Scientific Journal Impact Factor (SJIF) : 1.711
International Journal of Modern Trends in Engineering and Research
ISSN (ONLINE) : 2349 - 9745 ISSN (PRINT) : 2393 - 8161

Effect of changing hidden neurons and activation function on Back-Propagation (BP) Speed

Rasha M. Hassoon[1], Safana H. Abbas[2]

[1]Department of computer science-College of Education-University of al-mustansiriya
[2]Assist. Prof., Department of computer science-College of Education-University of al-mustansiriya

Abstract—The Back-Propagation (BP) is the best known and widely used learning algorithm in training multiple neural network. A vast variety of improvements to BP algorithm have been proposed since ninety's. in this paper, the effects of changing the number of hidden neurons and activation equation are investigated. According to the simulation results, the convergence speed have been improved and become much faster by the previous two modifications on the BP algorithm.

Key words—Back-Propagation, Neural Network, hidden neurons.

I. INTRODUCTION

Back-propagation (BP) is the most widely used learning in the neural network and can apply to realize standard logic gates and different Boolean functions. Truth table and initial weight values are all the requirements to implement any logic function using NNs [6]. There are two method, the first method talk about the number of neurons in the hidden layer that should be equal to the square number of input , and the second method talk about that the hidden neuron should be equal to the 2 multiply the number of input reduce one.

II. ARTIFICIAL NEURAL NETWORKS (ANN)

An ANN consists of number of very simple and highly inter connected processors, which are analogous to the biological neuron. The neurons are connected by weighted links passing signals from one neuron to another. Each neuron receives a number of input signal is transmitted through the neuron outgoing connection. The outgoing connection splits in to a number of branches that transmit the same signal. Neurons connected by links, each link has a numerical weights associated with it. Weights are the basic means of long-term memory in ANNs [3]. ANNs are at the forefront of computational systems designed to produce, or at least mimic, intelligent behavior. Neural Network Systems have successfully been developed and deployed to solve pattern recognition, capacity planning, business intelligence, robotics, or intuitive problem related aspects. In computer science, neural network gained a lot of stream over the last few years in areas such as forecasting data analytics, as well as data mining [5].

The function of the neuron is described by the following equations [3].

$$yj = fj(x) \quad (1)$$

and

$$x = \sum_{i}^{n} Pi\ Wij + bj \quad (2)$$

Where p_i be i^{th} the inputs of the system, w_{ij} is the weight in the i^{th} connection and b_j . Applying a transfer function f_j to the summation of all signals from each connection p_i [3].

III. BACK-PROPAGATION ALGORITHM (BP)

The BP neural network is multilayered, feedforward neural network and is by far the most extensively used. It is also considered one of the simplest and most general methods used for supervised training of multilayered networks. BP works by approximating the non-linear relationship between the input and the output by adjusting the weight values internally [4]. Generally, the BP network has two stages, training and testing, in his thesis will speed up the training stage only. The following figure shows the topology of the BP neural network that includes input layer, one hidden layer and output layer. It should be noted that BP neural networks can have more than one hidden layer. The following Pseudo coding describes the BP algorithm [1][2]:

```
Assign all network inputs and outputs:
Initialize all weights with small random numbers typically between -1 and 1
Repeat
    For every pattern in the training set
        Present the pattern to the network
//      propagated the input forward through the network:
        For each layer in the network
            For each node in the layer
                1. Calculate the weight sum of the inputs to the node.
                2. Add the threshold to the sum.
                3. Calculate the activation function for the node
            End
        End
//      propagate the errors backward through the network
        For every node in the output layer calculate the error signal
        End
        For all hidden layers
            For every node in the layer
                1. Calculate the node's single error.
                2. Update each node's weight in the network.
            End
        End
//      calculate global error
        Calculate the error function
        End
While ((maximum number of iterations < than specified) and
        (Error function is > than specified))
```

IV. FLOW CHART OF PROPOSED METHOD

The following flow chart illustrates the methodology that has been used to speed up the standard BP:

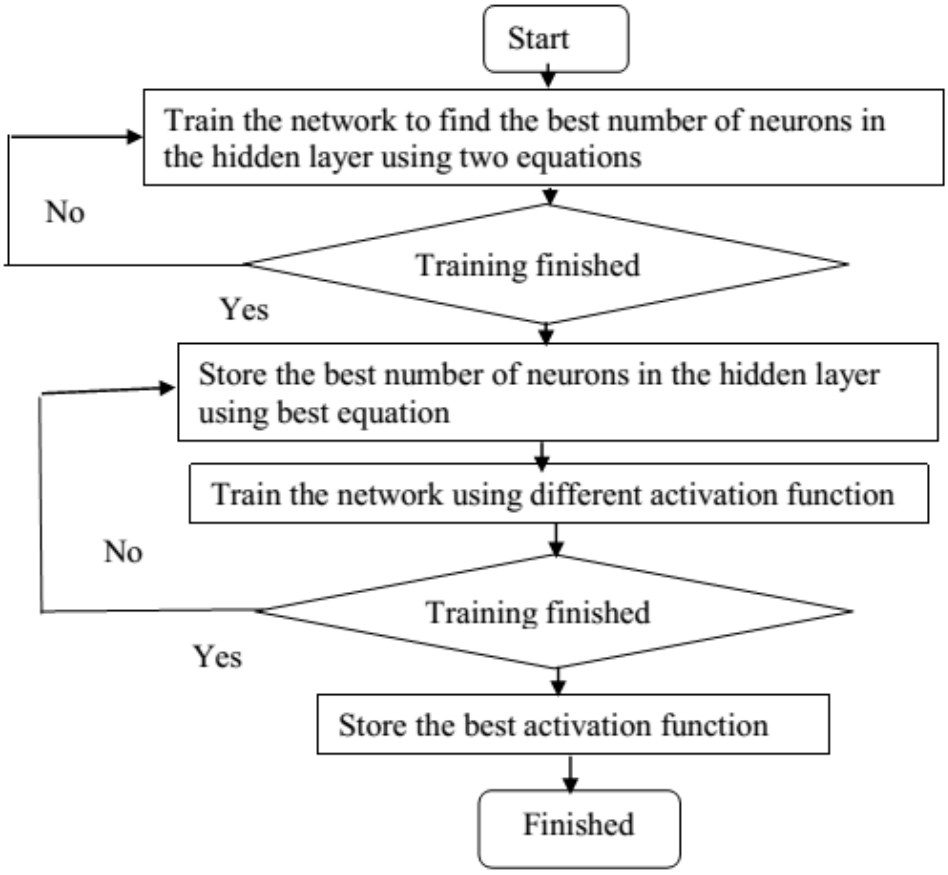

Figure (1): the modification in BP standard algorithm

4.1 Architecture of network:

In this part the number of hidden neurons will be changed to observe their effect on the BP algorithm speed. Two different equations will be used to calculate the number of hidden neurons, to make the BP algorithm faster, as shown in the equation (3) and equation (4).

$$h = 2n - 1 \qquad (3)$$

$$h = n^2 = 4 \qquad (4)$$

Variable input XOR logic function are trained using BP algorithm.

The random selection of a number of hidden neurons might cause either overfitting or under fitting problems. That is mean, if there are too many hidden neurons, that would be a low training error, but there would be high generalization error due to over fitting.

On the other hand, if there were too few hidden neurons, a high training error and high generalization error would result due to under fitting.

To verify the effect of changing the number of hidden neurons on the convergence speed, the following experiments were made:

1. Selecting an input pattern (fixed input logic function).
2. Using equation (3) to select the best number of hidden neurons.
3. Replacing equation (3) by equation (4).
4. Using trial and error to determine the best number of hidden neurons.

5. Changing the number of input patterns and repeating steps (2,3,4).

Applied different number of input to XOR problem, to test the previous steps:

1. 2-input XOR with different value of hidden neurons are used ,as shown in figure(2)

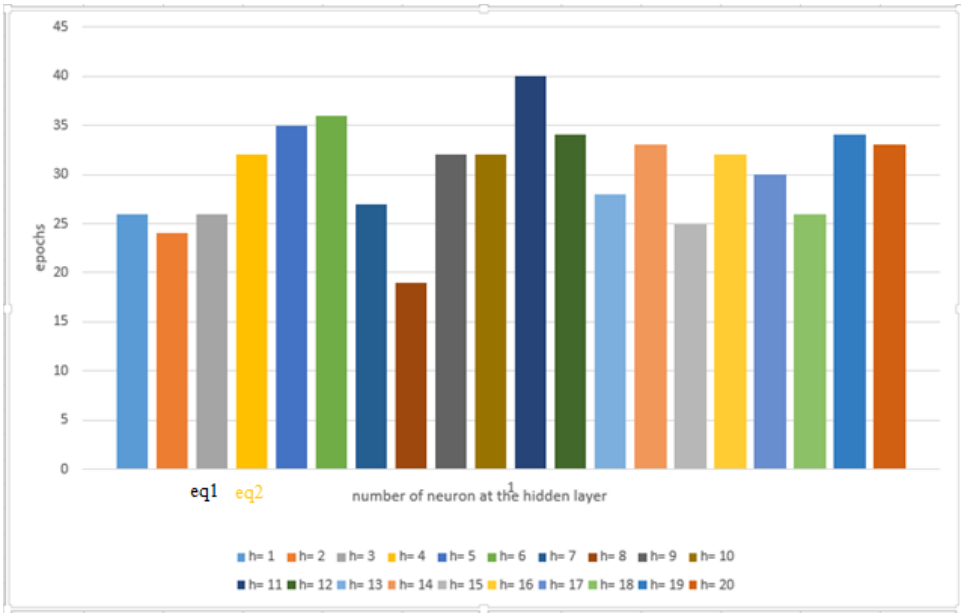

Figure (2) changing the number of hidden neurons with 2-input XOR

It is obviously clear from the figure (3.8) that the lower number of hidden neurons is obtained by using trial and error =8.

2. 3-input XOR with different value of hidden neurons are used, as shown in figure (3)

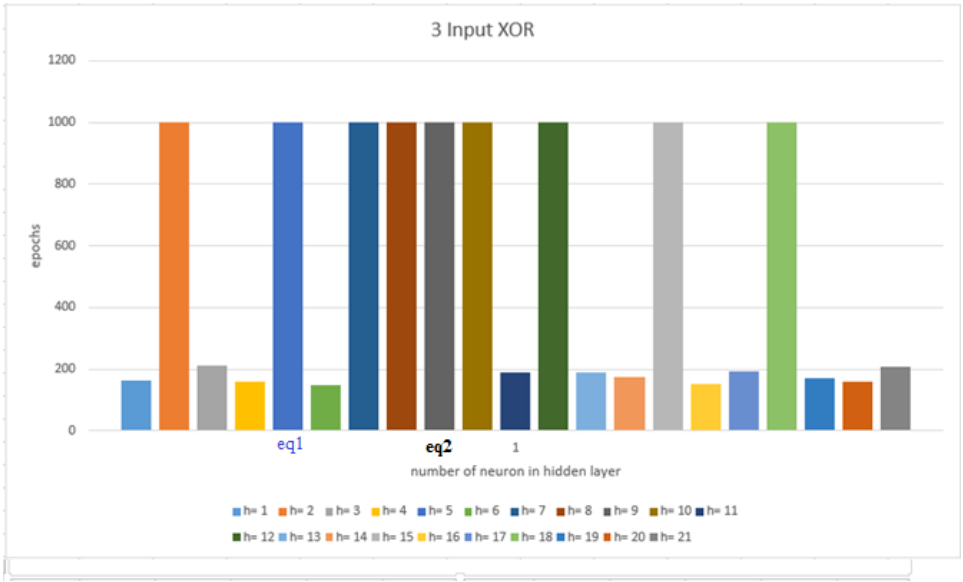

Figure (3) changing the number of hidden neurons with 3-input XOR

It's obviously clear from the figure (3) that the lower number of hidden neurons is obtained by using trial and error =6.

3. 4-input XOR with different value of hidden layer ,as shown in figure(3.10)

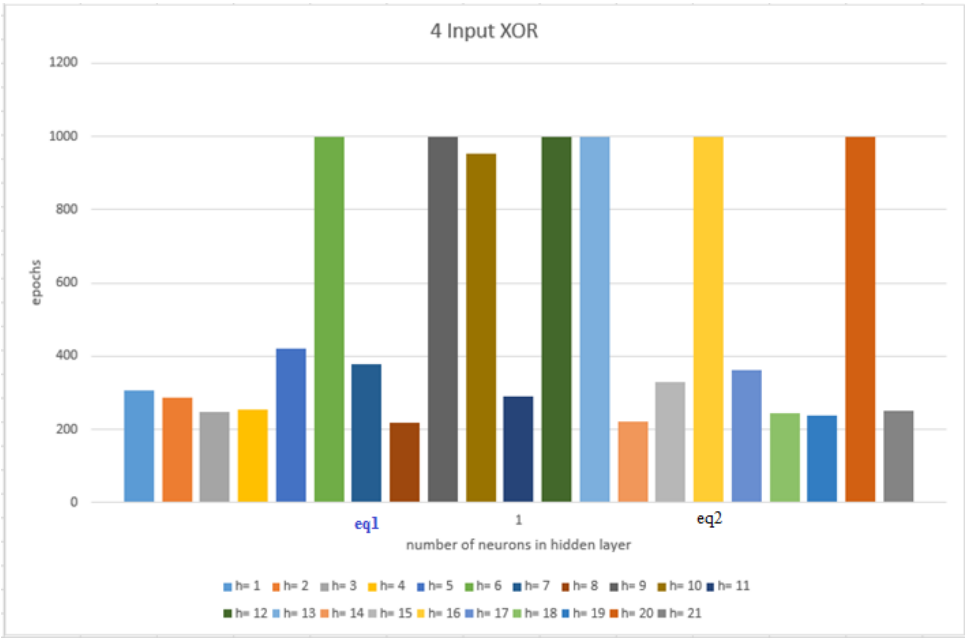

Figure (4) changing the number of hidden neurons with 4-input

It's obviously clear from the figure (4) that the lower number of hidden neurons is obtained by using trial and error =8.

4. 5-input XOR with different value of hidden layer ,as shown in figure(5)

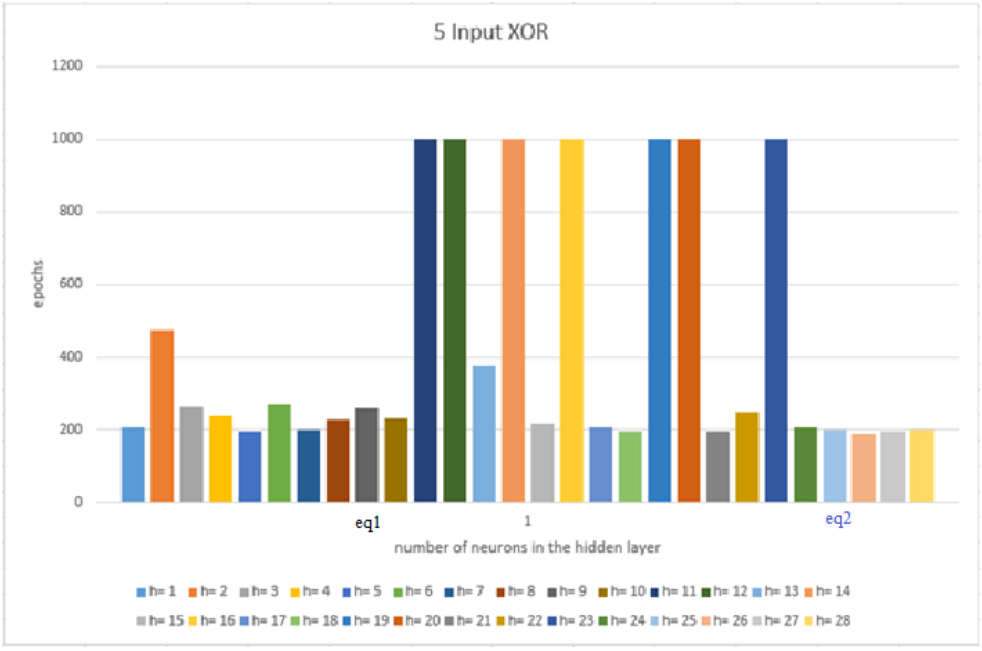

Figure (5) changing the number of hidden neurons with 5-input XOR

It is obviously clear from the figure (5) that the lower number of hidden neurons is obtained by using trial and error =26.

5. 6-input XOR with different value of hidden layer ,as shown in figure(6)

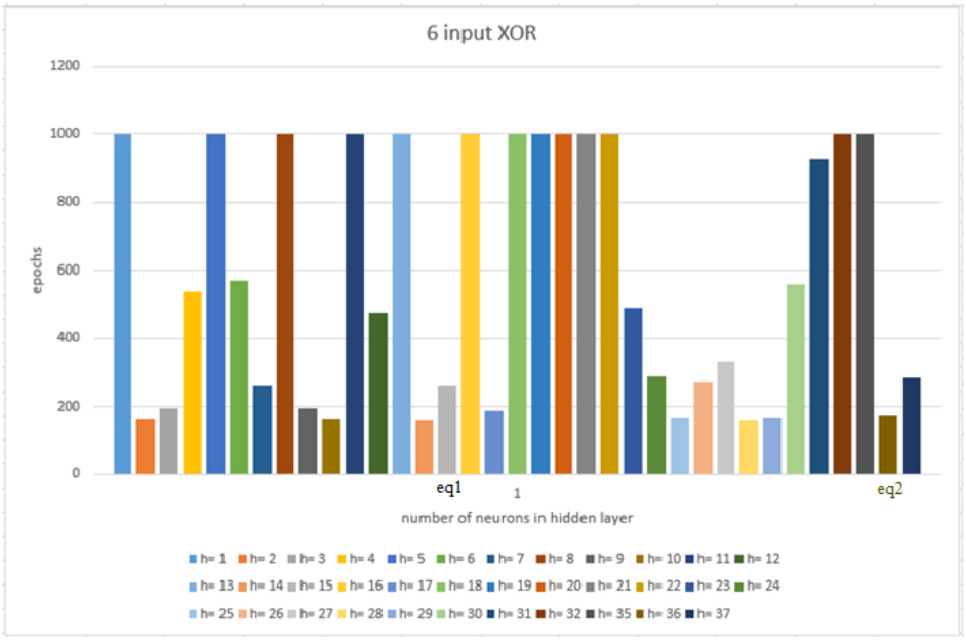

Figure (6) changing the number of hidden neurons with 6-input XOR

It's obviously clear from the figure (6) that the lower number of hidden neurons is obtained by using trial and error =19.

The XOR problem		
N	Number of neurons in hidden layer	Epochs
2	$h = 2n - 1$=3	26
	$h = n^2 = 4$	32
	Less number h=8	19
3	$h = 2n - 1$=5	1000
	$h = n^2 = 9$	1000
	Trial and error h=6	150
4	$h = 2n - 1$=7	377
	$h = n^2 = 16$	1000
	Trial and error h=8	219
5	$h = 2n - 1$=9	259
	$h = n^2 = 25$	199
	Trial and error h=26	189
6	$h = 2n - 1$=15	263
	$h = n^2 = 36$	176
	Trial and error h=14	160

Table (1): number of neurons in hidden layer effect in number of epochs

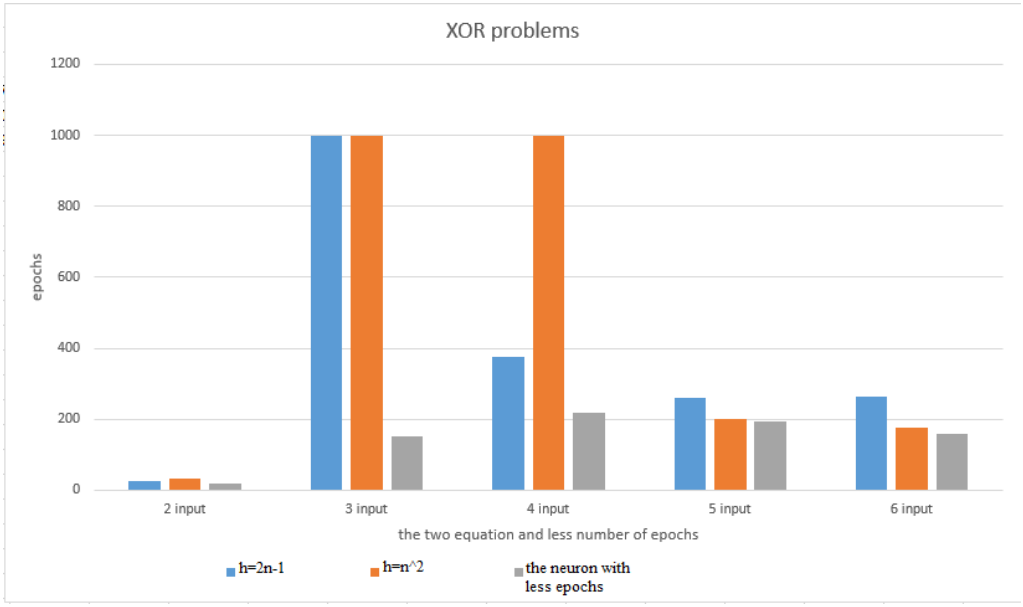

Figure (7): number of neurons in hidden layer effect in number of epochs

The conclusion from table (1) and figure (7) is that when the number of inputs neurons is high, it is better to use equation (4). And when the number of inputs neurons is low, it is preferred to use equation (3). If we have time its prefer to use try and error to get best number of hidden neurons.

4.2 Activation function:

By implementing the following different type of activation functions: (Binary step function, bipolar step function, Bipolar Sigmoidal function, Binary Sigmoidal function, and Ramp function), table (2) shows the number of epochs for each function:

Activation function	Number of epochs
Identity function: F(x)=x	10000
Binary step function: $F(x) = \begin{cases} 1 & x > 0.5 \\ 0 & x \leq 0.5 \end{cases}$ (5)	10000
Bipolar step function: $F(x) = \begin{cases} 1 & x > 0.5 \\ 0 & x = 0 \\ -1 & x < 0.5 \end{cases}$ (6)	10000
Binary Sigmoidal function: **Rang from [1 0]** $y = 1 + exp(-x))^{-1}$ (7)	10000
Bipolar Sigmoidal function: **Rang from [1 -1]** $y = \tanh(x)$ (8)	54
Ramp function: $F(x) = \begin{cases} x & x \geq 0 \\ 0 & x < 0 \end{cases}$ (9)	10000

Table (2): number of epochs for different activation functions

Test different activation functions and find bipolar sigmoidal activation function is the best because it have the less number of epochs as shown in figure(8).

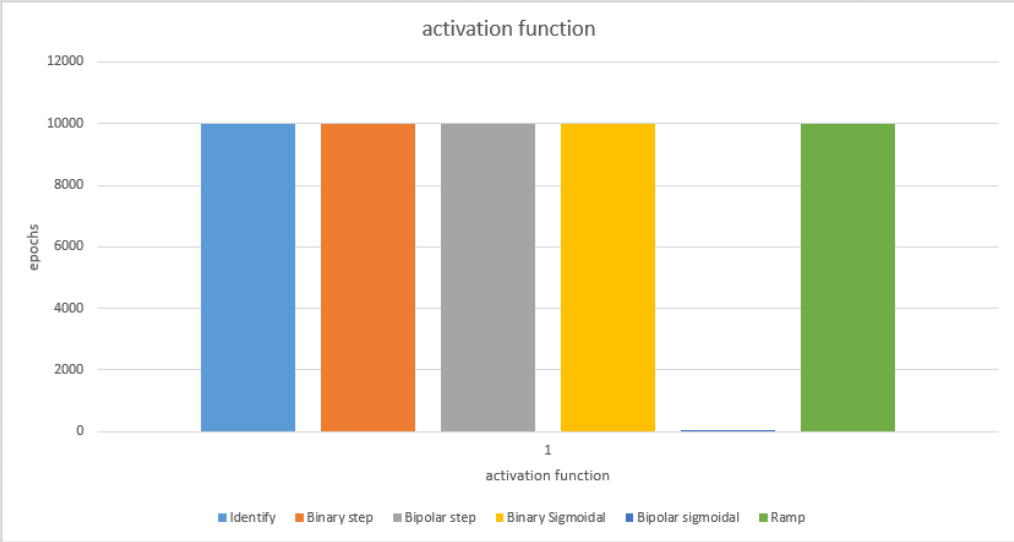

Figure (8) different activation functions

V. CONCLUSION

In this paper, an improved, an improved back propagation algorithm. Which is simple and generic to train feed-forward artificial neural networks on 2-bit XOR benchmark problems. There are many modifications first find that the number of hidden neurons is when the number of inputs neurons is high, it is better to use square the number of input, And when the number of inputs neurons is low, it is preferred to use two multiply the number of input and minimize one. If we have time it's prefer to use try and error to get best number of hidden neurons. Second train the network with different activation functions and find which activation function is make BP algorithm faster is Bipolar Sigmoidal activation funstion. Use the two the parameter to make the algorithm that faster than the standard BP algorithm.

REFERENCES

[1] Mitchell, T. M. , **Machine Learning** , chapter 4, 1997, Mc Graw-Hill.
[2] Donald R. Tveler , **The Backprop Algorithm**, chapter 2, 2001.
[3] Muthana Khallil Ibrahim, A Hybrid Artificial Neural Networks and Swarm Intelligent for Medical Image Recognition Based on FPGA. University of Technology, Iraq, 2011.
[4] Michael A. Arbib, **The hand Book of Brain Theory and Neural Networks**. Second edition, A Bradford Book THE MIT PRESS, Cambridge, Massachusetts, London, England, 2002.
[5] Dominique A. Heger, An Introduction to Artificail Neural Networks (ANN). DH Technologies, 2011.
[6] Dilip Sakar, **Methods to speed up Error Back-Propagation learning algorithm**. University of Miami, ACM Computing Surveys, Vol. 27, No. 4, December, 1995.

