

Fault Tolerance of Pruned Multilayer Networks

Bruce E. Segee
Michael J. Carter

Intelligent Structures Group
Department of Electrical and Computer Engineering
University of New Hampshire
Durham, NH 03824

Abstract

Techniques for dynamically reducing the size of a neural network during learning have been found by some investigators to speed learning convergence and improve network generalization. However, legitimate concern arises about the fault sensitivity of the pruned network relative to that of its parent. Work has been done to assess the tolerance of multilayer feedforward networks to the zeroing of individual weights, and to determine if network pruning during learning affects this tolerance. Multilayer networks having a single input and a single output were trained to produce the sine of the input value on the interval $[-\pi, \pi]$. Identical networks with identical initial weights were then trained using the skeletonization technique described by Mozer and Smolensky [Mozer 1989]. Each weight in these networks was zeroed in turn, and the effect on the RMS approximation error was noted. Surprisingly, the unpruned networks, which had considerably more free parameters, were found to be no more tolerant to weight zeroing than the pruned networks. Additionally, maintaining a separate relevance estimate for each node was found to be unnecessary as weight magnitudes associated with a node proved to be as reliable in estimating the relevance of the node.

1.0 Introduction

Multilayer feedforward networks are in very common usage in the field of neural networks. They are well suited to a variety of tasks, but have the drawback that they tend to learn very slowly, and thus require a great deal of training. In an effort to speed training, a technique which has shown considerable promise involves beginning with a network larger than needed, and as training proceeds, one gradually reduces the size of the network by removing nodes which do not contribute significantly. The method by which the size of the network is reduced is called pruning (or skeletonization). There are a variety of techniques for determining when and where to prune such as weight-elimination [Rumelhart 1988] and optimal brain damage [LeCun 1990].

The motivation for the work described in this paper was to assess the fault tolerance of multilayer networks trained using pruning and compare this to the fault tolerance of multilayer networks of the same original size and the same initial weights trained without pruning.

The goals of the research were twofold. The initial goal was to characterize the fault tolerance of a network to the zeroing of a single weight and to characterize the tolerance of a pruned network to the zeroing of a single weight with respect to an initially identical network. Secondly, it was desired to test whether Mozer and Smolensky's method of estimating the relevance of a node (discussed in Section 3.0) was an accurate predictor of the relevance of a weight associated with that node. Specifically, for a given size weight, does its association with a high-relevance node increase its effect on the output, relative to that of the same weight associated with a low relevance node?

2.0 Network Structure

The networks used for the experiments described in this paper are single-input, single-output, multilayer feedforward networks with sigmoidal activation functions. Each node in the network calculates the sigmoidal function of the weighted sum of its inputs and an adjustable bias term. The weights and biases are modified using backpropagation as described in [Rumelhart, 1986]. The error measure used in training is the sum squared error over the training set.

Networks having one, two, and three hidden layers were examined. The number of nodes in each layer was also varied. The networks were chosen to have between 150 and 250 weights.

3.0 Relevance measure and node pruning

Mozer and Smolensky [Mozer, 1989] defined the relevance of a node as the difference between the output error without the node and the output error with the node. Mathematically:

$$\rho_i = E_{\text{without unit } i} - E_{\text{with unit } i}$$

where ρ_i is the relevance of node i , and E represents the output error, or more generally, the network performance measure.

Although it is not difficult to measure the relevance directly, it does require a pass through the training data for each node. Mozer and Smolensky introduced a means of estimating the relevance of a node in a multilayer network during training in a manner very similar to standard backpropagation.

Mozer and Smolensky defined a parameter α for each node, which represents the attentional strength of the unit. The value of α may vary from zero to one. The α value may be thought of as gating the flow of activity from the unit. An α value of one would correspond to a normal unit, while an α value of zero would correspond to the unit having been removed from the network. Thus, the relevance becomes:

$$\rho_i = E_{\alpha_i=0} - E_{\alpha_i=1}$$

Mozer and Smolensky then made the gross approximation that the quantity $E_{\alpha_i=0} - E_{\alpha_i=1}$ is approximately equal to the value $\frac{-\partial E}{\partial \alpha_i}$. While this may seem rather crude, it did permit the use of a backpropagation-like procedure for estimating node relevance. Thus:

$$\rho_i \cong - \frac{\partial E}{\partial \alpha_i} = \hat{\rho}_i$$

The relevance estimate $\hat{\rho}$ was found to fluctuate strongly as training progressed. Thus an exponentially decaying time average of the derivative was used by Mozer and Smolensky as well as in the experiments described in this paper. The formula used is given below:

$$\hat{\rho}_i(t+1) = 0.8 \hat{\rho}_i(t) + 0.2 \frac{\partial E(t)}{\partial \alpha_i}$$

The error value used to estimate the relevance is an absolute error function (Mozer and Smolensky call it a linear error function) rather than the quadratic function used during backpropagation. The absolute error function is defined as follows:

$$E^1 = \sum_{i = \text{all inputs}} |\text{target } i - \text{output } i|$$

This error measure was used for estimating relevance, while the quadratic error measure was used for backpropagation. The quadratic error function is:

$$E^q = \sum_{i = \text{all inputs}} (\text{target}_i - \text{output}_i)^2$$

The linear error measure was chosen by Mozer and Smolensky in order to avoid the situation in which the relevance estimate of a node that is close to its target activity tends toward zero. In this situation, the derivative of the quadratic error goes to zero, while the derivative of the linear error function does not.

4.0 Experiment definition

A software package was developed which simulated a single-input, single-output, multilayer feedforward network having Mozer and Smolensky's relevance estimate incorporated. During operation, the network was trained using standard backpropagation to produce the sine value of inputs, sampled on the interval $[-\pi, \pi]$. Pruning could be enabled or disabled. When pruning was enabled, every 500 passes through the training set the least relevant node was removed.

The networks have been trained to produce the sine value of the input with an RMS error in the range $[0.01, 0.02]$. The RMS error is defined as follows:

$$\text{RMS error} = \sqrt{\frac{1}{N} \sum_{i=0}^N (\text{target}_i - \text{output}_i)^2}$$

where N is the number of training points. For the experiments described in this paper N was chosen to be 50.

After training was complete, weights within the networks were zeroed one at a time, and the change in RMS error was observed. No training was done after zeroing a weight and only a single weight was zeroed at a time. The weight value was restored prior to the next weight being zeroed.

Figures 1 through 4 are representative of the results obtained. The salient features of these figures are discussed below.

5.0 Experimental results/discussion

Figure 1 shows a typical plot of the change in the RMS approximation error caused by zeroing each of the weights in a network in turn. In general the magnitude of the weights associated with a node is an excellent predictor of the relevance of the node. While it is conceivable that there could be a node with low relevance having large weight values associated with it, such a node has never been found in any of our experiments.

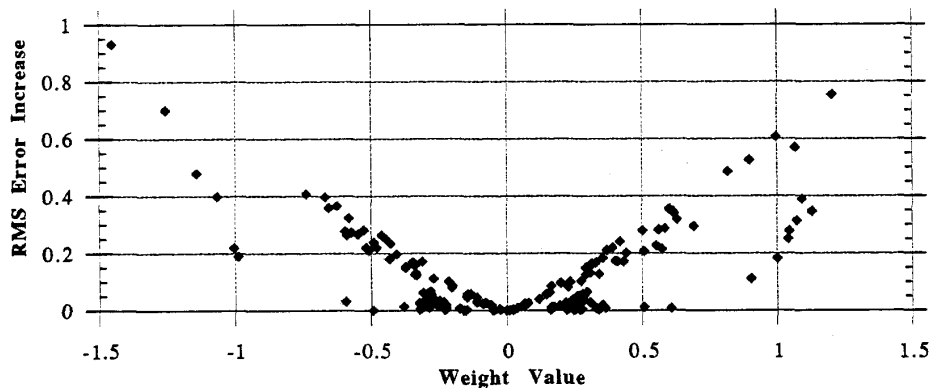


Figure 1: A typical plot of RMS approximation error increase from zeroing individual weights as a function of weight value

Figure 2 shows the RMS approximation error increase versus weight magnitude for a network trained using pruning. The networks shown in Figures 1 and 2 were initialized identically, i.e., at the start of training they had the same number of nodes in the same configuration and identical small random weights and biases. Both networks were trained for 30000 passes through the training set containing 50 equally spaced input values in the range $[-\pi, \pi]$. The network associated with Figure 1 was trained using standard backpropagation without momentum. The network associated with Figure 2 was trained using standard backpropagation with pruning. The least relevant node was pruned from the network after every 500 training passes.

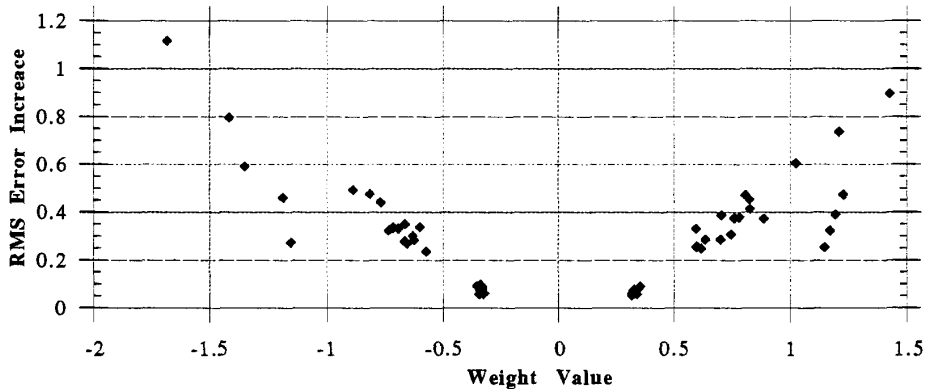


Figure 2: A typical plot of RMS approximation error increase from zeroing individual weights as a function of weight value in a network trained using relevance-based node pruning. Note that the network was initially identical to the network depicted in Figure 1.

It is most interesting to compare the plots in Figure 1 and Figure 2. The similarity is striking. In fact, the effect that pruning seems to have is to remove all small weights. This stands to reason since there is such a strong correlation between node relevance and weight magnitude (discussed in Section 5.2). In the pruned network, no weights smaller in magnitude than about 0.3 survived pruning. Of the weights remaining, there is essentially a one-to-one correspondence with weights in the unpruned network. In the pruned network, the weight magnitudes and corresponding RMS approximation error increases are slightly higher, but not significantly so. The fact that the significant weights of the two networks should be so similar is wholly remarkable. Since pruning makes significant changes in the network, and since the pruned and unpruned networks are so similar after training is complete, this suggests that the nodes which contribute significantly to the function approximation are determined when the network is initialized.

5.1 Fault Tolerance Considerations

Figure 1 shows quite clearly that a multilayer network is not inherently tolerant to the zeroing of a single weight, even when there are a large number of weights present. Recall that the function being learned was a sine wave, and thus has an RMS value of 0.707. An error which is a sizable percentage of this is a very large error. However, as is shown in Figure 1, there are several weights which, when any one of them is zeroed cause the RMS approximation error increase to be at least 0.707. The network represented in Figure 1 had 180 weights, and yet, the loss of a single weight could cause the RMS approximation error to exceed the RMS value of the function being approximated. This lack of fault tolerance finding is consistent with findings in locally generalizing networks like CMAC [Carter, 1990].

The network represented in Figure 2 is also not very fault tolerant for precisely the same reasons. The interesting thing to note, however, is that it has only 62 weights, or roughly one third of the weights of the network represented in Figure 1. Despite having almost two thirds

fewer weights the network is not significantly less fault tolerant, since there are approximately as many critical weights in both networks.

5.2 Examining the Relevance Estimate

There was found to be a strong correspondence between the magnitude of a weight value associated with a node and the relevance estimate of that node. Figure 3 shows a typical graph of weight vs. relevance for an unpruned network. The data for this graph came from the weights between the two hidden layers of a network containing two hidden layers, one containing 20 nodes and the other containing 11 nodes. Virtually without exception, graphs of weight value versus node relevance estimate on any layer of weights between two hidden layers look like this graph (qualitatively). Notice that as node relevance increases, the spread of weight values associated with that node also increases. This suggests several possible relationships between a weight value and the relevance of a node. For instance, relevance may be predicted by the largest weight value associated with a given node. The variance of the weights associated with a given node is also a predictor of node relevance. When a node has only a single weight (as from the last hidden layer to the output layer), then the relevance estimate is directly proportional to the weight magnitude.

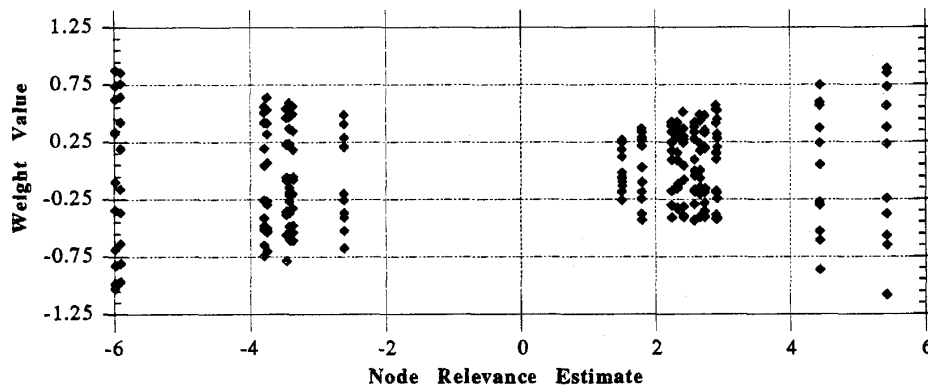


Figure 3: A plot showing typical weight values associated with a node as a function of nodal relevance

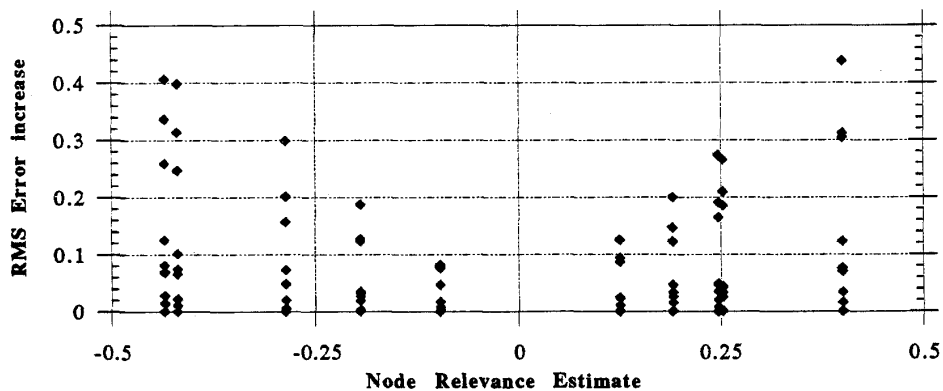


Figure 4: A typical plot of RMS approximation error increase as individual weights are zeroed as a function of the relevance estimate of the node

Figure 4 shows a typical plot of the increase in the RMS approximation error vs. relevance estimate of a node as each weight associated with a node is zeroed in turn. Generally, the largest RMS approximation error increases come from weights associated with nodes which have a high (absolute value) relevance estimate. However, for a given node there is a wide range of values for RMS approximation error increase, i.e., even for the most relevant nodes, there are weights which have virtually no effect on the RMS approximation error when they are zeroed. In general, however, the relevance of a node proved to be a reasonable predictor of the change in error caused by deletion of the maximum magnitude weight associated with the node. Recall that the relevance measure was created to estimate the impact of eliminating all of the weights associated with a given node.

6.0 Conclusions:

Contrary to the belief widely held, multilayer networks are not inherently fault tolerant. In fact, the loss of a single weight is frequently sufficient to completely disrupt a learned function approximation. Furthermore, having a large number of weights does not seem to improve fault tolerance.

The skeletonization algorithm was found by Mozer and Smolensky to have a number of useful properties. Interestingly, this algorithm does not seem to significantly impact the tolerance of the network to the zeroing of a single weight.

The relevance estimate used by Mozer and Smolensky does indeed correlate well with the true relevance of a node. However, it appears that in the case of backpropagation, an estimate based on the magnitude of the weights associated with a node would prove to be an accurate predictor as well, without requiring an extra pass through the network during each training iteration to update the relevance estimates. While it is possible for a low relevance node to have large weights, this does not seem to occur in practice.

Because of the strong correlation between weight magnitude and node relevance, it is very likely that pruning procedures such as Rumelhart's weight-elimination [Rumelhart 1988] which are based on weight magnitude also would not significantly affect the fault tolerance of a network.

Bibliography

Mozer, M. C. and Smolensky, P. "Skeletonization: A Technique for Trimming the Fat from a Network via Relevance Assessment", Department of Computer Science & Institute of Cognitive Science, University of Colorado, Boulder, Technical report CU-CS-421-89, January, 1989

D.E. Rumelhart, "Learning and Generalization", *Proc. IEEE Int. Conf. Neural Networks*, San Diego, 1988 (plenary address)

Y. LeCun, J.S. Denker, and S. A. Solla, "Optimal Brain Damage", *Advances in Neural Information Processing Systems 2*, D. S. Touretzky (Ed.), San Mateo, CA: Morgan-Kaufmann Publishing Co, 1990

D.E. Rumelhart and J.L. McClelland (1986), editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Volume 1, Chapter 8, The MIT Press, Cambridge

Carter, M.J., F.J. Rudolph, A.J. Nucci, "Operational Fault Tolerance of CMAC Networks", appears in *Advances in Neural Information Processing Systems 2*, D.S. Touretzky (Ed.), San Mateo, CA: Morgan-Kaufmann Publishing Co, 1990