results and Discussion

3.1 Results

The results of five-fold validation are shown in figure 2 below. There is plateauing of validation losses after approximately 20 epochs, with these losses remaining above

training losses throughout. The same results can be seen during network training, as shown in figure 3. After training, loss and accuracy using the test set were measured in ten trials, resulting in a mean loss of 1.4354 (range 1.0263 – 2.0795) and mean accuracy of 49.31% (range 37.93% – 55.17%) for the network. A confusion matrix for a single trial is shown in table 1.

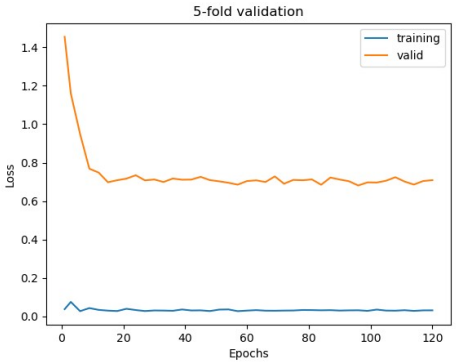

Fig. 2. Five-fold validation of the deep neural network with training and validation losses plotted.

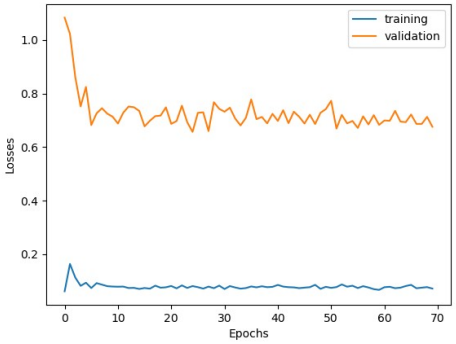

Fig. 3. Training and validation losses during network training

Table 1. Confusion matrix for the trained network on the test set (DS – dry sclerophyll)

	DS actual	Non-DS actual
DS predicted	3	11
Non-DS predicted	8	7

Interestingly, despite the low accuracy and failure of the model to generalize, a look at the activation vector angles at the last hidden layer shows large changes up to 4000 epochs. The same can be seen with the activation vector angles at the input layer but less pronounced. This behaviour is associated with a model that is at a shallow, local minimum and will learn eventually.[5] (See figure 4 below.)

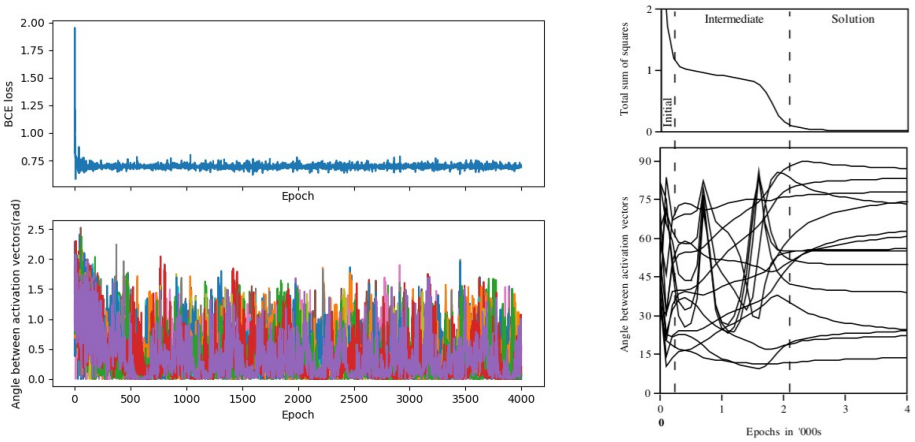

Fig. 4 (Left) A sample of angles between activation vectors of the last hidden layer show large variations as training progress. (Right) This behaviour is seen in networks that eventually learn, as reported by Gedeon. (reproduced from the same)[5]

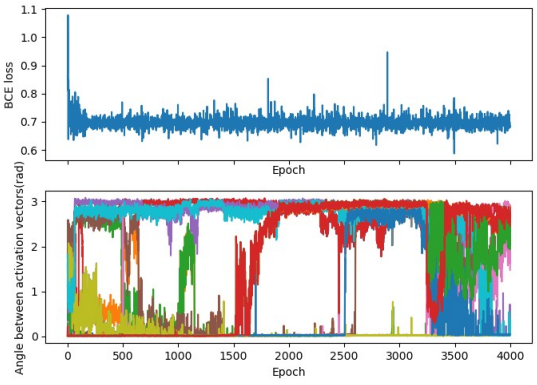

Fig. 5 Angles between activation vectors of the input layer. (also a sample) Some of the angles show only minimal changes during training.

Attempts were made to remove nodes from the last hidden layer but none of the angles met the threshold for similarity or complementarity. Attempts at pruning of the input layer were more fruitful and a number of node pairs were found to be similar or complementary. Results of the pruning are seen in figure 6.

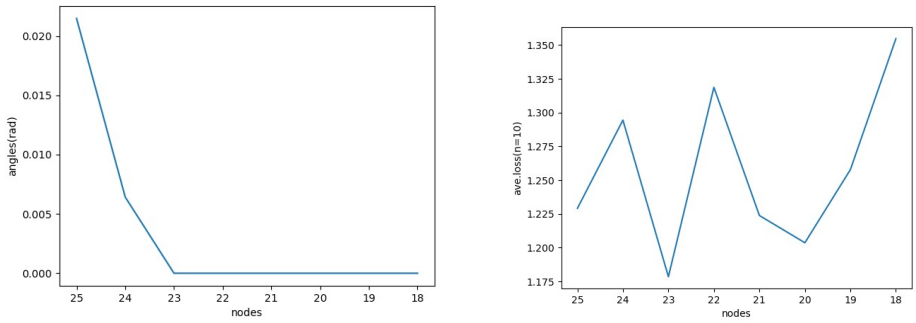

Fig. 6 (Left) Minimum angles as nodes are removed from the input layer (Right) Average losses after each node removal.

Mean loss at the end of pruning was 1.3547 (range 0.7917 – 1.9766) and mean accuracy was 49.31% (range 34.48% – 62.07%).

3.2 Discussion

In the paper by Milne et al, their neural network achieved accuracies of 52.6% on the training set and 65.7% on the test set.[6] For a quick comparison, we also created a three-layer, fully connected, feed-forward neural network with the sigmoid activation function for the hidden layer, BCE loss as the error criterion, and stochastic gradient descent for weights update. Results were similar for the deep neural network with a mean loss of 1.3680 and a mean accuracy of 55.17%. As the dataset is relatively well-balanced, this performance is virtually no better than chance. Deep neural networks are supposed to be able to perform well even with small datasets, as shown in a paper by Olson, Wyner and Berk[7] (this being the motivation for the current study). However in that paper, 50% of the datasets used had only 15 features or fewer and the number of datapoints in the datasets number up to 67,557. The problem in our model is likely a combination of insufficient amount of training data as well as having too many features, with network falling victim to the "curse of dimensionality". Depending on the nature of the problem, the number of training data required can range from a quadratic to an exponential function of the number of features.[8]

No similar or complementary neurons were found in the final hidden layer. Additionally, the final hidden layer showed behaviour of a network still undergoing

learning. This behaviour may be the result of a cliff structure in the parameter space of the network, resulting in large updates to the parameters and large changes to the output of the final hidden layer. The behaviour is more common in recurrent neural networks, though. [8] More simply, it may be that the final hidden layer's output is a function of the multiplication of many weight parameters from previous layers and hence, shows greater variability. Plus, the model may have gotten stuck at a local minimum.

In contrast, a number of similar neurons were found in the input layer. During training, several nodes also showed no change in the angles between their activation vectors. This redundancy in neurons may also be an indication to the potential for reducing the number of features in the inputs. Unfortunately, the poor performance of the network precludes evaluating the effect of network pruning on performance. Even with pruning of neurons at the input layer, measured losses were still within range of losses seen in the intact network.

4 Conclusion and Future Work

With the given dataset, a deep neural network is still insufficient even for a binary classification task. As such, the effect of network pruning on a deep neural network cannot be properly evaluated. The most significant cause of the poor performance of the network is likely the large number of features in the inputs. To remedy this, an encoder might be used to provide inputs of reduced size to a model or a genetic algorithm might be employed to select the optimum combination of features to use as input. Other methods that may be employed to improve performance include gradient clipping, if a steep gradient is indeed a problem, and reduction in the number of hidden layers, which may reduce overfitting. Alternatively, if the only objective is to evaluate network pruning, a different and well-studied dataset may simply be used instead, one with fewer categories and features. Measurement of the angles between activation vectors with training seem to indicate that redundant or complementary nodes can be found more in shallower layers. Future work might include measurement of angles and pruning on a per layer basis. Ultimately however, determining through trial and error or exhaustive search what combination of layers and nodes to prune without affecting performance is impractical for deep neural networks. A theoretical understanding of the effects of pruning on the output of downstream layers and nodes is worth pursuing.

References

1. T. Gedeon and D. Harris, "Network Reduction Techniques", in Proceedings International Conference on Neural Networks Methodologies and Applications, AMSE, vol. 1, pp. 119-126, San Diego, 1991
2. Y. Le Cun, J. Denker and S. Solla, "Optimal Brain Damage", in Advances in Neural Information Processing Systems, D. Touretzky, Ed. Morgan-Kaufman, 1990. pp. 598 – 605

3. B. Hassibi, D. Stork and G. Wolff, "Optimal Brain Surgeon and General Network Pruning", in IEEE International Conference on Neural Networks, San Francisco, CA, USA, 1993, pp. 293-299.
4. D. Stathakis, "How many hidden layers and nodes?", International Journal of Remote Sensing, vol. 30, no. 8, pp. 2133-2147, 2009. Available: 10.1080/01431160802549278 [Accessed 30 May 2021].
5. T. Gedeon, "Indicators of Hidden Neuron Functionality: the Weight Matrix versus Neuron Behaviour", Australasian Journal of Intelligent Information Processing Systems, vol. 3, no. 2, pp. 1 - 9, 1996
6. L. Milne, T. Gedeon and A. Skidmore, "Classifying dry sclerophyll forest from augmented satellite data: Comparing neural network, decision tree and maximum likelihood", in Proceedings 6th Australian Conference on Neural Networks, pp. 160 – 163, Sydney, 1995
7. Olson, M., Wyner, A., Berk, R., “Modern Neural Networks Generalize on Small Data Sets”, in 32nd Conference on Neural Information Processing Systems, NeurIPS, vol. 31, Montréal, Canada, 2018
8. C. Bishop, Pattern Recognition and Machine Learning. New York: Springer-Verlag, 2016, pp. 33-38.
9. I. Goodfellow, Y. Bengio and A. Courville, Deep learning. Cambridge, MA: The MIT Press, 2016, pp. 288-289.