TABU BASED BACK PROPAGATION ALGORITHM FOR PERFORMANCE IMPROVEMENT IN COMMUNICATION CHANNELS

Prof. J. K. Satapathy*, K. R. Subhashini**
Dept. of Electrical Engg, NIT, Rourkela-769008
India
*jksatapathy@nitrkl.ac.in **subppy@gmail.com

***Abstract*-This paper presents a new approach to equalization of communication channels using Artificial Neural Networks (ANNs). A novel method of training the ANNs using Tabu based Back Propagation (TBBP) Algorithm is described. The algorithm uses the Tabu Search (TS) to improve the performance of the equalizer as it searches for global minima which is many a time escaped while Back Propagation (BP) algorithm is applied for this purpose. From the results it can be noted that the proposed algorithm improves the classification capability of the ANNs in differentiating the received data.**

KEY WORDS
Artificial Neural Networks, Tabu Search, Equalization, Local Minima, Global Solution.

I. INTRODUCTION

The Back Propagation (BP) Algorithm revolutionized the use of Artificial Neural Networks (ANNs) in diverse fields of science and engineering, such as pattern recognition, function approximation, system identification, data mining, time series forecasting etc. These ANNs construct a functional relationship between input and output patterns through the learning process, and memorize that relationship in the form of weights for later applications [1, 2].

This BP algorithm belongs to the family of Gradient-based algorithms and they provide an easy way of supervised learning in multilayered feedforward ANNs. The method of gradient descent [3] is that a downhill movement in the direction of the negative gradient will eventually reach the minima of the performance surface over its parameter space. Since the gradient techniques converge locally, they often get trapped at suboptimal solutions [4] depending on the serendipity of the initial random starting point. Since obtaining a global solution is the main criterion of any adaptive system, an efficient search technique is highly desirable for such difficult nonlinear optimization problem.

The popularity of Tabu search has grown significantly in the past few years as a global search technique. The roots of the Tabu search go back to 1970's; it was presented in its present form by Glover [5, 6]. This technique is famous in solving many combinatorial problems like the traveling salesmen problem, design optimization, and the quadratic assignment problem. In this paper it is proposed to apply this technique to find the so called optimal values of the ANN parameters with reference to equalization of communication channels.

In section II the process of equalization is discussed. In section III a brief introduction to neural networks and its training using BP algorithm is presented. In section 1V proposed algorithm of using TS for training the ANNs is described. SectionV is dedicated for discussion on experimental results.

II. CHANNEL EQUALIZATION

The two principal causes of distortion [7] in a digital communication channels are Inter Symbol Interference (ISI) and the additive noise. The ISI can be characterized by a Finite Impulse Response (FIR) filter [8, 9]. The noise can be internal to the system or external to the system. Hence at the receiver the distortion must be compensated in order to reconstruct the transmitted symbols. This process of suppressing channel induced distortion is called channel equalization. The equalization in digital communication scenario is illustrated in Fig.1,

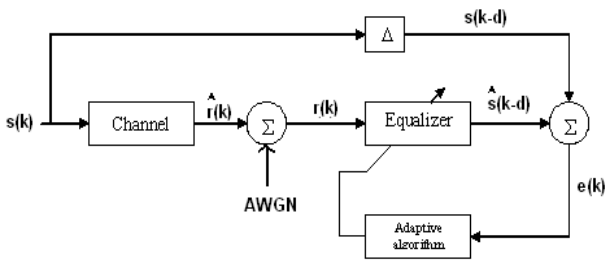

Figure. 1. Schematic of a Digital Communication System

where $s(k)$ is the symbol sequence transmitted through the channel. $N(k)$ represents Additive White Gaussian Noise (AWGN). $r(k)$ is the received data and $\hat{s}(k-d)$ is an estimate of the transmitted data. d is called the decision delay.

The channel can be modeled as an FIR filter with a transfer function

$$A(z) = \sum_{i=0}^{n_a-1} a_i z^{-i} \qquad (1)$$

where n_a is the length of the communication channel impulse response. The symbol sequence $s(k)$ and the channel taps a_i can be complex valued. In this study, however, channels and symbols are restricted to be real valued. This corresponds to the use of multilevel pulse amplitude modulation (*M-ary* PAM) with a symbol constellation defined by

$$s_i = 2i - M - 1, \quad 1 \le i \le M \qquad (2)$$

Concentration on the simpler real case allows us to highlight the basic principles and concepts. In particular, the case of binary symbols ($M = 2$) provides a very useful geometric visualization of equalization process.

The task of equalizer is to reconstruct the transmitted symbols as accurately as possible based on the noisy channel observations $r(k)$. Various equalizers can be classified into two categories, namely, the symbol-decision equalizer and the sequence-estimation equalizer. The later type is hardly used as it is computationally very expensive. The symbol-decision equalizers in its initial stages were implemented using Linear Transversal Filters. Later the advent of ANNs marked the modeling of equalizers which can provide superior performance in terms of Bit Error Rate (BER) compared to FIR modeling.

III. Neural Network based Equalizer

In this section a neural network based equalizer is considered. The neural network based equalizer outperforms the Linear Transversal filters in terms of BER as most of the communication channels requires nonlinear decision boundary [10]. The structure of the equalizer is shown in Fig. 2.

In the figure $r(k)$ represents received signal. The structure constitutes three significant parts- one input layer, a set of hidden layers, one output layer. All the nodes are interconnected by the weights w_{ij}^{l}, where i represents the destination node and j represents the source node. The superscript l gives the layer number.

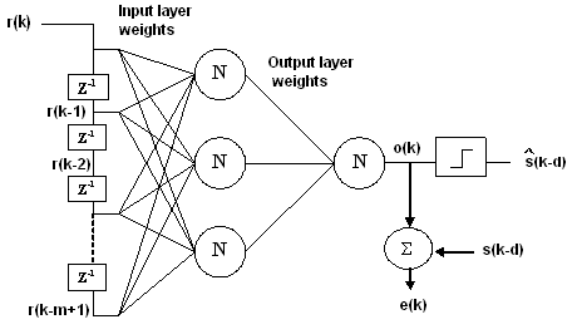

Figure. 2. Neural Network Equalizer.

An equalizer of order m implies that it has m input nodes in its input layer as shown in the figure. An equalizer will have a single node in its output layer. The signal received sequentially is allowed to propagate through the hidden layers up to the node in the output layer [11, 12].

The output of the each node y_i^l is the weighted sum of outputs of all the nodes in the previous layer and affected by the activation function, which here is the hyperbolic tangent function given by

$$\phi(x) = \frac{1 - e^{-ax}}{1 + e^{-ax}} \qquad (3)$$

where a represents the slope of the activation function. Mathematically the forward propagation of the neural network is given by [12]. $v_i^l = \sum_{j=1}^{N_{l-1}} (w_{ij}^{l-1} \cdot y_j^{l-1})$ (4)

$$y_i^l = \phi(v_i^l) \qquad (5)$$

where v_i^l is called the induced local field or activation potential of node i in layer l and N_{l-1} is the number of neurons in the layer $(l-1)$.

3.1 *Back Propagation Algorithm*

The BP algorithm consists of two passes through the different layers of the network: a forward pass and a backward pass [11]. The forward pass is mentioned earlier. In the backward pass the error signal, which is obtained by comparing the output of the node in the output layer with the desired response, is allowed to propagate against the direction of synaptic weights (hence the name back propagation algorithm) and local gradients δ_i^l at each node is computed using the following relation

$$\delta_j^l = \phi'(v_j^l) \cdot \sum_{i=1}^{N_{l+1}} (\delta_i^{l+1} \cdot w_{ji}^l) \qquad (6)$$

Due to the lack of availability of desired response at the hidden layers, it is not possible to compute the error at these nodes. Hence this local gradient is very important in providing the error measure at the hidden nodes. Using these local gradients the synaptic weights are updated as shown below.

$$\begin{pmatrix} Weight \\ Correction \\ \Delta w_{ji}^{l} \end{pmatrix} = \begin{pmatrix} learningrate \\ parameter \\ \eta \end{pmatrix} \bullet \begin{pmatrix} local \\ gradient \\ \delta_{j}^{l+1} \end{pmatrix} \bullet \begin{pmatrix} inputsignal \\ ofneuron \\ y_{i}^{l} \end{pmatrix}$$

The weight correction Δw_{ji}^{l} is added to the present weight after each iteration. The weights are updated till the mean square error (MSE) falls below some chosen threshold or when maximum number of iterations are completed.

IV. Tabu Based BP (TBBP) Algorithm

In this section first a short description to the Tabu search (TS) algorithm is presented and then the proposed algorithm of using TS for Back Propagation is discussed.

4.1 Tabu Search

Tabu search can be thought as an iterative descent method. An initial solution is randomly generated and a neighborhood around that solution is examined. If a new solution is found in the neighborhood that is preferred to the original, then the new solution replaces the old and the process repeats. If no solution is found that improves upon the old function evaluation, then unlike a gradient descent procedure which would stop at that point (a local minima), the TS algorithm may continue by accepting a new value that is worse than the old value.

Therefore a collection of solutions in a given neighborhood is generated and the final solution would be the best solution found so far for that particular neighborhood.

To keep from cycling, an additional step is included that prohibits solutions from recurring (hence the name Tabu) for a user defined number of solutions. This Tabu List (TL) is generated by adding the last solution to the beginning of the list and discarding the oldest solution from the list. During this procedure, the best solution found so far is retained. If the new generated weight is not in the TL, the search goes on and the weight is added to the TL.

To reject any solution, all the weights must be within the Tabu Ar ea (TA) of any entry in the TL. The Aspiration Criterion (AC) is used to activate the solutions that are tabued but around which there are some superior solutions.

4.2 The TBBP

The TBBP can be divided into two steps - the Superficial Search (SS) and the Deep Search (DS). In the SS we search for the solution which has the higher probability of finding good global solutions. The DS trains these solutions, found in SS, to find the best solution in the neighborhood of the solution.

The original weight $\mathbf{W}_0$ is randomly generated, where $\mathbf{W}_0$ include all the weights of the neural network. The SS trains this original weight to a state, $\mathbf{W}_0^{'}$ a point in the local minima, but is not at the bottom of the concave [4]. If the point is in TL, then this is considered to be in a searched concave and is not considered for DS.

There may be some $\mathbf{W}_i^{'}$s, where $i = 0,1,2,...$ which may be in the searched concave but may satisfy

$$E(\mathbf{W}_i^{'}) < (1 - AC)E(\mathbf{W}_b) \text{ or}$$
$$E(\mathbf{W}_i^{'}) < (1 + AC)E(\mathbf{W}_b) \qquad (7)$$

where $E(\mathbf{W}_i^{'})$ is the sum of error evaluated at the superficial state $\mathbf{W}_i^{'}$ and $\mathbf{W}_b$ is the best solution found so far.

The above equation is called the AC and is used to activate some of the tabued solutions. The solution obtained in the SS is further searched, called DS, in its neighborhood.

Steps involved in TBBP

Here the basic steps involved in the algorithm are discussed. It mainly consists of 7 steps.

(1) Generate an initial solution $\mathbf{W}_i$, $i = 0,1,2....$.

(2) Superficial search:

(i) The initial weight is trained with BP algorithm to obtain Superficial state $\mathbf{W}_i^{'}$ and $E(\mathbf{W}_i^{'})$.

(ii) If this solution is in TL and does not satisfy AC then go to step (1) to generate new solution.

(iii) Else add the solution to the TL and go to step (3) for DS.

(3) Deep Search:

(i) Deeply search $\mathbf{W}_i^{'}$ and get the corresponding deep state $\mathbf{W}_i^{''}$ and get its corresponding $E(\mathbf{W}_i^{''})$.

(ii) If $E(\mathbf{W}_i^{''}) < E(\mathbf{W}_b)$ then set $\mathbf{W}_b = \mathbf{W}_i^{''}$ and $E(\mathbf{W}_b) = E(\mathbf{W}_i^{''})$.

(4) Generate a new solution $\mathbf{W}_{ij}^{'}$ in the neighbor-hood of $\mathbf{W}_i^{'}$ and evaluate $E(\mathbf{W}_{ij}^{'})$.

(5) If this new superficial solution $\mathbf{W}_{ij}^{'}$ is in TL and does not satisfy AC then go to step (4) to generate a new neighbor, else add $\mathbf{W}_{ij}^{'}$ to TL and go to step (6).

(6) Deeply search $\mathbf{W}_{ij}^{'}$ similar to step (3) and update the best solution if the squared error obtained in this deep search is less than the best error square till now.

(7) If j is less than maximum number of neighborhood searches go to step (4). Or else finalize $\mathbf{W}_b$ as the best solution

IV. Experimental Results

In this section the experimental results are taken into account in order to compare the performance of the equalizer trained using BP and TBBP.

Both the BP and TBBP algorithms are written in C and compiled using Microsoft VC++ 6.0. The plots have been taken using Microsoft Excel 2003.

$$H_1(z) = 1.0 + 0.5z^{-1}$$

$$H_2(z) = 0.3482 + 0.8704z^{-1} + 0.3482z^{-2}$$

$$H_3(z) = 0.4084 + 0.8164z^{-1} + 0.4084z^{-2}$$

$$H_4(z) = 0.35 + 0.8z^{-1} + 1.0z^{-2} + 0.8z^{-3}$$

$$H_5(z) = 0.9413 + 0.3841z^{-1} + 0.5684z^{-2} + 0.4201\ z^{-3} + 1.0\ z^{-4}$$

[12,13,14] For this experiment a simple three layered neural network with decision feedback is considered. The equalizer order is chosen to have $m = 2,3,5$, feedback order $n_b = 1,2,4$ decision delay $d = 0,2,4$ and in the hidden layer only one node is considered to design an efficient and compact equalizer with an objective to appreciate the superior performance of TS even with this very small structure

In figure3 through figure7 it is shown that the performance of the equalizer when trained using TBBP is far better than that trained using BP .The Optimal performance of each channel is also plotted and in Table 1 the number of wrong decisions made by the equalizer when trained using BP algorithm and TBBP are tested. The figures are obtained by testing the equalizer with 10^6 samples.

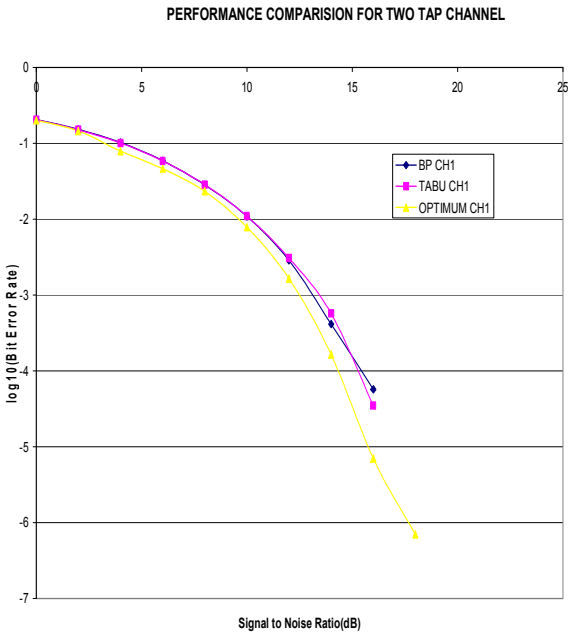

Figure 3:SNR Vs BER plot for BP and TABU of $H_1(Z)$

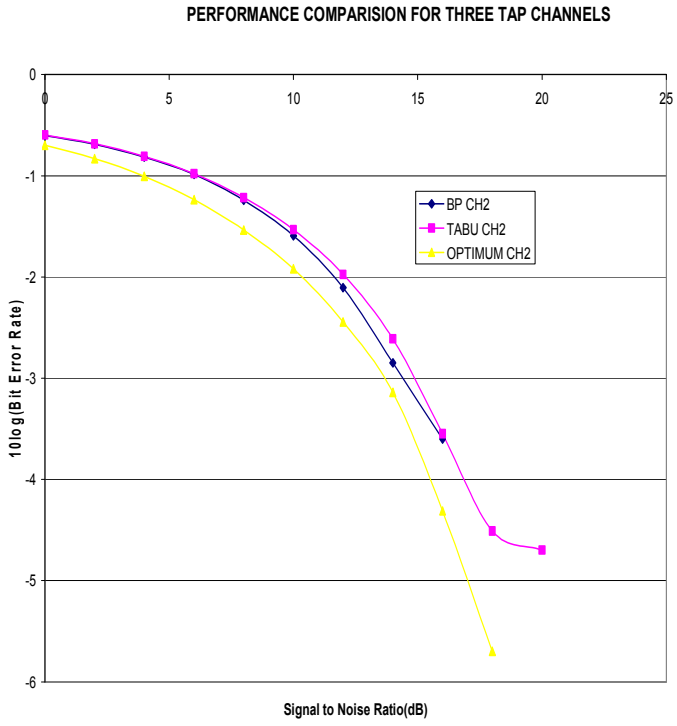

Figure 4: SNR Vs BER plot for BP and TABU of $H_2(Z)$

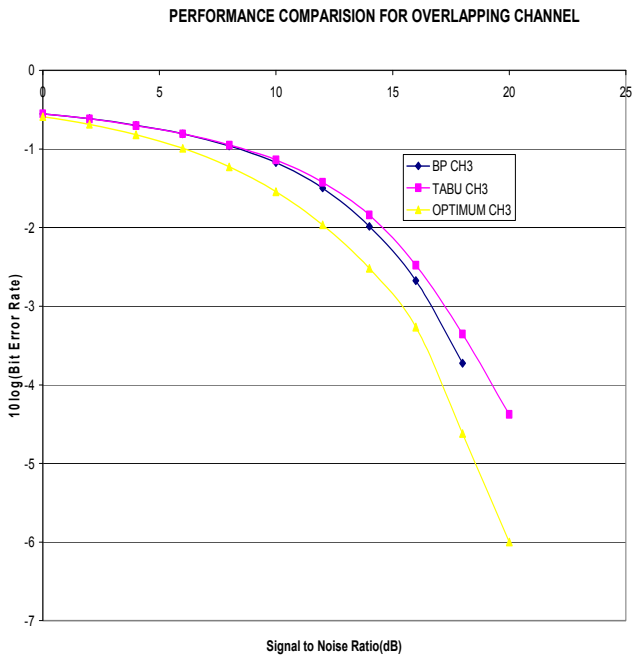

Figure 5: SNR Vs BER plot for BP and TABU of $H_3(Z)$

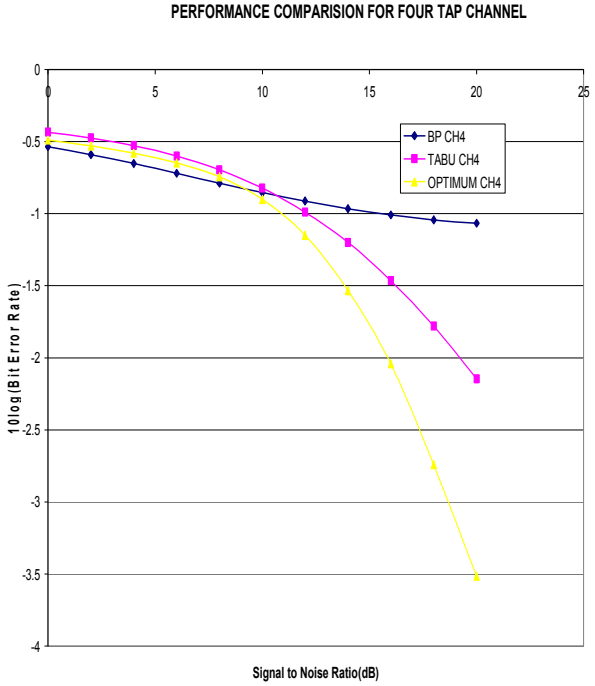

Figure 6:SNR Vs BER plot for BP and TABU of $H_4(Z)$

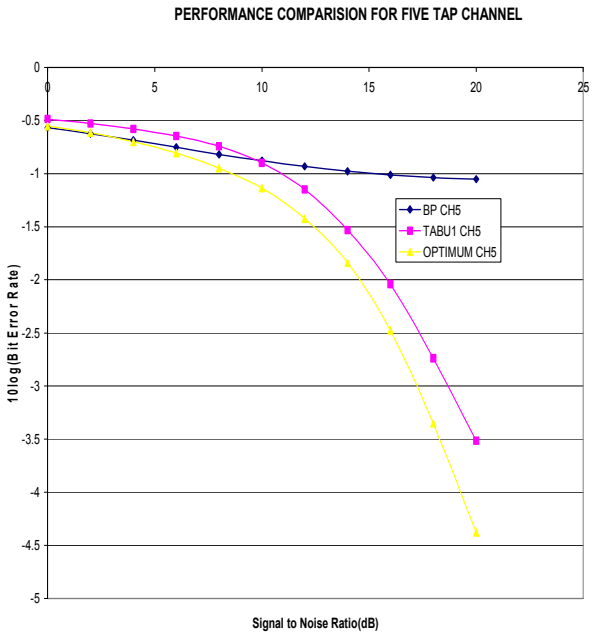

Figure 7: SNR Vs BER plot for BP and TABU of $H_5(Z)$

In TBBP, algorithm 50 superficial states have been searched and 100 neighbor solutions are generated in the neighbor-hood of each superficial state. But BP algorithm based neural network is trained with 2000 samples. In table 1, the figures say that the number of errors that occur in decision making by the equalizer are very less when trained by TBBP compared to training with BP. At 20dB SNR number of errors that occur during the classification is more than 80,000 for the case of BP. But when trained with TBBP, it is seen that slightly greater than 100 decision errors have occurred. This shows the superior performance of Tabu based BP algorithm compared to simple BP algorithm.

Table 1. Comparison of Number of errors in decision making

SNR(dB)	BP	TBBP
0	326628	322552
2	298639	292553
4	268182	259243
6	236535	220696
8	206227	174371
10	175914	118052
12	148047	62426
14	124420	24085
16	106366	6637
18	93063	1168
20	83081	111

To obtain similar performance, the BP algorithm needs a more complex structure. In figure 8 it is shown that for the

similar performance the BP algorithm needs 6 nodes in its hidden layer.

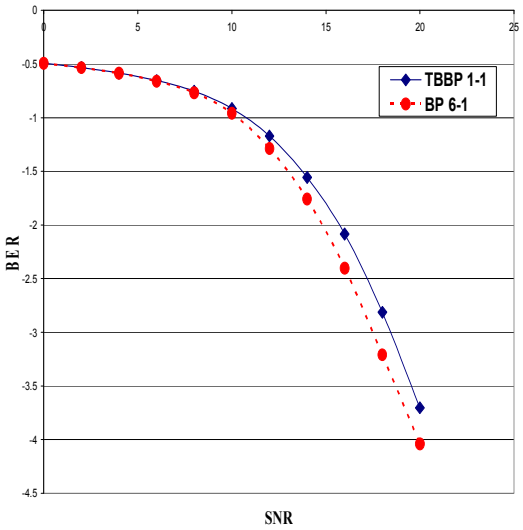

Figure. 8. SNR Vs BER plot for BP with 6-1 structure and TBBP with 1-1 structure

For a BP algorithm, it needs $(5+4)\times 6+6=60$ adjustable parameters (weights) to obtain the similar performance that a TBBP algorithm gives with just $(5+4)\times 1+1=10$ weights, i.e. the proposed TBBP algorithm reduces the required number of adjustable parameters from 60 to 10.

V. Conclusion

In this paper we propose a novel method of training a neural network using TBBP is proposed. The principal advantages of using the Tabu search is that it can jump out of the local minima by extending its search into the global space. Another advantage is that it avoids the searched concaves effectively and hence saving time. This paper presents the efficiency of the search, algorithm together with the neural network in improving the performance of the equalizer even with a simple structure.

References

[1] S. Haykin, *Neural Networks:A Comprehensive Foundation* (2nd Ed, Pearson Education, 2001)

[2] R. P. Lippmann, An Introduction to Computing with Neural Nets, *IEEE ASSP Magazine*,1987,4-22.

[3] B. Widrow and SD Stearns, *Adaptive Signal Processing*, (Englewood Cliffs , NJ: Prentice Hall 1985).

[4] Jian Ye, Junfei Qiao, Ming-ai Li,Xiaogang Ruan, A Tabu based neural network learning algorithm, *Neurocomputing, 70,* 2007, 875-882.

[5] F. Glover, Tabu Search – Part I, *ORSA Journal on Computing, vol.1,*1989, 190-206.

[6] F. Glover, Tabu Search – Part II, *ORSA Journal on Computing, vol.2,* 1990, 4-32.

[7] S.Haykin, *Adaptive Filter Theory,*(4th Ed, Pearson Education, 2002)

[8] S. Qureshi, Adaptive Equalization, *Proc IEEE*, 1985, 1349-1387.

[9] J. G. Proakis, *Digital Communications,* (New York: McGraw-Hill, 1983).

[10] S. Siu, G. J. Gibson and C. F. N. Cowan,Multi-layer perceptron structures applied to adaptive equalizers for data communications, *IEEE Proceedings ICASSP* Glasgow, Scotland, May 1989, 1183-1186

[11] Zhang. De-Xian, Liu. Can, Wang. Zi-Quiang,Liu.Nan-Bo, "A New Fast Learning Algorithm for Multilayer Feed Forward Neural Network", IEEE Proc. of ICMLC, (August 2006): pp- 2928-2934.