Indicators of Hidden Neuron Functionality: the Weight Matrix versus Neuron Behaviour

T.D. Gedeon [1]
Centre for Neural Networks
King's College
The University of London

Abstract

Pruning of redundant or less important hidden neurons from the popular back-propagation trained neural networks is useful for a host of reasons, ranging from improvements of generalisation performance, to use as a precursor for rule extraction. For pruning it is necessary to identify hidden neurons with similar functionality.

We have previously used a pruning process based on the behaviour of the hidden neurons in an image processing application to produce a quality driven compression by eliminating the least different hidden neurons.

In this work we consider the computationally cheaper alternative using only the trained weight matrix of the neural networks at each stage of the compression process.

We conclude that the weight matrix is not sufficient for differentiating the functionality of the hidden neurons for this task, being essentially the functional equivalence problem which is computationally intractable.

Assumptions

In this paper we will generally assume a feed-forward network of three layers of neurons. All connections are from neurons in one layer to the subsequent one, with no lateral, backward or multilayer connections. Each neuron has a simple weighted connection from each neuron in the previous layer. The network is trained as an auto-associator using a training set of input patterns with desired outputs being the same as the inputs, using back-propagation of error measures [1]. Training by back-propagation is popular because of its simplicity theoretically, and the ease of use and production of such networks. The method has been used successfully in a number of disparate areas ranging from pattern synthesis [2], to image compression [3-5].

Introduction

The major disadvantages of the back-propagation method are that it can be slow to train networks, and that the architecture required for a solution to a problem is not currently determinable *a priori*. In practice, it is deciding the number of hidden neurons which is most difficult. Many workers have remarked that to train networks successfully or at an effective rate, more hidden neurons are required than the minimal number. Any extra neurons which end up duplicating the functionality of existing neurons do nothing but decrease the speed of the network by increasing its size. Also, significant time is lost for restarting the training process from scratch.

Brute force methods to find minimal size networks by eliminating randomly chosen neurons from trained networks have been used; recently some approaches have emerged delineating properties to choose which neurons should be eliminated.

The seminal work on pruning trained networks [7] uses the outputs of neurons in a two stage pruning process, which operates by inspection. Such pruning by inspection is difficult even on small examples, some automatable process would be ideal. Properties such as *relevance* [8, 9], *contribution* [10], *sensitivity* [11], *badness* [12], and *distinctiveness* [13] have been described in detail elsewhere.

We will briefly describe *distinctiveness* here, as it is our own, and conceptually one of the simplest, and also forms the basis of comparison for the rest of this work. Note that all of the methods are at least as computationally expensive as our own method.

The *distinctiveness* of hidden neurons is determined from the neuron output activation vector over the pattern presentation set. That is, for each hidden neuron we construct a vector of the same dimensionality as the number of patterns in the training set, each component of the vector corresponding to the output activation of the neuron. This vector represents the functionality of the hidden neuron in (input) pattern space.

In this model, vectors for clone neurons would be identical irrespective of the relative magnitudes of their outputs and will be recognised. In normal pruning, angular separations of up to about 15° are considered too similar and one of them is removed. The weight vector of the neuron

1 On leave from The University of New South Wales

0-8186-7174-2/95 $04.00 © 1995 IEEE

which is removed is added to the weight vector of the neuron which remains. With low angular separations as above, the averaging effect is insignificant and the mapping from weights to pattern space remains adequate in that the error measure is no worse subsequently.

This produces a network with one fewer neuron which requires no further training. Similarly, neurons which have an angular separation over about 165° are complementary, and both can be removed. In this work we are not interested in distinguishing the different forms of similarity (ie complementarity), hence angular separations over 90° are mapped back to the 0° to 90° range, this range linguistically is from *identical* to *orthogonal* in behaviour.

Application domain

In the image compression application, we use a feed-forward neural network we call *auto-associative* network, as the same patterns are used as the input and target patterns. The hidden layer consists of fewer neurons than the input layer, thus compressing the image.

As the output layer is the same size as the input layer, it is used to recover the compressed image. Clearly this is a form of lossy compression, with the loss in quality on decompression being dependent on the overall goodness of the (generalised) representation the hidden neurons manage to form of the patterns being compressed.

In the case of image compression, the degree of compression desired could be used to determine a hidden layer size. Given the earlier observations, however, we should use a somewhat larger size initially. Nevertheless, for the measure of the degree of compression to be meaningful, any neurons with redundant functionality after training must be removed. Otherwise, results are not comparable between different training regimes, as well as inconsistent using the same regime, given that with different initial random weights, a different number of functionally useless neurons may result.

Brief summary of previous work

A 64 by 64 image with 4 bits per pixel was chosen for testing our method. The image was broken into 16 non-overlapping 16 by 16 images, as shown in Figure 1. Each 16 by 16 image is a separate training pattern.

The image was broken into non-overlapping pieces so as to allow for generalisation to have clearly occurred – since there is little obvious similarity between the pieces, and particularly the large areas of white space around the edges of the image could be expected to interfere. This allowed a single image to be used, thus simplifying the discussion.

Note the tall profile of the pixels, due to using a subset of ASCII varying in image density to represent gray levels. The image was captured by the author off the electronic net in the U.K. – the picture originated in the U.S. and had been distributed worldwide.

The network architecture used was a 256 input, x neuron hidden layer, and 256 output network, each of the pixels in the 16 by 16 parts of the image being a single input, and output. The number of bits per pixel were mapped onto the range from 0 to 1 as input, and the output values remapped to the simulated gray scale. The initial value of x was 16.

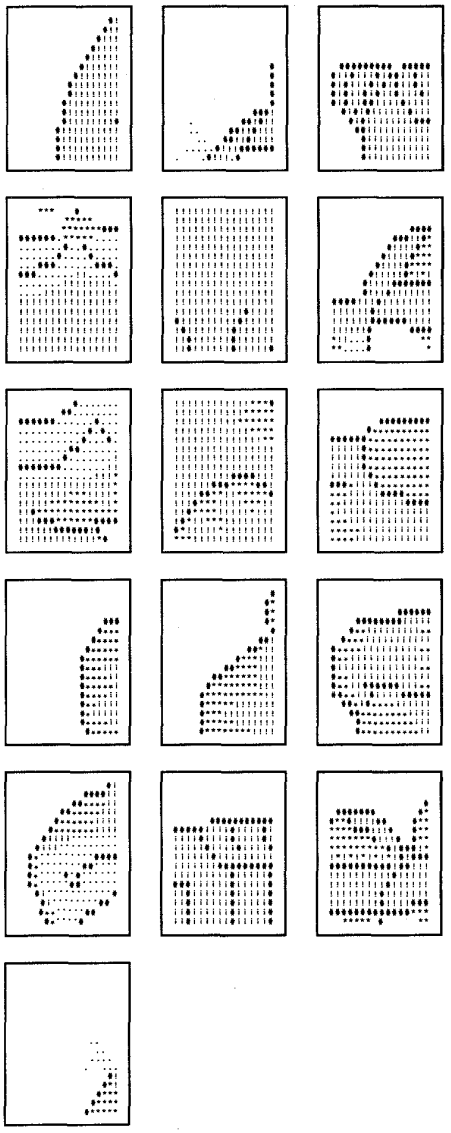

Figure 1. Broken up image

This gives a minimum compression ratio of 16 to 1, which is reasonable given the (deliberately) generalised network architecture used. The picture is clearly recognisable in the 7 neuron case at a compression ratio of 37 to 1, and

only marginally recognisable in the 6 neuron case, which is at a compression ratio of 43 to 1. The use of overlapping patches, or training on a number of similar entire images [14] would boost the compression ratio of both stages of our process. The network was initially trained for 200 presentations of all 16 patterns.

Using the *distinctiveness* angular measures, we progressively reduce the size of the hidden layer. Beyond a certain point, the neurons we remove are significantly distinct – we are trading image quality for degree of compression.

Figure 2 shows the relationship between the number of hidden neurons and the total sum of squares error measure which can be taken as a rough indicator of the image quality. Initially there is little drop in quality as neurons are removed. Subsequently, each further neuron removed produces a significant degradation in quality.

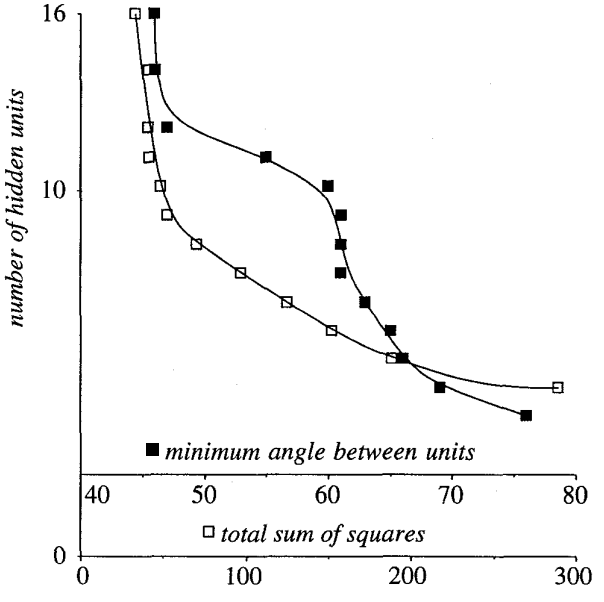

Figure 2. Number of neurons versus image quality and neuron significance

Figure 2 also shows the relationship between the number of hidden neurons and the smallest angle between hidden neurons.

These values are all much higher than the standard 15° we use for pruning redundant neurons. It is interesting to note that the removal of the first few neurons in the vicinity of 60° does not cause a marked reduction in the error measure. These neurons are the equivalent of the secondary backup neurons we introduced for increasing network damage resistance in [15]. Since we are removing significant neurons at each stage, the network will require retraining. After the removal of a neuron, the network is trained for 200 epochs.

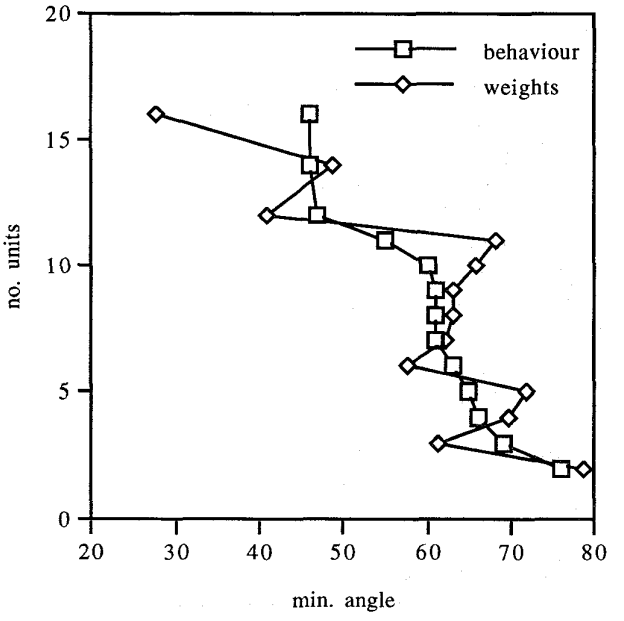

Figure 3. Weights and behaviour during pruning

Results and conclusions

In another application domain, we have used the weights on the input connections to the hidden neurons successfully to differentiate between significant inputs and relatively less significant inputs [16]. In that manner, here the weights connecting the hidden neurons and the 256 neuron output layer are normalised between the global maxima and minima found, and then used to form 256 dimensional vectors. The angles between these vectors are then calculated an in the *distinctiveness* approach. The results are shown in Figure 3.

Clearly, the weight matrix measure is coarsely approximating the behaviour of the hidden neurons over the experiment progressively reducing the number of neurons.

This is due to the use of the static weight matrix, as it ignores the smoothing effect of the transfer function of the hidden neurons. We may also speculate that there may be many possible weight matrix configurations giving rise to similar neuron behaviour over the range of patterns likely to be encountered.

These alternative matrix configurations for different neurons may explain the erratic weight curve in Figure 3. That is, the actual implementation of the function within the black box is different. Note that the implementation is subtly different over time within the same network, as shown by the differing distances between the two curves in Figure 3.

We can conclude that the weight distinctiveness on the static weight matrix is not sufficiently fine an instrument for differentiating the functionality of the hidden neurons for this task, and we are forced to continue the use of

computationally more expensive approaches which use measures based on dynamic network behaviour. We can conclude this because we have used te same instrument (distinctiveness analysis) on both the true functionality of the neural network black box being its behaviour, and on its innards, being the static weight matrix.

Obviously the static weight matrix together with the training patterns contains all of the information which provides the dynamic behaviour of the network, the issue is thus whether we need to run the network and observe its behaviour or can we use its static properties. We have concluded here that the former is necessary.

Alternative approaches attempting to discover whether two networks are identical in functionality whether by algorithmic means or by using another neural network would be unlikely to succeed given our observations that even in a single network where the observed functionality changes smoothly as more neurons are removed, but the same analysis of the weight matrix shows discontinuous changes in the weight matrix.

Further investigation is required as to the size of subset of the training set which will still provide a reasonable prediction of the functionality of the hidden neurons in a network.

References

1. Rumelhart, DE, Hinton, GE, Williams, RJ, "Learning internal representations by error propagation," in Rumelhart, DE, McClelland, *Parallel distributed processing*, Vol. 1, MIT Press, 1986.

2. Lewis, J, Probing the Critic: Approaches to Connectionist Pattern Synthesis, *Proceedings International Joint Conference on Neural Networks*, vol. 1, pp. 85-89, Seattle, 1991.

3. Cottrell, G, Munro, P, & Zipser, D, "Learning internal representations of gray scale images," *Proc. 9th Ann. Cog. Sci. Soc. Conf.*, Seattle, pp. x-y, 1987.

4. Namphol, A, Arozullah, M, & Chin, S, "Higher Order Data Compression with Neural Networks, *Proceedings International Joint Conference on Neural Networks*, vol. 1, pp. 55-59, Seattle, 1991.

5. Gedeon, TD, & Harris, D "Progressive Image Compression," *Proceedings International Joint Conference on Neural Networks*, vol. 4, pp. 403-407, Baltimore, 1992.

6. Gedeon, TD, Ramer, A, Padet, C & J "Abstracting uncertain knowledge: Case for neural nets application," *International Journal on Intelligent Automation and Soft Computing*, 10 pages, (in press), 1995.

7. Sietsma, J, & Dow, RF, "Neural net pruning - why and how," *IJCNN*, vol. 1, pp. 325-333, 1988.

8. Mozer, MC, Smolenski, P, "Using relevance to reduce network size automatically,", *Connection Science*, vol. 1, pp. 3-16, 1989.

9. Segee, BE, & Carter, MJ, "Fault Tolerance of Pruned Multilayer Networks," *IJCNN*, vol. 2, pp. 447-452, Seattle, 1991.

10. Rumelhart, DE, "Learning and Generalisation," *Proc. IEEE Int. Conf. on Neural Networks*, San Diego, 1988.

11. Sanger, D, "Contribution analysis: a technique for assigning responsibilities to hidden units in connectionist networks", *Connection Science*, vol. 1, pp. 115-138, 1989.

12. Karnin, ED, "A simple procedure for pruning back-propagation trained neural networks," *IEEE Transactions on Neural Networks*, vol. 1, pp. 239-242, 1990.

13. Hagiwara, M, "Novel back propagation algorithm for reduction of hidden units and acceleration of convergence using artificial selection," *IJCNN*, vol. 1, pp. 625-630, 1990.

14. Gedeon, TD, Harris, D, "Network Reduction Techniques," *Proc. Int. Conf. on Neural Networks Methodologies and Applications*, AMSE, vol. 1, pp. 119-126, San Diego, 1991.

15. Fleming, MK & Cottrell, W, "Categorisation of Faces Using Unsupervised Feature Extraction," *IJCNN*, vol. 2, pp. 65-70, 1990.

16. Gedeon, TD, Harris, D, "Creating Robust Networks," *IJCNN*, pp. 2553-2557, Singapore, 1991.

17. Wong, PM, Gedeon, TD and Taggart, IJ "An Improved Technique in Porosity Prediction: A Neural Network Approach," *IEEE Transactions on Geoscience and Remote Sensing*, (in press) 1995.