Fast Training of Multilayer Perceptrons with a Mixed Norm Algorithm

Sabeur Abid[1], Farhat Fnaiech[1], B. W. Jervis[2], and Mohammed Cheriet[3]

[1] ESSTT, 5 Av. Taha Hussein, 1008, Tunis, Tunisia, Email: Sab.abid@esstt.rnu.tn; Fnaiech@ieee.org
[2] Applied Electronics Research Group, School of Engineering, Sheffield Hallam University, Sheffield, S1 1WB, England, Email: B.W.Jervis@shu.ac.uk
(3) Laboratory for Image, Vision and artificial intelligence, ETS 1100 Rue Notre dame Ouest, Montreal, Qc, H3c 1K3, Canada. Email : Cheriet@gpa.etsmtl.ca

ABSTRACT

A new fast training algorithm for the Multilayer Perceptron (MLP) is proposed. This new algorithm is based on the optimization of a mixed Least Square (LS) and a Least Fourth (LF) criterion producing a modified form of the standard back propagation algorithm (SBP). To determine the updating rules in the hidden layers, an analogous back propagation strategy used in the conventional learning algorithms is developed. This permits the application of the learning procedure to all the layers. Experimental results on benchmark applications and a real medical problem are obtained which indicates significant reduction in the total number of iterations, the convergence time, and the generalization capacity when compared to those of the SBP algorithm.

Index terms: Multilayer perceptron (MLP), Fast training algorithm, Least Square (LS), Least Fourth (LF), Standard Backpropagation (SBP).

1. Introduction

The Standard Back propagation (SBP) algorithm is the most widely applied multilayer neural-network learning algorithm. Unfortunately, it suffers from a number of shortcomings. One such shortcoming is the slow convergence. Several iterations are required to train a small network, even for a simple problem. Reducing the number of iterations and speeding the learning time of MLP are appealing subjects of research [1-5].

In [1], the authors have proposed a new fast algorithm based on a modified form of the conventional SBP algorithm. It

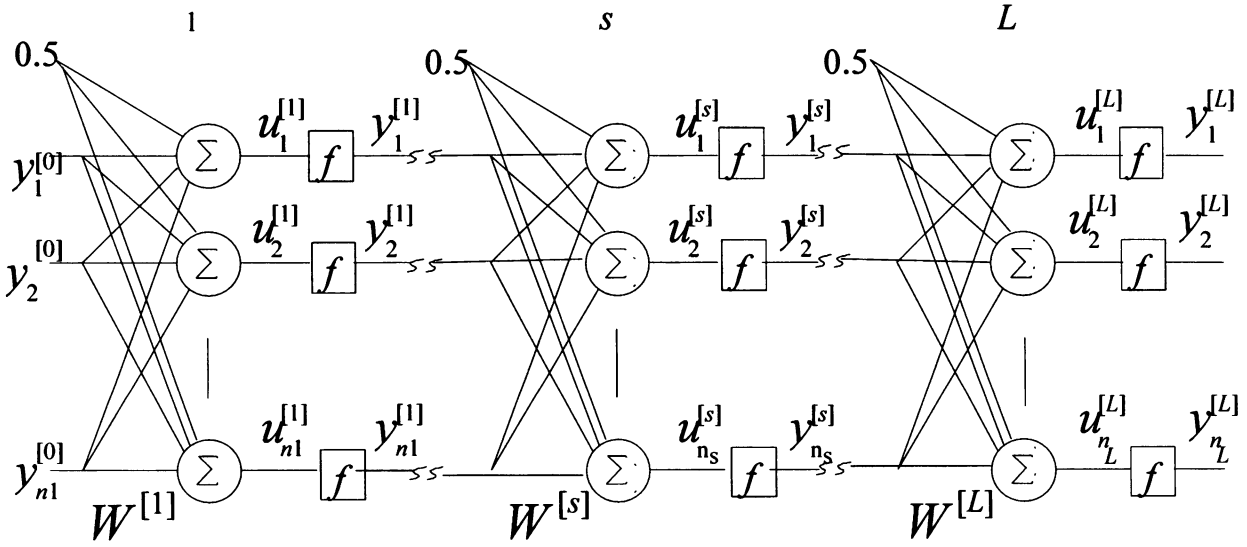

Fig.1: Fully connected feedforward multilayer perceptron.

0-7803-9048-2/05/$20.00 ©2005 IEEE

consists of minimizing the sum of the squares of linear and nonlinear errors. In [4], the authors have used the Marquardt algorithm to design a new fast MLP training algorithm, but this algorithm is still of a high computation complexity.

In this paper a new algorithm is presented, which is considerably faster than the SBP algorithm. This algorithm is based on a Mixed Least Square and Least Fourth (MLSLF) algorithm.

The Least Square and the Least Fourth algorithms belong to the family of the Least Mean square (LMS) and the Least Mean Fourth (LMF) algorithms. The LMS algorithm is the most widely used algorithm for adaptive filters in many applications [6]. Subsequently, the LMF algorithm was also proposed as a special case of the more general family of steepest descend algorithms [7] [8].

The new proposed MLSLF algorithm consists of minimizing a criterion expressed as the sum of the squares and the fourths errors for all output neurons and for the current pattern. Training patterns are run through the network until convergence is reached. We found that this algorithm requires a lower number of iterations for convergence in comparison with the SBP algorithm. In some examples we can reach a decrease of more than 40% training iterations with better generalization.

Note that although for completeness we shall treat the case of more than one hidden layer. Sometimes more accurate solutions can be more rapidly achieved with two layers.

The paper is structured as follows: in section 2 we describe the feedforward MLP used and describe the computation of the parameters. In section 3 we introduce the new MLSLF algorithm. In section 4, experimental results and comparison between the MLSLF and the SBP algorithms are given. Finally, in section 5, we present the main conclusions.

2. Review of the Standard Backpropagation algorithm

The SBP algorithm has become the standard algorithm used for training multilayer perceptron as shown in Fig.1. It is a generalized Least Mean Squares (LMS) algorithm that minimizes the sum of the squares of the errors between the actual and the desired outputs, E_p.

The linear and nonlinear neuron outputs for the current pattern are given respectively by

$$u_j^{[s]} = \sum_{i=0}^{n_{s-1}} w_{ji}^{[s]} y_i^{[s-1]} \tag{1}$$

$$f(u_j^{[s]}) = \frac{1}{1+e^{-u_j^{[s]}}} = y_j^{[s]} \tag{2}$$

Now:

$$E_p = \frac{1}{2}\sum_{j=1}^{n_L} \left(e_j^{[L]}\right)^2 \tag{3}$$

where the nonlinear error signal is:

$$e_j^{[L]} = d_j^{[L]} - y_j^{[L]} \tag{4}$$

$d_j^{[L]}$ and $y_j^{[L]}$ are respectively the desired and the current outputs, for the j^{th} unit. In (3), P denotes the p^{th} pattern, n_L is the number of the output units. The gradient descent method is given by:

$$\Delta w_{ji}^{[s]} = -\mu \frac{\partial E_p}{\partial w_{ji}^{[s]}} \tag{5}$$

where $w_{ji}^{[s]}$ is the weight of the i^{th} unit in the $(s\text{-}1)^{th}$ layer to the j^{th} unit in the s^{th} layer. Since the SBP algorithm is treated in the literature [9][10], only a summary of the important steps is given here:

- Compute the error signals for the output layer using:

$$\delta_j^{[L]} = f'(u_j^{[L]}) e_j^{[L]} \tag{6}$$

- Compute the error signals for the hidden layers, i.e. for $s = L\text{-}1$ to 1, by:

$$\delta_j^{[s]} = f'(u_j^{[s]}) \sum_{r=1}^{n_{s+1}} \left(\delta_r^{[s+1]} w_{rj}^{[s+1]}\right) \tag{7}$$

- Update the weights according to the following equation:

$$w_{ji}^{[s]}(k+1) = w_{ji}^{[s]}(k) + \mu \delta_j^{[s]} y_i^{[s-1]} \tag{8}$$

where μ is the learning coefficient and f' is the first derivative of f.

As we note from (3) and (4), the SBP algorithm minimizes the squared error at the output of the activation function with respect to the weights.

In order to increase the convergence speed of the SBP, we propose to use a combined norm in the optimization criterion.

Table I : The New MLSLF algorithm

Step 1: Initialization:

* From layer s=1 to L, set all $y_0^{[s-1]}$ to values different from 0, (0.5 for example).

* Randomize all the weights $w_{ji}^{[s]}$ at random values.

* Choose a small positive value β.

Step 2: Select training pattern:

Select an input/output pattern to be fed into the network.

Step 3: Run selected pattern p through the network for each layer (s), (s = 1... L) and calculate for each node j the linear and nonlinear outputs: equations (1) and (2).

Step 4: Error signals:

* For the output layer L: calculate the output errors: equation (4)

* For the hidden layers: s =L-1 to 1 compute the local signal errors: equations (17) and (18).

Step 5: Updating the synaptic coefficients:

For the output layer L, update the weights: equation (10)

For any node j of the layer s=1 to L modify the synaptic coefficients using equation (16).

Step 6: Testing for the ending of the running:

Various criteria are tested for ending. We can use the mean square error of the network output as a convergence test or we can run the program for a fixed number of iterations. If the condition is not verified, go back to Step 2.

3. The New Mixed Least Square and Least Fourth (MLSLF) Algorithm.

The new proposed minimizing criterion, based on the squared and fourth errors is given by:

$$E_p = \sum_{j=1}^{n_L} \frac{1}{2}\left(e_j^{[L]}\right)^2 + \sum_{j=1}^{n_L} \frac{1}{4}\beta\left(e_j^{[L]}\right)^4 \qquad (9)$$

where β is a positive weighting coefficient governing the amplitude of the Least Forth quantity.

Next, we derive the updating equations for both the output and the hidden layers.

a. Learning of the output layer

The weight update rule for the ouput layer can be derived by applying the gradient descent method to E_p, we get:

$$\Delta w_{ji}^{[L]} = -\mu \frac{\partial E_p}{\partial w_{ji}^{[L]}}$$

$$= \mu \frac{\partial y_j^{[L]}}{\partial u_j^{[L]}} \frac{\partial u_j^{[L]}}{\partial w_{ji}^{[L]}} e_j^{[L]}\left(1 + \beta\left(e_j^{[L]}\right)^2\right)$$

$$= \mu f'(u_j^{[L]}) y_i^{[L-1]} e_j^{[L]}\left(1 + \beta\left(e_j^{[L]}\right)^2\right) \qquad (10)$$

Note that for $\beta = 0$, we get the SBP algorithm.

b. Learning of the hidden layers

First we apply the gradient descent method to E_p for the layer [L-1] and then we generalize the results for the other hidden layers.

For the [L-1]th hidden layer we can write:

$$\Delta w_{ir}^{[L-1]} = -\mu \frac{\partial E_p}{\partial w_{ir}^{[L-1]}} \qquad (11)$$

Application of the chain rule yields:

Table II : Multiplication operation number of the MBP/SBP algorithm for one pattern

Algorithm	SBP	MLSLF
Errors	$2n_L + \sum_{s=1}^{L-1} n_s(n_{s+1} + 2)$	$2n_L + \sum_{s=1}^{L-1} n_s(n_s^2 + n_{s+1} + 2)$
Updating	$\sum_{s=1}^{L} n_s(n_{s-1} + 2)$	$\sum_{s=1}^{L} n_s(n_{s-1} + 3)$

$$\Delta w_{ir}^{[L-1]} = -\mu \frac{\partial E_p}{\partial e_j^{[L]}} \frac{\partial e_j^{[L]}}{\partial y_j^{[L]}} \frac{\partial y_j^{[L]}}{\partial u_j^{[L]}} \frac{\partial u_j^{[L]}}{\partial y_i^{[L-1]}} \cdot \frac{\partial y_i^{[L-1]}}{\partial u_i^{[L-1]}} \frac{\partial u_i^{[L-1]}}{\partial w_{ir}^{[L-1]}}$$

$$= \mu f'(u_i^{[L-1]}) y_r^{[L-2]} \sum_{j=1}^{n_L} e_j^{[L]} f'(u_j^{[L]}) w_{ji}^{[L]}$$

$$+ \mu\beta f'(u_i^{[L-1]}) y_r^{[L-2]} \sum_{j=1}^{n_L} \left(e_j^{[L]}\right)^3 f'(u_j^{[L]}) w_{ji}^{[L]} \quad (12)$$

Let us define the local signal errors by:

$$\delta_{1i}^{[L-1]} = \sum_{j=1}^{n_L} e_j^{[L]} f'(u_j^{[L]}) w_{ji}^{[L]} \quad (13)$$

$$\delta_{3i}^{[L-1]} = \sum_{j=1}^{n_L} \left(e_j^{[L]}\right)^3 f'(u_j^{[L]}) w_{ji}^{[L]} \quad (14)$$

The updating equation (12) for the [L-1]th layer becomes:

$$\Delta w_{ir}^{[L-1]} = \mu f'(u_i^{[L-1]}) y_r^{[L-2]} \delta_{1i}^{[L-1]} + \mu\beta f'(u_i^{[L-1]}) y_r^{[L-2]} \delta_{3i}^{[L-1]} \quad (15)$$

The differentiations may be performed layer by layer. Hence, for a given layer s the updating equation (10) becomes:

$$\Delta w_{ji}^{[s]} = \mu f'(u_i^{[s-1]}) y_r^{[s-2]} \delta_{1j}^{[s]} + \mu\beta f'(u_i^{[s-1]}) y_r^{[s-2]} \delta_{3j}^{[s]} \quad (16)$$

where :

$$\delta_{1j}^{[s]} = \sum_{j=1}^{n_s} e_j^{[s]} f'(u_j^{[s]}) w_{ji}^{[s]} \quad (17)$$

$$\delta_{3j}^{[s]} = \sum_{j=1}^{n_s} \left(e_j^{[s]}\right)^3 f'(u_j^{[s]}) w_{ji}^{[s]} \quad (18)$$

Finally, Table I summarizes the new algorithm.

c. Comparison of the computation complexity

Table II gives a comparison of the number of multiplication operations needed for each algorithm to compute the error signals and the updating equations for one pattern. Obviously, the proposed algorithm is slightly more complex than the SBP algorithm. However, it is shown below to have faster convergence behavior in terms of the number of iterations needed and in computation time.

4. Experimental results

To compare the performances of the new algorithm with respect to the conventional SBP, both algorithms are used to train the networks for the same problem. In this paper we present three examples, the 4-b parity checker (logic problem), the circle in the square problem (analog problem), and the brain diseases classification (a real medical problem). For both algorithms, the learning parameters are selected with respect to a performance criterion. However, an exhaustive search for the best possible parameters is beyond the scope of this paper, and optimal values may exist for either algorithm. In order to make suitable comparison we keep the same neural network size for testing both training algorithms.

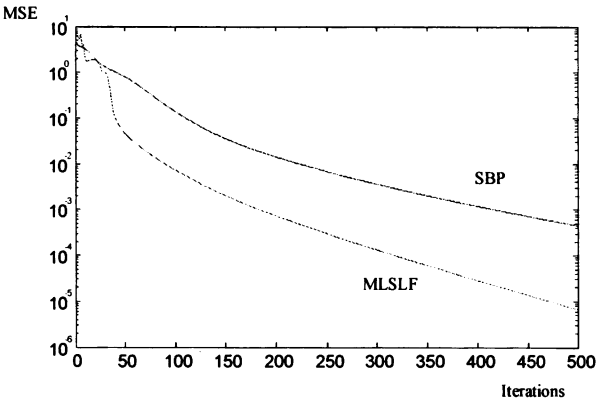

Fig. 2: Learning curve for the MLSLF versus SBP algorithm for the 4-b parity checker (μ=5; β=5.01)

a. The 4-b parity checker

The aim of this application is to determine the parity of a 4-bit binary number. For more details see [1]. Fig. 2 shows the average of the Mean Squared Error (MSE) over 100 different weight initializations. We note that the MLSLF algorithm is highly faster than the SBP. The new MLSLF algorithm remains below 5×10^{-4} after only 220 iterations as opposed to the SBP algorithm at 500 iterations. Based on these experiments, clearly we see that there is an improvement ratio, nearly 2.25, for the number of iterations. The computation time per iteration was calculated and found to be similar for the two algorithms. Then the improvement ratio for the convergence time is about 2. We note that the new algorithm is very sensitive to the choice of β and one should make many trials to find a good value to ensure the rapidity of the algorithm.

b. Circle in the square problem

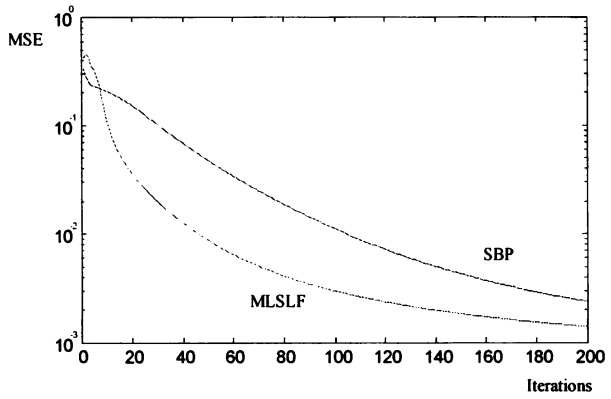

Fig. 3: Learning curve for the MLSLF versus SBP algorithm for the circle in the square problem

In this application, the neural network have to decide if a point of coordinate (x, y) varying from -0.5 to +0.5 is in the circle of radius equal to 0.35 [1]. Fig. 3 shows the average of the MSE over 100 different weight initializations versus the iteration number for both algorithms during training. The new algorithm remains below $2x10^{-2}$ after 100 iterations as opposed to the SBP algorithm at 180 iterations. Notice that there is an improvement ratio of about 1.8 in the number of iterations and in the learning time.

c. Brain diseases classification

In this application, we use features from Contingent Negative Variation (CNV) waveforms of the electroencephalograms to classify four types of subjects: Huntington's disease and Parkinson's disease patients, patients with schizophrenia, and normal subjects. To study the redundancy between the CNV data used for the disease classification before the training step, we define a characterization capability as the ratio between "inter-class" and "intra-class" deviations of each feature [3].

The features are set into vectors $x_{k,n}$ of 17 elements, with k the disease index and n the example index. Let define N_k as the number of examples in disease class k (11, 16, 20 and 40 respectively for the Huntington's disease, Parkinson's disease, schizophrenia and normal cases).

For classification purposes, the subjects were classified in the conventional way using the different features of CNV. These features were used as input vectors to the MLP neural network. Two networks were trained using the SBP and the MLSLF algorithms with 75% of the available data and tested with the remaining 25%. The network weights were updated on each presentation of a feature vector. For each disease class, the *classification sensitivity* is defined as the ratio of positive tests over the total number of training tests. The trained networks have 17 input units, two hidden layers with 7 and 11 units respectively, and 4 output neurons. The threshold for ending the learning phase is chosen to be a mean squared error of 0.01. The performances of the two algorithms are summarized in Table III.

5. Conclusion

In this paper, we have proposed a new fast algorithm for training neural networks based on a new criterion combining the squared and the fourths errors. The convergence of the new MLSLF algorithm requires less iterations than the SBP and provides better generalization. Comparing to the SBP, the MLSLF needs one more tuning parameter β that governs the learning speed and can cause the divergence in case of bad choice. It is very important to develop the theory to find rules that help to choose this factor. It will be also important to compare the performances of this new algorithm with other fast algorithms like the Modified Back propagation algorithm introduced in [1], Recursive Least Square algorithm in [5] and the Marquardt algorithm in [4].

Table III: Results of neural network training with the MLSLF and SBP algorithms (μ=0.5; β=1.25)

	SBP	MLSLF
Number of iterations for reaching convergence	2800	1900
Computation time (s)	980	630
Sensitivity of learning (%)	81.03	87.51
Sensitivity of generalization (%)	55.55	75.22

6. References

[1] S. Abid, F. Fnaiech and M. Najim: "A Fast Feed-Forward Training algorithm using A Modified Form of the standard Back Propagation Algorithm" *IEEE Trans. Neural Network* Vol.12, No 2, March 2001.

[2] S. Abid, F. Fnaiech, and M. Najim, "Evaluation of the feedforward neural network covariance matrix error", *IEEE International Conference on Acoustic Speech and Signal Processing* (ICASSP' 2000) June 5-9, Istanbul, Turkey.

[3] N. Fnaiech, M. Sayadi, F. Fnaiech, B. W. Jervis, and M. Cheriet, "A Pruned Multilayer Neural Network For Brain Diseases Classification", *Proc. of* NNESMED/CIMED 03, July 2003 Sheffield, UK.

[4] M. T. Hagon and M. B. Menhaj "Training feedforward networks with the Marquardt algorithm" *IEEE Transactions on Neural Networks*, Vol. 5, pp. 989, November 1994.

[5] R. S. Scalero and N. Tepedelenglioglu, "A Fast New Algorithm for Training Feedforward Neural Networks" *IEEE Transactions on signal processing.* Vol 40. No, 1, January 1992.

[6] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, Englewood Cliffs, NJ, 1991.

[7] E. Walach and B. Widrow, "The Least Mean Fourth (LMF) Adaptive Algorithm and its family", *IEEE Trans. Inf. Theory, vol.* IT-30, pp. 275-283, Feb. 1984.

[8] A. Zerguine "A Time-Varying Normalized Mixed-Norm LMS-LMF Algorithm", 11th European Signal Processing Conference EUSIPCO 2002, Paper coded #394. Toulouse, France, 03-06 September 2002.

[9] A. Cichocki. R. Unbehauen, "Neural network for optimization and signal processing", John Wiley & Sons Ltd. Baffins Lane, Chechester. West Sussex PO19 1UD, England 1993.

[10] R. P. Lippman, "An introduction to computing with neural networks", *IEEE. ASSP. Mag.* Vol 4. No, 2 April, 1987.