Vectorized Backpropagation and Automatic Pruning for MLP Network Optimization

Suryan Stalin and T. V. Sreenivas
Department of Electrical Communication Engineering
Indian Institute of Science, Bangalore-560012, India

Abstract– In complicated tasks such as speech recognition, neural network architectures have to be improved for better learning and recognition performance. This paper presents an analysis of the backpropagation algorithm and reveals the significance of *vectorized backpropagation and automatic pruning* for better learning performance and MLP network optimization. During the learning phase, the network which uses vectorized backpropagation converges within 20% - 50% of the iterations required for the standard MLP to converge without affecting the test set performance. The network pruning algorithm reduces the number of hidden nodes and connection weights. The pruned network with only 40% connection weights of the unpruned network gives the same learning and recognition performance as the parent unpruned fully connected network.

I. Vectorized Backpropagation

The multilayer perceptron network (MLP) using standard backpropagation (BP) algorithm[1] minimizes output mean square error by modifying the weights and thresholds after each input pattern is introduced to the network. The learning procedure can be expressed as

$$w_{uv} \leftarrow w_{uv} - \eta \frac{\partial E_p}{\partial w_{uv}}, \quad (1)$$

$$RHS = w_{uv} - 2\eta \sum_{j=1}^{N} e_{jp} \frac{\partial e_{jp}}{\partial w_{uv}}, \quad (2)$$

where w_{uv} is the weight from node u to node v, η is the rate of learning, e_{jp} is the j^{th} dimensional output error for the input pattern p and $E_p = \sum_{j=1}^{N} e_{jp}^2$ is the output mean square error for the pattern p.

Consider the pattern classification to be an N-class problem and let us choose the desired class identification vectors to be the N orthonormal vectors of the output space. In the N dimensional output space, $\mathbf{E}_p = [e_{1p}, ..., e_{Np}]^t$ is the output error vector for the pattern p. Let $\mathbf{E}_p^k$ and $\mathbf{E}_p^{k+1}$ be the error vectors before and after the weight updation for the input pattern p presented during the k^{th} scan of the training set and let the j^{th} component of the vector be e_{jp}^k and e_{jp}^{k+1} respectively. In the MLP network learning, the backpropagation algorithm uses the gradient descent approach which assures that

$$| \mathbf{E}_p^{k+1} | \leq | \mathbf{E}_p^k | \quad (3)$$

However, this condition does <u>not</u> assure reduction of the individual components of the error vector. i.e.,

$$| e_{jp}^{k+1} | \not\leq | e_{jp}^k | \qquad for\ some\ j\ and\ k. \quad (4)$$

Thus, it is possible that the network may have unlearned for a particular class, while the average learning for all classes put together has improved with the training iterations. Fig. 1 shows the variation of $\mathbf{E}_p$ during the learning process. The output error vector after an update has the freedom of lying anywhere within the hyper-sphere of radius equal to the magnitude of error vector before the update. Thus, if we can formulate the learning rule such that all of the components e_{jp}^k are simultaneously reduced through each iteration, faster convergence of the neural network could be expected. This corresponds to a constrained minimization in which the updated error vector can lie only within the N-dimensional cuboid formed by the error vector before the updation as shown in Fig. 1.

This formulation of the learning rule in which each of the individual components of the error vector $\mathbf{E}_p$

0-7803-0999-5/93/$03.00 ©1993 IEEE

are minimized through each iteration of learning is termed as "vectorized backpropagation(VBP)."

A. Multiplane MLP Network

The goal of minimizing the error components e_{jp} individually can be achieved if the network has a single output node. Thus, the multi-class pattern recognition problem has to be viewed as several two-class recognition problems. This can be achieved by modifying the learning rule (2) which results in the new multiplane MLP (MPMLP) architecture.

Consider the BP learning rule(2) where the connection weight w_{uv} is altered by the gradient of the error vector components. i.e.,

$$w_{uv}^{j} \Leftarrow w_{uv} - 2\eta\nabla_j \tag{5}$$

Now, let us formulate a new network with one single out put node corresponding to the j^{th} class and a stack of such networks for the N different classes. Each weight w_{uv} of the standard MLP network in learning rule (2) is extended as a vector $\mathbf{W}_{uv} = [w_{uv}^1, ..., w_{uv}^N]^t$, where w_{uv}^j is the weight in the j^{th} stack. Each of the networks in the stack receive identical inputs and can be learnt independent of others in the stack. Thus, learning rule for the j^{th} stack can be written as

$$w_{uv}^{j} \Leftarrow w_{uv}^{j} - 2\eta e_{jp}\frac{\partial e_{jp}}{\partial w_{uv}^{j}} \tag{6}$$

The learning rule (6) provides a joint minimization of the individual error components e_{jp} than the minimization of the total error $\mid \mathbf{E}_p \mid$ as in the standard BP. i.e.,

$$\mid e_{jp}^{k+1} \mid \leq \mid e_{jp}^{k} \mid \qquad \forall\ j,k. \tag{7}$$

The new architecture of a stack of MLPs, referred to as multiplane MLP, is shown in Fig.2.

The choice of orthonormal target vectors to identify each class makes each plane of the MPMLP to solve a 2-class recognition problem with a single output node. Since each output space orthonormal vector of the MLP is representing a separate class, j^{th} plane of the MPMLP network learns to recognize j^{th} class independent of the learning in other planes. This is because the output error is propagated back only within each plane and there is no link across different planes.

II. Network Optimization

Vectorized backpropagation is useful for better convergence of the network. However, the realization through MPMLP increases the number of network weights. This can result in memorization effects and suboptimum local minima in learning due to the limited training data [3]. On the other hand, if the network size is too small, the network will fail to converge in learning. Since each plane of the MPMLP network is dealing with a different recognition task, the optimum size for each of the planes will be different. The Network pruning is a method in which a network of larger size is chosen and then iteratively reduced to an optimum size. Such Pruning accomplished during the learning process has been referred to as "dynamic pruning [4]." Pruning has also been performed after the learning process is complete [3]. However, in both methods, the pruning is limited to the reduction of network nodes only. Pruning of nodes leads to a drastic decision because many connections go through a node. Instead of such a fully connected MLP, wherein each node in a layer is connected to all nodes in the adjoining layers, reduced number of interconnections can be advantageous in some cases [6].

An automatic pruning algorithm [2] has been developed which is applicable to a general MLP network. In this method, pruning is incorporated into the learning algorithm and the network is iteratively reduced such that an optimum configuration is achieved along with the network convergence. Using an integrated measure of *forward and backward significance,* least effective connection weights are identified and removed.

A. Pruning Algorithm

Let C_m be the number of nodes in the m^{th} layer of an MLP network where $m = 0$ is the input pattern layer and $m = 3$ is the final layer. Let the output at node i in the $(m-1)^{th}$ layer be x_i and the weight connecting node i to node j in the m^{th} layer is w_{ij}^m. For each pattern **p** introduced to the network during learning, the backpropagation algorithm makes use of each connection weight in the network in two instances. In the forward direction, for final layer output calculations, each connection weight w_{ij}^m con-

tributes $x_i w_{ij}^m$ to the total activation $\sum_i x_i w_{ij}^m$ of the node to which it is connected. In backward error propagation, each weight w_{jk}^m contributes $w_{jk}^m \delta_k^m$ to the total correction factor $\sum_k w_{jk}^m \delta_k^m$ for the weight w_{ij}^{m-1} in the previous layer as shown in (8).

$$w_{ij}^{m-1} \Leftarrow w_{ij}^{m-1} - \eta x_i \sum_k w_{jk}^m \delta_k^m \qquad (8)$$

where

$$\delta_k^m = -(d_k - y_k) y_k (1 - y_k) \; for\ the\ final\ layer \qquad (9)$$

$$\delta_j^m = \sum_k w_{jk}^{m+1} \delta_k^{m+1} \; for\ other\ layers \qquad (10)$$

In (9), d_k and y_k represent the target output and the observed output at node k in the final layer. If both forward and backward contributions from a particular weight are not significant over the entire training set, it is clear that the weight is not playing a significant role in the learning process and such a link can be marked for pruning. However, changing the network structure too soon can affect the convergence of the learning process.

To determine the optimum degree of pruning, a parameter called *pruning factor* (f_p) is introduced ($0 < f_p < 1$). A connection weight w_{ij}^m is pruned if the following conditions are satisfied.

$$| x_i w_{ij}^m | < | \sum_i x_i w_{ij}^m | f_p \; for\ all\ \mathbf{p} \qquad (11)$$

$$| w_{ij}^m \delta_j | < | \sum_j w_{ij}^m \delta_j | f_p \; for\ all\ \mathbf{p} \qquad (12)$$

$f_p = 0$ implies learning without pruning and $f_p = 1$ means pruning to the maximum extent. Accordingly, by varying f_p the degree of pruning can be varied. If all weights attached to a node are pruned, it is equivalent to the removal of that node. Usually f_p is chosen close to 0. Elimination of the connection weights can cause convergence problems to the backpropagation algorithm, but care has been taken through (11) and (12) that the error caused by pruning is minimal.

B. Decision regions

Each perceptron in the standard MLP network with input vector X and weight vector W partitions its input space into 2 regions characterized by $XW^t > \theta$ and $XW^t < \theta$ with a hyperplane given by $XW^t = \theta$ where, θ is the perceptron threshold [1]. Thus, C_1 perceptrons in layer-1 form in general C_1 hyper planes in the feature space [5]. These can partition the space into a maximum of 2^{C_1} decision regions (Fig. 3(b)). Input patterns occurring in different regions give rise to a different layer-1 output pattern with each node output being in the range of 0 to 1. The layer-1 outputs form an C_1 dimensional space where in the range is limited to an C_1 *dimensional hyper cube.* Thus, different decision regions in the input feature space are mapped to corresponding zones in the hyper cube (Fig. 3(c)). For perceptrons with hard limiting transfer functions (binary output), decision regions in the feature space are mapped to the corners of the cube. For sigmoidal perceptrons, the decision regions are mapped to the neighborhood of different corners. Removal of a node from layer-1 eliminates the corresponding hyper plane. This makes the decision region to loose one of its boundary and starts spreading in volume. These principles can be similarly extended to layer-2 and layer-3. Layer-2 forms decision regions by hyper planes in the space derived from layer-1. Node removal in layer-2 makes decision region to spread in layer-1. Thus, in general, pruning should be applied only until the decision region for a pattern class starts to overlap with another.

In Fig. 3(a), the decision boundary in the feature space corresponding to the first node in layer-1 is given by

$$XW_1^t = \theta_1 \qquad (13)$$

$$i.e., \quad x_1 w_{11} + x_2 w_{21} = \theta_1 \qquad (14)$$

Pruning the link w_{21} modifies the boundary to

$$x_1 = \frac{\theta_1}{w_{11}} \qquad (15)$$

which represents a line parallel to the x_2-axis. This means that the perceptron decision is independent of the parameter x_2. Generalizing this fact for an N_m dimensional layer, *pruning the link w_{ij}^m modifies the hyper plane, in the space derived by $(m-1)^{th}$ layer, parallel to the axis along the i^{th} dimension.* Also, output of node j in the m^{th} layer becomes independent of its i^{th} dimensional input.

III. Experimental results

MLP network's ability to tolerate distortion in the input patterns and yet classify them correctly depends on how well the network has learnt from the training set and this is referred to as network generality. The test set performance, defined as the proportion of the number of untrained patterns classified correctly by the network to the total number of all possible input patterns is used as a measure of network generalization. This is measure applicable to discrete pattern finite extent problems.

The learning as well as test set performance of the MPMLP is evaluated using a 3 class binary picture recognition problem and results are compared with that of an MLP. Binary pictures made of 3×3 and 5×5 grids (Fig.4) representing characters I, O and X are used for training. For the 5×5 grid experiment an 18-18-3 node fully connected architecture is used for the MLP network where as in MPMLP, the same number of nodes are distributed into 3 planes. For an n node output MLP network, the maximum possible mean square error per pattern is n. A value of 0.01% of the maximum possible error is taken as the threshold for convergence. The test set comprised of the $(2^9 - 3)$ untrained patterns for the 3×3 grid experiment and $(2^{25} - 3)$ for 5×5 grid experiment. The MPMLP, with the same number of total nodes as the MLP, converges within 20% - 50% of the iterations required for the MLP to converge (Fig.5(a) and 5(b)) without any deterioration in the testing performance. The test set performance of the MPMLP is found marginally better than that of the MLP network.

The addition of the pruning algorithm to the MPMLP network learning is evaluated in comparison with the parent unpruned MPMLP network which is trained with the *same random initialization* as the pruned network. Also the pruned MPMLP performance is compared with that of the pruned MLP network.

Weight reduction is measured using a reduction coefficient(ρ) given by

$$\rho = \frac{\sum_{m=1}^{m=3} \acute{C}_m}{\sum_{m=1}^{m=3} C_m C_{m-1}} \tag{16}$$

where $\acute{C}_m$ is the number of weights remaining unpruned between the m^{th} and $(m-1)^{th}$ layer. Thus the reduction coefficient for the m^{th} layer is given by

$$\rho_m = \frac{\acute{C}_m}{C_m C_{m-1}} \tag{17}$$

In all the experiments, for a fixed f_p, layer-1 weights have been subjected to maximum pruning followed by progressively less pruning for succeeding layers ($\rho_1 < \rho_2 < \rho_3$).

The reduction in the number of connection weights and the test set performance with respect to f_p is shown in Fig.6(a) and 6(b) for both MLP and MPMLP networks. Fig. 6(a) shows a greater reduction in connection weights of MPMLP than MLP. This justifies the effectiveness of pruning for MPMLP. As Fig. 6(b) shows, with the pruning factor in the range of 0-0.4, the network size is reduced with least deterioration, if not marginal improvement, in the testing performance. For f_p close to unity the network size becomes too small to maintain network generality.

Also, unless f_p is close to unity, pruning does not seem to affect the number of learning iterations for convergence. Thus, incorporating pruning while learning in an MPMLP architecture leads to an optimized architecture which can provide better performance.

References

[1] B. Widrow and M. A. Lehr, "30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation," IEEE Proceedings, Vol 78, No. 9, Sept. 1990.

[2] Suryan Stalin and T. V. Sreenivas, "Vectorised Backpropagation and Automatic Pruning for MLP Network optimization," Technical report , Dept. of ECE, Indian Institute of Science, Aug 1992.

[3] J. Sietsma and R.J.F. Dow, "Neural net pruning - Why and how," Proc. IEEE, Int.Joint Conf. Neural Networks, Vol 1, pp 325-333, 1988.

[4] B. E. Segee and M. J. Carter, "Fault tolerance of multilayer networks," Proc. IEEE, Int.Joint Conf. Neural Networks, Vol 2, pp 447-452, 1991.

[5] J. Makhoul, A. El-Jaroudi, and R. Schwartz, "Partitioning Capabilities of Two-Layer Neural Networks," IEEE Trans. on Signal Processing, Vol 1, No 6, pp 1435-440, June 1991.

[6] T. V. Sreenivas, Unnikrishnan, V. S. and D. N. Dutt, "Pruned Neural Network for Artifact Reduction in EEG Signal," IEEE, Int. Conf. Engineering and Medicine in Biology (EMBS), Florida, 1991.

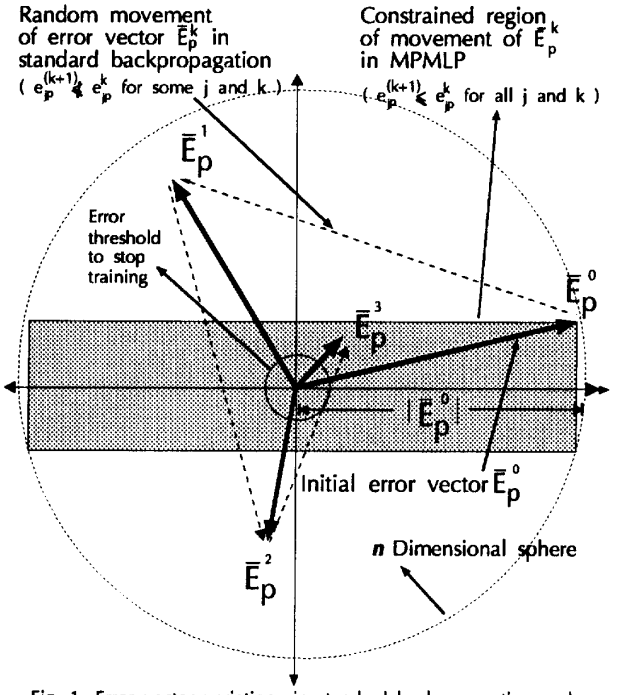

Fig. 1. Error vector variations in standard backpropagation and vectorized backpropagation.

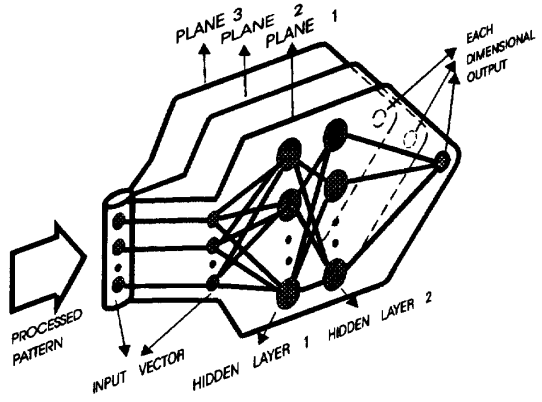

Fig. 2. Multiplane MLP network for N=3 class problem. The network in each plane provides recognition of a single class.

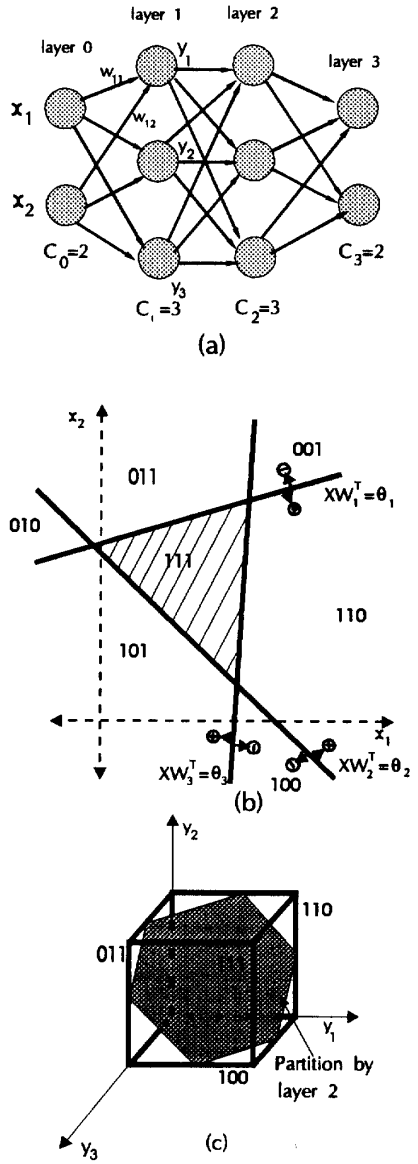

Fig. 3. (a) An MLP network with two dimensional input space. (b) Decision regions formed by layer 1. (c) N dimensional hyper cuboid in the layer 1 output space.

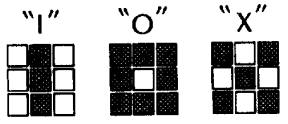

Fig. 4. 3x3 grid binary patterns representing characters I, O and X.

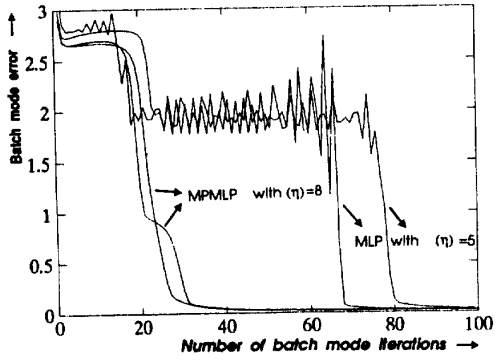

Fig. 5(a). Error convergence of MLP and MPMLP networks during the training phase of a 3 class 3x3 grid binary picture recognition problem. The best learning coefficients: η_{MLP}=5; η_{MPMLP}=8.

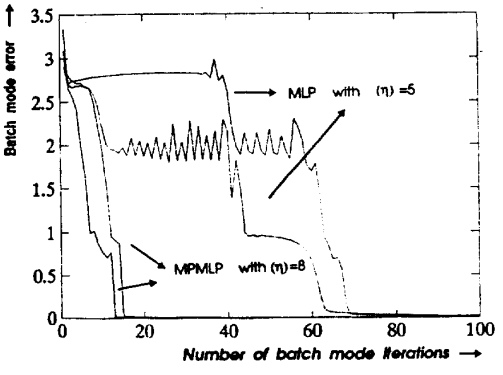

Fig. 5(b). Error convergence of MLP and MPMLP networks during the training phase of a 3 class 5x5 grid binary picture recognition problem. The best learning coefficients: η_{MLP}=5; η_{MPMLP}=8.

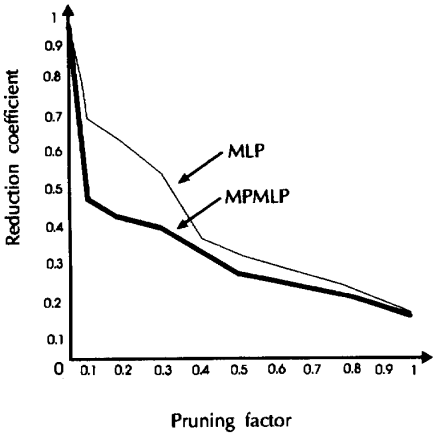

Fig. 6(a). Variation of reduction coefficient (ρ) with respect to pruning factor (f_p) for the MLP and MPMLP networks.

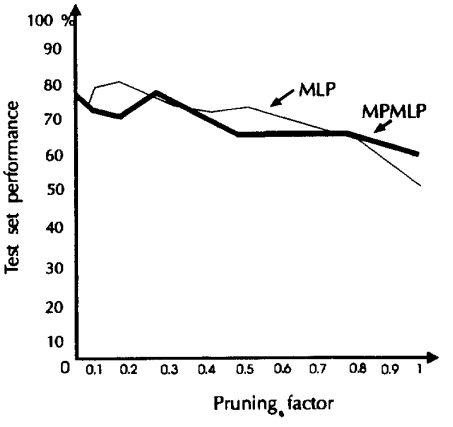

Fig. 6(b). Variation of reduction coefficient (ρ) with respect to pruning factor (f_p) for the MLP and MPMLP networks.