**MVA'90** IAPR Workshop on Machine Vision Applications Nov. 28-30,1990, Tokyo

# Improving the Performance of Multilayer Perceptron through Empirical Maximum Likelihood (EML) Learning Rule

S.Semnani and M.J.J.Holt
Department of Electronic and Electrical Engineering
Loughborough University of Technology

Loughborough, Leics. LE11 3TU
England.

## Abstract

This paper reports on a probabilistic learning method that can be used to train a multilayer perceptron (network). The method can be viewed as a variation on the popular learning rule of backpropagation (BP). The alternative method is based on the criterion of empirical maximum likelihood (EML) rather than the sum-of-squares cost function used in the conventional BP. In tests where simple networks are applied to experimental data, the EML algorithm converges to a near optimum solution, without increasing the computational complexity. Moreover the number of iterations required for convergence is often found to decrease. The results also illustrate that the error landscape produced by EML is simpler than BP. Hence this leads to a more efficient learning rule for multilayer networks.

## Introduction

Neural networks and connectionist models of computation are experiencing a renewed popularity. Algorithms which allow a network of neurons to learn input-output relationships from examples have been recently developed and successfully applied to various difficult problems[1].

In layered neural networks, a widely used learning algorithm is the backpropagation method, reported by Rumelhart et al[2]. This is among the most versatile and effective training methods reported to date.

However, it is generally acknowledged that BP suffers from slow convergence and a tendency to get trapped in local minima. Existing solutions to the problem involve either increasing the number of hidden nodes[3] or introducing a random component to the search algorithm[4]. Both approaches result in an increase in the computation involved in the learning process.

In this paper an alternative approach, based on a reformulation of BP method in terms of empirical maximum likelihood criterion, is presented. This is formulated by assuming the network outputs as probabilistic values[5]. It will be shown that this approach reduces the convergence rate without increasing the computational complexity, leads to more efficient generalisation capability, and produces a simpler error landscape with comparision to the conventional BP.

In the following a short summary of the BP algorithm is given. The EML formulation of the algorithm is then stated, and it is shown to reduce the number of calculation per iteration. We then present the results of simulations comparing the conventional BP and its EML formulations, in the training of neural networks with experimental data arising from template matching.

## The Backpropagation Algorithm

The general idea of BP algorithm is to make appropriate adjustments to the network weights in the direction of negative gradient, such that the overall error of the network is minimised. This error is most commonly formulated as the sum of squares of the errors between outputs from the network and desired outputs used for training, and is given by :

$$E = \frac{1}{2} \sum_{c} \sum_{j} (d_{jc} - y_{jc})^2 \qquad (1)$$

where the summation is over all training cases c and all output nodes j, $d_{jc}$ is the desired output (usually coded as 0 or 1).

In the following formulae, we have adopted a similar notation to that used in ref.2, where a single pattern presentation is considered, both the inputs to and outputs from a node are described by $y_j$, and are distinguished only by the value of the subscript.

In any given node, the output $y_j$ is related to the inputs $\{y_i\}$ by:

$$x_j = \sum_{i} y_i w_{ij} \qquad y_j = \frac{1}{1 + e^{-x_j}} \qquad (2)$$

The network weights $w_{ij}$ are optimised by an iterative procedure in which the adjustments to the weights at the $n^{th}$ iteration is given by:

$$\Delta w_{ij}(n) = \eta \delta_j y_j + \alpha \Delta w_{ij}(n-1) \qquad (3)$$

where $\eta$ is the learning constant, $\alpha$ is the momentum term, and $\delta_j$ is an error derivative associated with a node j after a pattern presentation. For an output node $\delta_j$ is given by[2]:

$$\delta_j = (d_j - y_j)\, y_j (1 - y_j) \qquad (4)$$

and for a hidden node $\delta_j$ is:

$$\delta_j = y_j(1 - y_j)\sum_k \delta_k w_{jk} \qquad (5)$$

where the summation k is over the number of connections to the nodes in the next layer. The above is only a summary of the BP algorithm, presented here to set the stage for the discussion of the EML learning rule. However detailed analysis of the BP method is presented in Rumelhart et al[2].

### Empirical Maximum Likelihood

The word empirical is used here since this approach does not assume a prior form of distribution for each class of data (as is the case for a maximum likelihood Gaussian classifier, for example). In the EML learning method, the values of input and output vectors are defined as probabilities, thus instead of minimising an error function, the EML learning rule attempts to maximise the likelihood or probability of the training sequence.

For a given pattern, $y_j$ can be assumed to be the probability that a pattern will belong to class j, then the probability of all the training data being correct will be:

$$P = \prod_c \prod_j \left\{ d_{jc} y_{jc} + (1 - d_{jc})(1 - y_{jc}) \right\} \qquad (6)$$

The proposed method maximises the log likelihood given by:

$$\begin{aligned} L &= \log P \\ &= \sum_c \sum_j \log \left\{ d_{jc} y_{jc} + (1 - d_{jc})(1 - y_{jc}) \right\} \end{aligned} \qquad (7)$$

Since L is non-positive we negate it to obtain a cost function to be minimised given by:

$$\begin{aligned} E &= -L \\ &= -\sum_c \sum_j \left\{ d_{jc} \log y_{jc} + (1 - d_{jc}) \log(1 - y_{jc}) \right\} \end{aligned} \qquad (8)$$

where the rearrangement with in the summation is possible because $d_{jc}$ is always either 0 or 1.

When the BP algorithm is reformulated in terms of the above cost function , the feed-forward formulae remain the same and it is only the error derivatives for output units that are effected, since they are explicitly depend on the form of the cost function. Hence following the BP formulation, the error derivative $\delta_j$ associated with an output node j, using the EML cost function is calculated as:

$$\delta_j = ( d_j - y_j ) \qquad (9)$$

The remaining formulae in the backpropagation method remain unchanged, but will of course give different values since they use terms evaluated by expression (9). Since it is not necessary to evaluate the cost function in the course of backpropagation learning, the only computational difference is dropping of the term $y_j(1 - y_j)$ from the formula for $\delta$ of an out put node in the conventional BP. The consequences of this modification are three fold:

Firstly, compared with the standard formulation, there is a saving of two floating point multiplications per training sample in each iteration, in the computations associated with the learning process for a two class problem. If there are more than two classes, the number of multiplications saved is multiplied by the number of outputs from the network.

Secondly, the magnitude of error derivatives at the output layer is increased by a factor of at least 4, compared with the conventional BP, so it is appropriate to reduce the learning rate $\eta$ by a similar factor when applying the EML formulation.

Thirdly, the term $y_j(1 - y_j)$ in the conventional BP formulation is effectively a bell-shaped weight function applied to each point's contribution to the gradient, which achieves its maximum for samples in the feature space which are on the decision boundary, and which tails off towards zero as the decision boundary moves away from the sample, even though the sample may become misclassified in the process.

This accounts for a phenomenon often observed in training multilayer networks, that there are large regions of the weight space far removed from the global minimum in which the error function is plateau-shaped. When the weight vectors starts in, or wanders into, such a region, the conventional BP may adjust the weight very slowly, or worse still stop completely because the gradients are so small as to be beyond the numerical range of the computer.

In the EML formulation however, the unweighted gradient $d_j - y_j$ approaches unit magnitude as the decision boundary moves further away from a misclassified sample, and consequently such plateau regions should not occur.

This is illustrated with a simple abstract example. Consider a classification problem in which there is a single feature x and two classes (1 and 2). Let the training set for this problem consist of the following six samples - Class1: x= 0.0, 0.1, 0.3; Class2: x= 0.2, 0.4, 0.5. Let a single layer network for this example consist of a single node, having two weights $w_0$ and $w_1$. Figure 1(a) is a contour plot of the conventional BP cost function against the two weights. The surface consists of a deep ravine, containing the global minimum, and surrounded by plateaus on two sides and a hill at one end. Figure 1(b) shows the EML cost function plotted against the two weights over the same range. Again the global minimum, lies in a deep ravine, but in this case the entire surface is an elongated bowl shape and contains no plateaus within the plotted range. In this example, a gradient descent algorithm should converge relatively quickly to the minimum of the EML function regardless of the starting point, but in the conventional BP algorithm, starting points outside the ravine are liable to extremely slow convergence when the weight vector either starts in, or wanders into, one of the plateau regions.

Similar properties can be expected in the cost functions arising from multilayer networks. With suitably chosen parameters, the EML formulation should not exhibit the same tendency to slow down as a result of plateau regions.

## Performance on Experimental Data

Experimental data were generated from the template matching of binary digital images of text characters at low resolution and under noisy conditions. This problem arises in the compression of document images, and the data used are from documents where conventional template matching has proved unsatisfactory[7].

A feature which is commonly used to detect a match between two characters in the same typeface is the weighted Exclusive-OR (WXOR) pixel error count[8]. This alone is insufficient, and a good distinction is only achievable when other easily measured features of characters, such as dimensions, number of black pixels and topology are also taken into account. In classification terms, this is a two-class problem (match and no match), where there can be up to seven 'hand-crafted' features.

The data used for this test were obtained by measuring four features of pairs of characters extracted from a single document. 50 pairs were used for training, and a further 500 pairs were used for testing the trained classifier. A two-layer network with two hidden nodes was used for training. Larger networks have also been applied to the problem but the improvement has been negligible.

Figure (2) shows the learning and generalisation curve for both the conventional BP and its EML formulation, averaged over 10 random starting points, to remove the dependency on the start-up weights. From the graph it can be seen that, the EML rule converges to a steady state faster than its counterpart. Furthermore the EML rule provides an efficient generalisation capability, and for this problem this has lead to a 3% improvement in the recognition rate on the test data.

## Conclusions

The EML formulation of the backpropagation algorithm for training neural networks has been tested on both abstract and experimental data. The analysis and tests presented here have shown a number of factors in favour of the alternative method:

1. It involves fewer calculations per iteration. In addition to speeding the calculation on a conventional computer, this could also reduce the complexity of hardware implementations of neural networks.
2. It reduces the number of instances where backpropagation fails to locate the global minimum of the cost function.
3. The number of iterations required to reach convergence is often much reduced.
4. The ability of a network to generalise is improved.

The results in this paper are typical of only a small subset of the problems which neural nets are applicable. Nevertheless, we believe that the EML learning rule has many advantages over conventional BP, and strongly deserves further investigation.

## References


1 T.J. Sejnowski and C.R. Rosenberg, 'Parallel networks that learn to pronounce English text'. Complex Systems, Vol. 1, 1987, pp. 145-168.

2 D.E. Rumelhart, G.E. Hinton and R.J. Williams 'Learning representations by backpropagating errors', Nature, Vol.323, 1986, pp. 533-536.

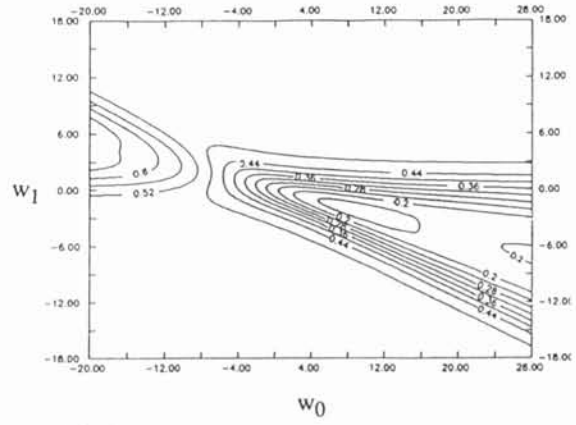
3 R.P. Lippmann, 'An introduction to computing with neural nets', IEEE ASSP Magazine, April 1987, pp. 4-22.

4 R.G. Hoptroff and T.J. Hall, 'Learning by diffusion for multilayer perceptron', Electronics Letters, Vol.25, 1989, pp. 531-533.

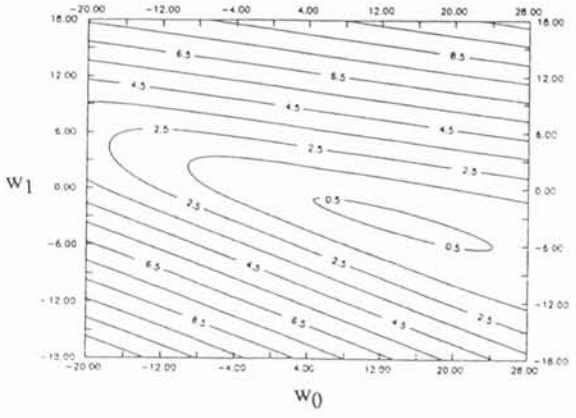
5 G.E. Hinton, 'Connectionist learning procedures', Technical Report CMU-CS-87-115, Carnegie Mellon University, 1987.

6 M.J.J. Holt and C.S. Xydeas, 'Compression of document image data by symbol matching', Proc. Int. Conf. Advances in Image Processing and Pattern Recognition, North Holland, 1986, pp. 184-190.

7 W.K. Pratt et. al., 'Combined symbol matching facsimile data compression system', Proc. IEEE, Vol. 68, 1980, pp. 222-228.




(a) Square-error-sum (BP cost function)




(b) Negative log likelihood (EML cost function)

**Figure (1)** Contour Plots of BP and EML cost functions

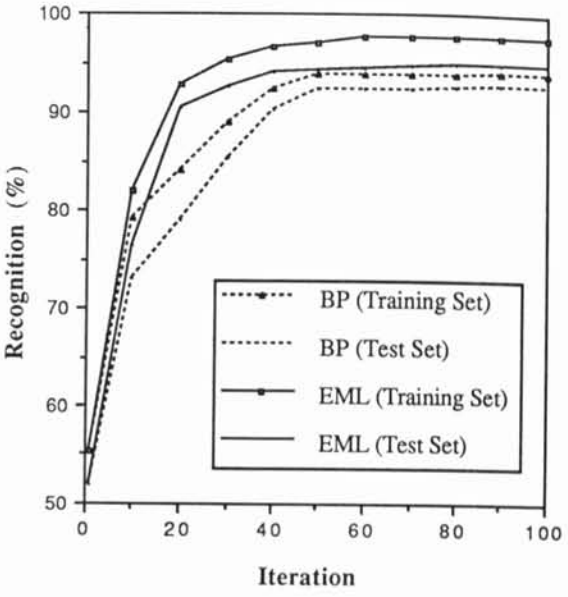



**Figure (2)** Learning and Generalisation curves BP and EML.