International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

# Training Feed Forward Neural Networks Using Optimized Back Propagation Algorithm

**V. Sowjanya**
Asst. Professor, Department of CSE,
Prasad V Potluri Siddhatha Institute of Technology, kanuru, Vijayawada.

**J. Sirisha**
Asst. Professor, Department of IT,
Prasad V Potluri Siddhatha Institute of Technology, kanuru, Vijayawada.

**T. Kranthi kumar**
Asst. Professor, Department of MCA,
Prasad V Potluri Siddhatha Institute of Technology, kanuru, Vijayawada.

***Abstract* — Neural network (NN) is one of the most important data mining techniques. It is used with both supervised and unsupervised learning. Training NN is a complex task of great importance in problems of supervised learning. Most of NN training algorithms make use of gradient-based search. These methods have the advantage of the directed search, in that weights are always updated in such a way that minimizes the error, which called NN learning process. However, there are several negative aspects with these algorithm such as dependency to a learning rate parameter, network paralysis, slowing down by an order of magnitude for every extra (hidden) layer added and complex and multi-modal error space. The back-propagation algorithm is one of the most famous algorithm to train a feed forward network. Instead of its success rate the quest for development is observed through the various standard modifications in-order to meet the challenges of complex applications. This paper proposes an optimized back propagation algorithm to train feed forward artificial neural networks.**

***Keywords* — Error Rate, Neural Network, Stability, Supervised learning.**

## I. INTRODUCTION

Neural networks are members of a family of computational architectures inspired by biological brains [1] [2]. Such architectures are commonly called "connectionist systems", and are composed of interconnected and interacting components called nodes or neurons (these terms are generally considered synonyms in connectionist terminology, and are used interchangeably here). Neural networks are characterized by a lack of explicit representation of knowledge; there are no symbols or values that directly correspond to classes of interest. Rather, knowledge is implicitly represented in the patterns of interactions between network components[2]. A graphical depiction of a typical feedforward neural network is given in Figure 1. The term “feedforward” indicates that the network has links that extend in only one direction. Except during training, there are no backward links in a feedforward network; all links proceed from input nodes toward output nodes.

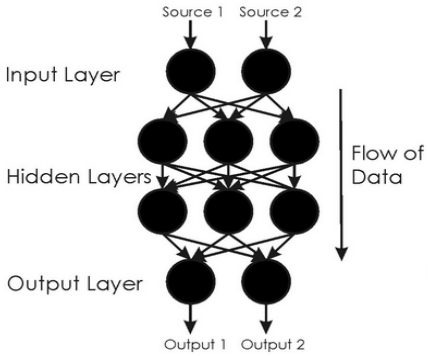


Fig.1. A typical feed forward neural network.

Individual nodes in a neural network emulate biological neurons by taking input data and performing simple operations on the data, selectively passing the results on to other neurons (Figure 2). The output of each node is called its "activation" (the terms "node values" and "activations" are used interchangeably here). Weight values are associated with each vector and node in the network, and these values constrain how input data (e.g., satellite image values) are related to output data (e.g., land-cover classes). Weight values associated with individual nodes are also known as biases. Weight values are determined by the iterative flow of training data through the network (i.e., weight values are established during a training phase in which the network learns how to identify particular classes by their typical input data characteristics).

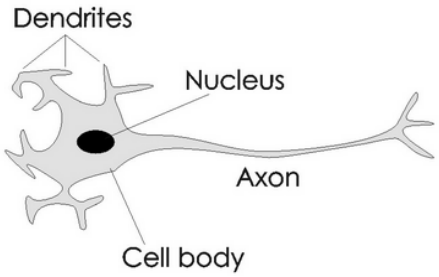


Fig.2(a)

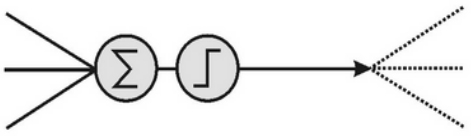

Fig.2(b)

Fig.2. Schematic comparison between a biological neuron (Fig.2a) and an artificial neuron (Fig.2b).

Figure 2 shows the Schematic comparison between a biological neuron and an artificial neuron [3][4]. For the biological neuron, electrical signals from other neurons are conveyed to the cell body by dendrites; resultant electrical

signals are sent along the axon to be distributed to other neurons. The operation of the artificial neuron is analogous to (though much simpler than) the operation of the biological neuron: activations from other neurons are summed at the neuron and passed through an activation function, after which the value is sent to other neurons.

Once trained, the neural network can be applied toward the classification of new data. Classifications are performed by trained networks through

- The activation of network input nodes by relevant data sources [these data sources must directly match those used in the training of the network].
- The forward flow of this data through the network.
- The ultimate activation of the output nodes. The pattern of activation of the network's output nodes determines the outcome of each pixel's classification.

In order to train a neural network to perform some task, we must adjust the weights of each unit in such a way that the error between the desired output and the actual output is reduced. This process requires that the neural network compute the error derivative of the weights. In other words, it must calculate how the error changes as each weight is increased or decreased slightly. The back propagation algorithm is the most widely used method for determining the error derivative of the weights. This paper presents an optimized back propagation algorithm to train feedforward neural networks.

## II. Architecture of Artificial Neural Networks

The basic architecture consists of three types of neuron layers: input, hidden, and output layers. In feed-forward networks, the signal flow is from input to output units, strictly in a feed-forward direction. The data processing can extend over multiple (layers of) units, but no feedback connections are present. Recurrent networks contain feedback connections. Contrary to feed-forward networks, the dynamical properties of the network are important. In some cases, the activation values of the units undergo a relaxation process such that the network will evolve to a stable state in which these activations do not change anymore. In other applications, the changes of the activation values of the output neurons are significant, such that the dynamical behavior constitutes the output of the network.

A neural network has to be configured such that the application of a set of inputs produces the desired set of outputs. Various methods to set the strengths of the connections exist. One way is to set the weights explicitly, using a priori knowledge. Another way is to train the neural network by feeding it teaching patterns and letting it change its weights according to some learning rule. The learning situations in neural networks may be classified into three distinct sorts. These are supervised learning, unsupervised learning.

### *A. Supervised Learning*

Supervised learning which incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be. During the learning process global information may be required. Paradigms of supervised learning include error-correction learning, reinforcement learning and stochastic learning. An important issue conserning supervised learning is the problem of error convergence, ie the minimisation of error between the desired and computed unit values. The aim is to determine a set of weights which minimises the error. The best-known examples of this technique occur in the backpropagation algorithm, the delta rule, and the perceptron rule.

### *B. Unsupervised Learning*

Unsupervised learning uses no external teacher and is based upon only local information. It is also referred to as self-organisation, in the sense that it self-organises data presented to the network and detects their emergent collective properties. Paradigms of unsupervised learning are Hebbian learning and competitive learning [5].

## III. Backpropagation Algorithms

Basic backpropagation [6] is currently the most popular supervised learning method that is used to train multilayer feedforward neural networks with differentiable transfer functions. It is a gradient descent al gorithm in which the network weights are moved along the negative of the gradient of the performance function.

The basic backpropagation algorithm performs the following steps:

*1. Forward pass:* Inputs are presented and the outputs of each layer are computed.

*2. Backward pass:* Errors between the target and the output are computed. Then, these errors are "back-propagated" from the output to each layer until the first layer. Finally, the weights are adjusted according to the gradient descent algorithm with the derivatives obtained by backpropagation.

The key point of basic backpropagation is that the weights are adjusted in response to the derivatives of performance function with respect to weights, which only depend on the current pattern; the weights can be adjusted sequentially or in batch mode. The asymptotic convergence rates of backpropagation is proved in [7]. Traditionally, the parity function has been used as an important benchmark for testing the efficiency of a learning algorithm. Empirical studies in [8] show that the training time of a feedforward net using backpropagation while learning the parity function grows exponentially with the number of inputs, thereby rendering the learning algorithm to be very time-consuming. Unfortunately, a satisfactory theoretical justification for this behavior is yet to be shown. Also, it is well known that the backpropagation algorithm may get stuck in local minima, and in fact, in general gradient descent algorithms may fail to classify correctly data that even simple perceptrons can classify correctly [9].

## IV. Need of Stability in Neural Networks

When working with neural network based control, stability is required for the neural network and the overall

International Journal of Electronics Communication and Computer Engineering
Volume 3, Issue 6, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

system. Stability criteria must be established for both for the controller and the controlled system. Vos, Valavani, and von Flotow [10] comment that a problem with neural network use is the lack of guaranteed stability for the weight update. They do not propose a stability guarantee but discount neural networks as a plausible controller because of the lack of stability.

Perfetti [11] developed a proof of asymptotic stability of equilibrium points. To characterize the local dynamic behavior near an isolated equilibrium point, it is sufficient to construct the Jacobian matrix of the linearization around the equilibrium and to check its eigenvalues. If all such eigenvalues have negative real parts, the equilibrium point is asymptotically stable. This approach, called Lyapunov's first method, is impractical for neural networks, as their order of complexity is usually very large. Perfetti showed through the use of Gerschgorin's disks that the slopes around an equilibrium point are all greater than zero, thus proving that the neural network at an equilibrium point is stable.

Renders, Saerens, and Bersini [12] proved the input-output stability of a certain class of nonlinear discrete MIMO systems controlled by a multi-layer neural network with a simple weight adaptation strategy. The stability statement is only valid, however, if the initial weight values are not too far from the optimal values that allow perfect model matching. The proof is based on the Lyapunov formalism. They proposed to initialize the weights with values that solve the linear problem. This research is an extension of Perfetti's paper, showing that, if there are no local minima between the initial weights and the global minimum, the weights will asymptotically converge on the global minimum.

A stability proof was developed for the linear perceptron, which limits the learning rate to one over the maximum eigenvalue of the system. No stability proof has been developed for the neural network that limits its learning rate. Perfetti showed that an equilibrium point is asymptotically stable. Renders and Bersini showed that the neural network will asymptotically converge on the global minimum if there are no other local minimums between the initial weights and the global minimum weights. The following section shows an optimized back propagation algorithm that provides stability in the NN.

## V. Proposed Algorithm

Let us consider the following unknown discrete-time nonlinear system:

$$y(k) = f[Xk] \qquad \text{------(1)}$$

where $Xk = [\, x1(k) \ldots, xi(k), \ldots, xN(k) \,]^T = [\, y(k-1), \ldots, y(k-n), u(k-1), \ldots, u(k-m) \,]^T \in \Re^{Nx1}$ $(N = n + m)$ is the input vector, $u(k-1) \in \Re$ is the input of the plant, $y(k) \in \Re$ is the output of the plant, and $f$ is an unknown nonlinear function $f \in C^{\infty}$. The output of the NN with one hidden layer can be expressed as

$$\hat{y}(k) = V_k \Phi_k = \sum_{j=1}^{M} V_{jk} \phi_{jk}$$

$$\Phi_k = [\phi_{1k}, \ldots, \phi_{jk}, \ldots, \phi_{Mk}]^T$$

$$\phi_{jk} = \tan h \left( \sum_{i=1}^{N} W_{ijk} x_i(k) \right) \qquad \text{----- (2)}$$

where $i = 1, \ldots, N$, $j = 1, \ldots, M$, $Xk \in \Re^{Nx1}$ is the input vector given by (1), $\hat{y}(k) \in \Re$ is the output of the NN, $V_k \in \Re^{1XM}$ and $W_k \in \Re^{MXN}$ are the weights of the output and the hidden layer of the NN, respectively, $W_{ijk} \in \Re$ ,$xi(k) \in \Re$, $\phi_k \in \Re^{1XM}$ , $\phi_{jk} \in \Re, V_{jk} \in \Re$.

Figure 3 shows the feedforward neural network:

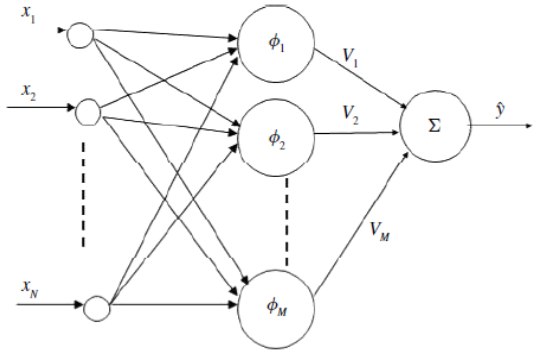


Fig.3. Architecture of the Neural Network

The proposed algorithm with a new time-varying rate is uniformly stable for online identification, the identification error converges to a small zone bounded by the uncertainty, and the weight error is bounded by the initial weights error.

Let us define the identification error $e(k) \in \Re$ as follows:

$$e(k) = \hat{y}(k) - y(k) \qquad \text{--------- (3)}$$

where $y(k)$ and $\hat{y}(k)$ are defined in (1) and (2), respectively.

The proposed algorithm uses a new time-varying rate as follows:

$$V_{jk+1} = V_{jk} - \alpha_k \phi_{jk} e(k)$$

$$W_{ijk+1} = W_{ijk} - \alpha_k \sigma_{ijk} e(k) \qquad \text{-------- (4)}$$

where the new time varying rate $\alpha k$ is

$$\alpha_k = \frac{\alpha_0}{2\left(\frac{1}{2} + \sum_{j=1}^{M} \phi_{jk}^2 + \sum_{j=1}^{M}\sum_{i=1}^{N} \sigma_{ijk}^2\right)}$$

where $i = 1, \ldots, N$, $j = 1, \ldots, M$.

The proposed algorithm is as follows:

1. Obtain the output of the nonlinear system $y(k)$ with (1). Note that the nonlinear system may have the structure represented by (1), the parameter $N$ is selected according to this nonlinear system.
2. Select the following parameters: $V1$ and $W1$ as random numbers between 0 and 1, $M$ as an integer number and $\alpha 0$

as a positive value smaller than or equal to 1. Obtain the output of the NN $\hat{y}(1)$ with (2).

3. For each iteration *k*, obtain the output of the NN $\hat{y}(k)$ with (2), obtain the identification error *e(k)* with (3), and update the parameters *Vjk*+1 and *Wi j k*+1 with (4).

4. Note that the behavior of the algorithm could be improved by changing the values of *M* or *α*0.

## VI. CONCLUSION

Neural networks are members of a family of computational architectures inspired by biological brains. Such architectures are commonly called "connectionist systems", and are composed of interconnected and interacting components called nodes or neurons. The term “feedforward” indicates that the network has links that extend in only one direction. Except during training, there are no backward links in a feedforward network; all links proceed from input nodes toward output nodes. The back-propagation algorithm is one of the most famous algorithm to train a feed forward network. This paper presents an optimized back propagation method to train feed forward neural networks. This optimized method provides stability in the NN.

## REFERENCES

[1] McClelland, J.L., Rumelhart, D.E., and Hinton, G.E., 1986. “The appeal of parallel distributed processing”, in Parallel Distributed Processing: Explorations in the Microstructure of Cognition - Foundations, Vol.1, MIT Press, Cambridge, pp.3-44.

[2] Luger, G.F., and Stubblefield, W.A. 1993. Artificial Intelligence: Structures and Strategies for Complex Problem Solving. 2nd Edition, Benjamin/Cumming Publishing, Redwood City, California.

[3] Winston, P.H., 1991. Artificial Intelligence. Addison-Wesley Publishing Co., Reading, Mass.

[4] Rich, E., and Knight, K., 1991. Artificial Intelligence. McGraw-Hill, New York.

[5] An introduction to neural computing. Aleksander, I. and Morton, H. 2nd edition.

[6] P. J. Werbos, Backpropagation through time: what it does and how to do it, in The Roots of Back-propation: From Ordered Derivatives to Neural Network and Political Forecasting, Wiley, New York, Wiley-Interscience Publication pp. 269-294 1994.

[7] G. Teasuro, Y. He and S. Ahmad, Asymptotic Convergence of Backpropagation, Neural Computation, 1 (1989), pp. 382-391.

[8] G. Teasuro and B. Janssens, Scaling Relationships in Back-Propagation Learning, Complex Systems, 2 (1988), pp. 39-44.

[9] M. Brady, R. Raghavan and J. Slawny, Backpropagation fails to separate where perceptrons succeed, IEEE Transactions on Circuits and Systems, 26 (1989), pp. 665-674.

[10] Vos, D. W., L. Valavani, and A. H. von Flotow, 1991, "Intelligent Model Reference Nonlinear Friction Compensation using Neural Networks and Lyapunov Based Adaptive Control," Proceedings of the 1991 IEEE International Symposium on Intelligent Control, pp. 417 – 422.

[11] Perfetti, R., 1993, "Asymptotic Stability of Equilibrium Points in Dynamical Neural Networks," IEEE Proceedings-G, Vol. 140, No. 6, pp. 401-405.

[12] Renders, Js. M., M. Saerens, and H. Bersini, 1994, "Adaptive neurocontrol of MIMO systems based on stability theory," IEEE International Conference on Neural Networks , pp. 2476-2481.

## AUTHORS PROFILE

**V. Sowjanya**
working as Asst. Professor, Department of Computer Science and Engineering, at Prasad V Potluri Siddhatha Institute of Technology, Kanuru, Vijayawada. Her research Interest includes Data Mining, Neural Networks and Network Security.

**J. Sirisha**
working as Asst. Professor, Department of Information Technology, at Prasad V. Potluri Siddhatha Institute of Technology, Kanuru, Vijayawada. Her research Interest includes Data Mining, Neural Networks and Network Security.

**T. Kranthi Kumar**
working as Asst. Professor, Department of MCA, at Prasad V. Potluri Siddhatha Institute of Technology, Kanuru, Vijayawada. His research Interest includes Data Mining, Neural Networks and Network Security.