Research Paper E-ISSN No : 2455-295X | Volume : 2 | Issue : 7 | July 2016

CHARACTER RECOGNITION BY ARTIFICIAL NEURAL NETWORK METHODS

Abdulrahim M. Ahmad | Sahar Mahdie Klim [1] | Maab Alaa Hussain

[1] College of Engineering -Misan University.

ABSTRACT

Within the domain of computer science, numerous objectives arise. Automatic processing of data is imperative in many fields and for this purpose a number of techniques are implemented; one of which is Artificial Neural Network. Artificial intelligence is concerned with building an intelligent machine, whereas artificial neural network is after creating a learning system which is based on little and simple framework program capable of responding to problem and getting feedback on how it does so. Based on the human brain principle, the artificial neural network is built in such a way that makes it capable of learning things that will enable it later to solve problems according to the bases information it has learned. It is also capable of solving adaptive problems due to its ability to conduct adaptive learning. The current paper study several properties where the human nervous system and the artificial neural network are compared. Also an application for the recognition of scanned handwritten characters by artificial neural network is employed.

KEYWORDS : Neural Network, Artificial Neural Network, Back propagation, Bayesian Artificial Neural Network.

1- INTRODUCTION

Neural Network: Definition

Realizing that the brain computes in a way entirely different from that of conventional digital computer helped inspire the work on artificial neural networks, also known as neural networks, right from the beginning. [1]

Since artificial neural networks are an attempt at modeling the information processing capabilities of nervous systems, it is necessary that we initially tackle the essential properties of biological neural networks concerning information processing. By so doing, we are able to design abstract models of artificial neural networks, then simulate and analyze them [2].

Based on the human brain principle, ANNs are meant to function as similar to real human brain as possible [3]. For instance, through an example an ANN learns what to. An ANN is built and configured to serve a specific application or task [4]: data classification, pattern recognition applications, to name just a few.

Table1: Comparison between computer and brain neural network [5]

	Human Brain	Computer
processing units	About 1011	About 109
Type/name of the processing units	Neurons	Transistors
Method of calculation	Mostly parallel	Mostly in serial
Possible switching operation	About 1013 s-1	About 1018 s-1
Actual switching operation	About 1012 s-1	About 1010 s-1
Switching time	About 10-3s	About 10-9s
Data storage	Associative	Addressed-based

This paper presents a survey of artificial neural network methods. First the theoretical studies are introduced including mathematical aspect followed by illustration of the definitions and properties. The work is in the form of an intelligent system based on the neural network that is built to recognize scanned English characters and save them with extension (.bmp). The study comprises a number of methods used for that purpose.

2- LITERATURE SURVEY

This section provides a survey of some essential writing on green registering within the accompanying subsections:

- Pontus and Forslund [6], in 2003, presented an EEG based BCI that is designed for automatic classification of two dimensional hand movements. This aims at building an intuitive communication system that is to be operated by people who suffer from severe motor impairments. This experiment yielded results showing that the EEG signals in effect contain extractable and classifiable information concerning the performed movements, during physical and imagined movements alike.

- In their study of the effectiveness of cross validation, Bayesian regularization, early stopping, and bagging to mitigate overfitting and improve generalization for pricing and hedging derivative securities, Ramazan Gencay and Min Qi's [7] in 2001 investigation marked the ability of Bayesian regularization to generate pricing and delta-hedging errors considerably smaller than the baseline neural-network (NN) model and the Black-Scholes model for some years.

3- ARTIFICIAL NEURAL NETWORK:

3.1 Biological Neural Network:

The components of the biological neural network are nerve cells (neurons), which are interconnected in the way shown in Fig. 2. The neuron's nucleus is located in the cell body of the neuron where most of the neural\computation" occurs. The passage of neural activity from one neuron to another takes place by means of electrical triggers which, in turn, travel between cells down the neuron's axon via an electrochemical process of voltage-gated ion exchange along the axon as well as diffusion of neurotransmitter molecules through the membrane over the synaptic gap. The function of an axon resembles that of a connection wire except that the mechanism of signal flow is not through electrical conduction but rather through charge exchange that is transported via diffusion of ions. Passing by the neuron's cell, down the axon and then through synaptic junctions at the end of the axon through a very narrow synaptic space to the dendrites and/or soma of the next neuron, this transportation process occurs at an average rate of 3 m/sec.

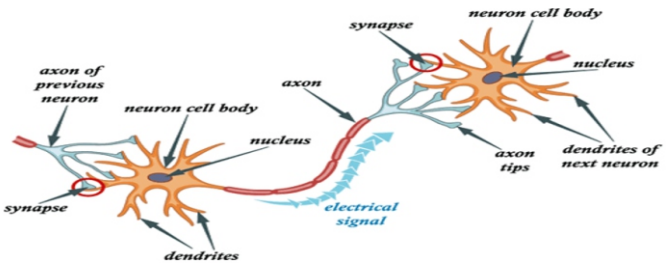


Figure (1) Interconnection of biological neural nets.

It is worth noting that interconnections are not all equally weighted since some receive higher priority (a higher weight) than others, some are exciter and some are inhibitory (serving to block transmission of a message). This variation results from differences in chemistry and the existence of a chemical transmitter and modulating substances inside and close to the neurons and the axons, and in the synaptic junction. This characteristic interconnection between neurons and weighting of messages is essential to artificial neural networks as well (ANNs) [8].

3.2 Structure of an Artificial Neural Network:

The definition that is provided in the current section as an overview of the general structure of neural network will be the basis of the work in the following sections.

3.2.1 Topology of Neural Network

Neural network topologies are classified according to interconnection and are arranged within layers; this yields two well-known topologies:

- Feed Forward Topology
- Recurrent Topology

3.3.2 Feed forward network:
The nodes in this topology follow a hierarchical order into layers with the input layers as the start and the output layers as the end, Fig. (2). The computational power of a network is provided chiefly by the number of hidden layers in-between the input and output layers. Connection between nodes in successive layers is provided via uni-direction paths that begin from one layer (source) and end at the succeeding layer (sink). The output of one layer feeds the nodes of the next layer in a forward direction and prevents feedback flow of information in the structure. {Application multilayer layer perception network and radial basic function network [9].

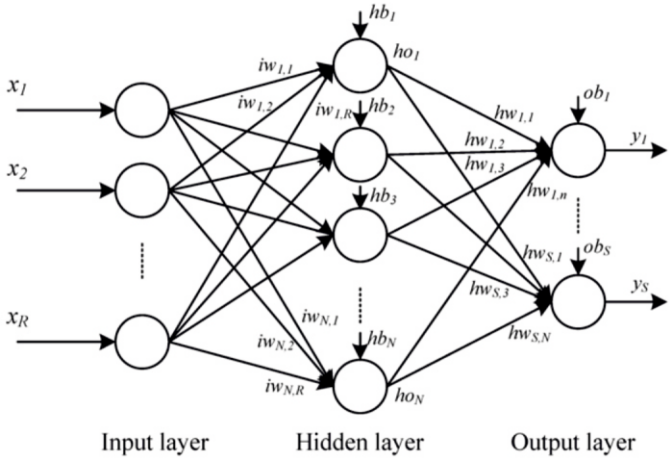


Figure (2) feedforword neural network [9]

3.3.3 Recurrent Network:
Input or output neurons are not specified in recurrent networks, rather direct recurrent network and feed forward network are found with shortcuts [10].

- ***A Direct recurrent network*** contains a neuron j, with the weight w_{jj}, that is connected to itself, as shown in Fig. (3). This entails that the diagonal in the Hinton diagram is different from zero; whereas, elsewhere it is zero.

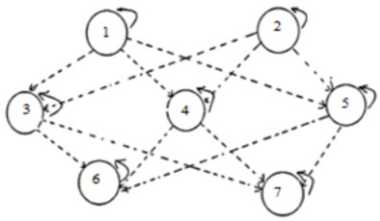
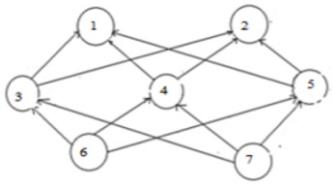


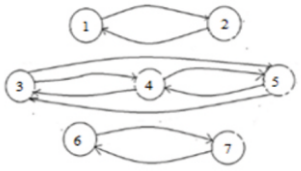
Figure (3) Direct re-current neural network

- ***An Indirect recurrent network*** is based on the feed forward principle where connections between neurons and their preceding layer are provided, as shown in Fig. (4).



Figure(4) Indirect re-current network

- In the case of ***Lateral recurrence network***, connections are permitted between neurons that belong to the same layer as in Fig. (5).



Figure(5)Lateral recurrence network

4- METHODS OF NEURAL NETWORK: GRADIENT DESCENT, BACK PROPAGATION AND BAYESIAN REGULARIZATION
The focus in this section will be on a type of neural network called back propagation neural network. Back-propagation networks are the most widespread neural network model that has so far been the focus of attention on research among the existing models altogether.

4.1 Perceptron
4.1.1 Single layer perceptron
Viewed as the simplest neural network, the perceptron typically has an input layer made of a vector of real value (neurons). It computes a combination of the input linearly and sends the value 1 as output if the linear combination result is greater than a fixed threshold, or -1 otherwise [11].

Consider a perceptron with n input values $x_1,x_2,\ldots,x_n$, its output o $(x_1,x_2,\ldots,x_n)$ can be computed and represented as:

$$O(x_1,x_2,\ldots,x_n)=\begin{cases}1 & \text{if } w_0+w_1x_1+,\ldots,+w_nx_n>0\\ -1 & \text{otherwise}\end{cases} \quad (1)$$

w_k, being a real-value constant called weight, determines the "weight" of contribution of the variable x_k in the linear combination [12].

Definition 1: (of input neuron). It is like an identity function, also called identity neuron. It is responsible for forwarding the exact received information.

Definition 2: A single layer perceptron (SLP), is made of two layers, one is the input layer and the second the output layer. It was previously mentioned that a perceptron has a unique output, however, aSLP is admitted because it can be seen as an association of different perception with the same input layer [13].

4.1.2 Perceptron training rule
To provide a further survey of complex networks, an examination of the rule used by perceptron for its learning problem is now in order. The question to be posed here is how to figure out the best weight vector that enables the perceptron to produce the right values -1 or 1 as output. Different ways are suggested in an answer to this question; among which two methods are examined: namely the "Delta rule" and the "perceptron rule". Their algorithms converge to acceptable hypothesis under some conditions and are, therefore, significant for artificial neural networks [14].

Below are the steps of learning algorithm:
1- Select between (0-1).

Where is the learning rate.

2- Setup the learning patterns, each of which consists of a sequence of $x_1,x_2,\ldots,x_n$ and is accompanied by a category information y^d.

3- Select random values for initial weights $w_0,w_1,\ldots,w_n$.

4- Select the pattern x from the training set randomly for each iteration t.

5- Calculate the activation value of a of perceptron unit.

X_0 always equal to 1.

$$a=\sum_{0}^{n} w_i x_i$$

6- Calculate the output y :

$$y=\begin{cases}1.....if\, a\geq 0\\ 0.....otherwise\end{cases}$$

7- In case an error occurs for this pattern, update the weight and bias.

If $y \neq y^d$
$W_i(new) = w_i(old) + y^d x_i$
$b_i(new) = b_i (old) + y^d$

else

$W_i(new) = w_i(old)$
$b_i(new) = b_i (old)$

8- Stop if the weight does not change, else go to step 4.

4.2 Back propagation:
4.2.1 Back propagation algorithm
The back propagation is based on the equation Eq.2

$$E(\vec{w})=\frac{1}{2}\sum_{i=1}^{M}(t_{kd}-o_{kd})^2 \quad (2)$$

where t_{kd} denotes the desired output, and O_{kd} is the output computed by the network [15]. The back propagation algorithm is defined as follows:

1. Initialize w by setting all weights to small random numbers

2. Present a training pattern to the input layer and calculate the outputs from all nodes in the network in a feed-forward way according to Eq.3.

$$y = \frac{1}{1 + e^{-\beta \sum_{i=1}^{n} w_i x_i}} \quad (3)$$

3. Adjust the weights in the output layer:

$$w_{ij}^{(p+1)} = w_{ij}^{(p)} - \eta\beta y_j^2(1 - y_j)(y_j - d_j) + \alpha(w_{ij}^{(p)} - w_{ij}^{(p-1)}) \quad (4)$$

where η is the learning rate, β the spread of the threshold function and α the momentum term. The actual output from node j is denoted yj and the desired output dj.

4. Work the way backwards through the network, and update the rest of the weights according to the back-propagation rule

$$w_{ij}^{(p+1)} = w_{ij}^{(p)} - \eta\beta y_j^2(1 - y_j)\sum_k \delta_k w_{jk} + \alpha(w_{ij}^{(p)} - w_{ij}^{(p-1)}) \quad (5)$$

The error term δ_k is the error of the successor node k given by:

$\delta_k = \beta y_k(1 - y_k)(y_k - d_k)$ For output nodes,

and:

$\delta_k = \beta y_k(1 - y_k)\sum_m \delta_m w_{jm}$ for the nodes in the hidden layers.

5. Repeat from 2 until result is good enough [6].

In this algorithm, the variable symbol δ_n stands for the error term associated with the unit *n*.

4.3 Resilient Back Propagation Algorithm (RBP):
The algorithm RBP is a local adaptive learning scheme that executes supervised batch learning in feed – forward neural networks. RBP basically eliminates the harmful influence of the size of the partial derivative on the weight step. Accordingly, only the sign of the derivative is taken to indicate the direction of the weight update. The algorithm acts on each weight aside. For each weight, in case of sign change of the partial derivative of the total error function compared to the last iteration, the update value for that weight is multiplied by a factor η−, where $0 < \eta^- < 1$. If the last iteration produces the same sign, the update value is multiplied by a factor of η+, where $\eta^+ > 1$. The above mentioned manner is used to calculate update values for each weight. Ultimately, each weight is changed by its own update value, in the opposite direction of that weight's partial derivative. This minimizes the total error function. η+ is empirically set to 1.2 and η− to0.5.

In order to attain this, we introduce for each weight w_{ij} an individual update-value $\nabla_{ij}(t)$ which slowly determines the size of the weight update. Being introduced as a second learning rule, it determines the evaluation of update-value $\Delta_{ij}(t)$. The observed behavior of the partial derivative during two successive weight updates by Eq.6. forms the basis of this estimation.

$$\nabla_{ij}(t) = \begin{cases} \eta^+ . \Delta_{ij}(t-1), if \frac{\partial E}{\partial w_{ij}}(t) . \frac{\partial E}{\partial w_{ij}}(t-1) > 0 \\ \eta^- . \Delta_{ij}(t-1), if \frac{\partial E}{\partial w_{ij}(t)} . \frac{\partial E}{\partial w_{ij}}(t-1) < 0 \\ \Delta_{ij}(t-1), otherwise \end{cases} \quad (6)$$

Where $0 >> 1^+$

A clarification of the adaptation rule derived from the above formula is due here. Evidently, whenever the partial derivative of the equivalent weight w_{ij} varies its sign (which indicates that the last update was large in magnitude and the algorithm has skipped over a local minima) the update - value $\Delta_{ij}(t)$ is decreased by the factor η−. If the derivative holds its sign, the update-value will increase to an extent to speed up the convergence in shallow areas. When, for each weight, the update-value is settled in, the weight-update itself follows a very simple rule: when the derivative is positive, the weight is decreased by its update value; when negative, the update-value is added as shown in Eq.7.

$$w_{ij}(t) = \begin{cases} -\Delta_{ij}(t), if \frac{\partial E}{\partial w_{ij}}(t) > 0 \\ \Delta_{ij}(t), if \frac{\partial E}{\partial w_{ij}}(t) < 0 \\ 0, else \end{cases} \quad (7)$$

$W_{ij}(t+1)=w_{ij}(t)+w_{ij}(t)$

The mere exception here is that if the partial derivative changes sign, which means that the previous step was too large and the minimum was missed, it follows that the previous weight update is reverted by Eq.8:

$$\Delta w_{ij}(t) = \Delta w_{ij}(t-1), if \frac{\partial E}{\partial w_{ij}}(t) . \frac{\partial E}{\partial w_{ij}}(t-1) < 0 \quad (8)$$

The derivative changes its sign, once more, in the following step on account of that "backtracking" weight-step. Avoiding double punishment of the update value requires the absence of adaptation of the update-value in the succeeding step which may be done in application by setting in the $\Delta_{ij}(t)$ update rule above.

$$\frac{\partial E}{\partial w_{ij}}(t-1) = 0$$

The partial derivative of the total error is given by:

$$\frac{\partial E}{\partial w_{ij}}(t) = \frac{1}{2}\sum_{p=1}^{p} \frac{\partial E_p}{\partial w_{ij}}(t)$$

So the weights are updated only after all training patterns are presented, as the partial derivative of the error must be accumulated for all P training patterns [16].

4.4 Bayesian Artificial Neural Network (BANN):
This network is capable of converting a non-linear problem into a statistical problem with linear regression formulation [17]. BNNs mainly model the relationship from the data without overfitting the noise by optimizing the regularization parameter. A disadvantage of ANNs, as compared to conventional neural network, lies in choosing their optimal architecture based on trial and error. BBNs, on the other hand, control model complexity automatically by estimating effective parameters of the network.

4.4.1 Maximum Likelihood (ML) – Maximum A Posteriori (MAP)
Below are some statistical concepts on which properties of the Bayesian artificial neural network is based[12].

The ML sets the parameter ɵ for a given data v using Eq.9:

$$\theta^{ML} = \arg\max_{\theta} P(v/\theta) \quad (9)$$

Uses the setting ɵ which maximizes the distribution a posteriori of the parameter ɵ. The MAP:

$$\theta^{MAP} = \arg\max_{\theta} P(v/\theta)P(\theta) \quad (10)$$

According to Eq. (10), p(ɵ) denotes the prior distribution .The posteriori stands for our beliefs regarding the range of possibilities and their associated credibility [18].

The term "maximum likelihood" stands for the ɵ parameter which apparently belongs to the model that most likely generates the observed data.

if $p(\theta) = const$

then the MAP is equivalent to the ML setting……. (For a given 'flat' prior, this means) The logarithm function is a strictly increasing function. It follows that for a given Function $f(\theta)$:

$$\theta_{opt} = \arg\max_{\theta} f(\theta) \Leftrightarrow \theta_{opt} = \arg\max_{\theta} \log f(\theta) \quad (11)$$

It is noteworthy that when optimizing the MAP or its logarithm, this can help find the MAP parameters.

$$\log p(\theta/v) = \log p(v/\theta) + \log p(\theta) - \log p(v) \quad (12)$$

It is convenient to use the logarithm likelihood since normally under the i.i.d assumptions the summation of data terms holds as in Eq. 13:

$$\log p(\theta/v) = \sum_n \log p(v^n/\theta) + \log p(\theta) - \log p(v) \quad (13)$$

4.4.2 Maximum likelihood (ML) training of belief networks

Consider the following model that represents the relationship of exposure to asbestos (a), being a smoker (s) and the incidence of lung cancer (c)

$$P(a, s, c)=P(c\,a, s)\,P(a)\,P(s) \quad (14)$$

The paragraph and figure below depict the problem:
Each variable is binary, dom(a) = {0,1} , dom(s) = {0,1}, dom(c) = {0,1}. It could be assumed that there is no direct relationship between Smoking and exposure to Asbestos. This sort of assumption is what we may be able to elicit from medical experts. In addition, it could also be assumed that there is a list of patient records, each row represents a patient's data. We can learn the table entries p(ca , s) by counting the number of c is in state 1 for each of the 4 parental states of a and s:

$$p(c=1a=0, s=0)=0, p(c=1a=0, s=1)=0{:}5$$

$$p(c=1a=1, s=0)=0.5, \quad p(c=1a=1, s=1)=1$$

In a similar way, based on counting, p(a = 1) = 4=7, and p(s = 1) = 4=7. Thus, these three CPTs complete the full distribution specification [20].

4.4.3 Bayesian Belief Network (BBN) Training:

In this section we use the case study with variables (a), (s) and (c). The use of the BBN is an alternative to the use of ML and MAP. In the previous section the cancer scenario was defined:

$$P(a, s,c)=P(ca, s)\,P(a)\,P(s)$$

where only the independence was specified. Now let's presume that the entries of the table are also taken into account. Consider a set $V=\{(a^n, s^n, c^n); n=1,\ldots,N\}$ of visible observations. Our aim here is to learn the suitable and appropriate distribution for the entries [19].

4.4.4 Independence of Local and Global parameters:

In Bayesian learning of BNs, there is a need to specify a prior on the joint table entries. Since dealing with multi-dimensional continuous distributions is generally computationally problematic, it is found practicable to specify only uni-variate distributions in the prior. As we will see below, this has a pleasing consequence that for i.i.d. data the posterior also factorizes into uni-variate distributions.

A convenient assumption ***for Global parameter independence*** is that the prior factorizes over parameters. So, for our Asbestos, Smoking, Cancer example, we assume

$$P(\theta_a, \theta_s, \theta_c)= P(\theta_a)\,P(\theta s)\,P(\theta)$$

The joint model below is obtained by assuming that the data is i.i.d

$$P(\theta_a, \theta_s, \theta_{c,v})= P(\theta_a)\,P(\theta_s)\,P(\theta_c)\prod_n P(a^n/\theta_a)\,p(s^n|\theta_s)P(c^n|s^n, a^n, \theta_c)$$

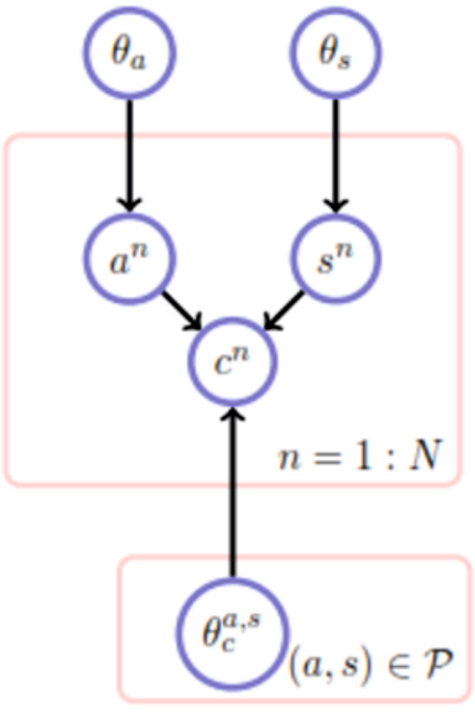


Figure (6). Belief Neural Network

Consider the Belief Neural Network of Fig.10, the learning process then corresponds to the inference of:

$$P(\theta_a, \theta_s, \theta_c|v) = \frac{P(v|\theta_a, \theta_s, \theta_c)P(\theta_a, \theta_s, \theta_c)}{P(v)} = \frac{P(v|\theta_a, \theta_s, \theta_c)\,p(\theta_a)P(\theta_c)}{P(v)} \quad (15)$$

Then what results from the global independence assumption is a posterior distribution that factorizes over the conditional tables. Still, the parameter θ_c is itself 4 dimensional [20].

4.4.5 Feedforward NN and Bayesian formula

A feed forward NN is considered as a multilayer perceptron. It represents an example of a non-linear model of classification. Consider an input vector $x=\{x_1,x_2,x_3,\ldots,x_p\}$ the structure in question is illustrated by eq. 16:

$$g\left\{w_{00}+\sum_{j=1}^{m} w_{0j} f\left(w_{j0}+\sum_{k=1}^{p} w_{jk}x_k\right)\right\} \quad (16)$$

where: $W=\{w_{jk}\}$ weights.

w_{jk}: Connection from input to hidden layer connection from hidden layer to output.

$g(.)$: The activation function is the $f(.)$ activation function for hidden neuron.

The j^{th} hidden layer has output as in Eq. 17:

$$z_j = f\left(w_{j0}+\sum_{k=1}^{p} w_{jk}x_k\right) \quad (17)$$

The model is defined as a non-linear regression if the error term is added as in Eq. 18:[20]

$$y = g\left\{w_{00}+\sum_{j=1}^{m} w_{0j} f\left(w_{j0}+\sum_{k=1}^{p} w_{jk}x_k\right)\right\}+\varepsilon \quad (18)$$

4.4.6 Bayesian Regularization of Neural Networks

An ideal NN model is characterized as having small errors in as well as out of sample. In order to produce a network that generalizes well, there exists a proposed method to constrain the size of the network parameters by means of regularization. The idea is that there is a degree of smoothness in the true underlying function; so when the parameters in a network are kept small, the network response will be smooth. Any modestly oversized network, therefore, must be capable of sufficiently representing the true function, rather than capturing the noise. With regularization, the objective function becomes [7]:

$$F=\lambda E_D + \alpha E_W \quad (19)$$

In this equation (19), F represents the objective function; E_w and E_D represent the sum of the square of the network weights and the sum of the squared errors respectively.

The density function of a Bayesian network is given according to Baye's rule and the weights are considered as random variables.

$$P(w|D, \alpha, B, M) = \frac{P(D|w, \beta, M)\,p(w|\alpha, M)}{P(D|\alpha, \beta, M)} \quad (20)$$

With w being the network weights' vector, the network used is represented by M .D the data vector and the neural.

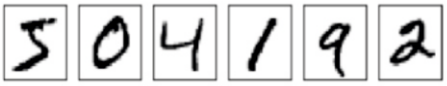
5- SIMULATION AND RESULTS

5.1 Simulation

The resilient back propagation algorithm, which was simulated by MATLAB R2013a software, compared the performance of the algorithm to Gradient Descent algorithm and Bayesian Regularization algorithm relative to Regression, RMSE and Accuracy.

5.2 Experimental part:

Experiment is performed by classifying the dataset as Training, Testing and Validation data. MNIST (Mixed National Institute of Standards and Technology) database is employed in the experiment and is subdivided into training and testing datasets. For convenient test of performance, the training dataset is independent of testing dataset. There are 70,000 scanned handwriting images in MNIST dataset. These are grey scale images size 28 by 28. First 60,000 images are taken as training dataset. The remaining 10,000 are used as testing dataset (Pan et al, 2014). Below are a few images from MNIST:



The data set is classified in the following manner:
1. Training Set: Images are used for learning in a way that fits the weights of classifier.

2. Validation Set: Parameters of a classifier are tuned by validation set.

3. Testing Set: Performance of network is assessed by testing set. Experiment is analyzed at three layers of Neural Network;

1. Input Layer
2. Intermediate (Hidden) Layer
3. Output Layer

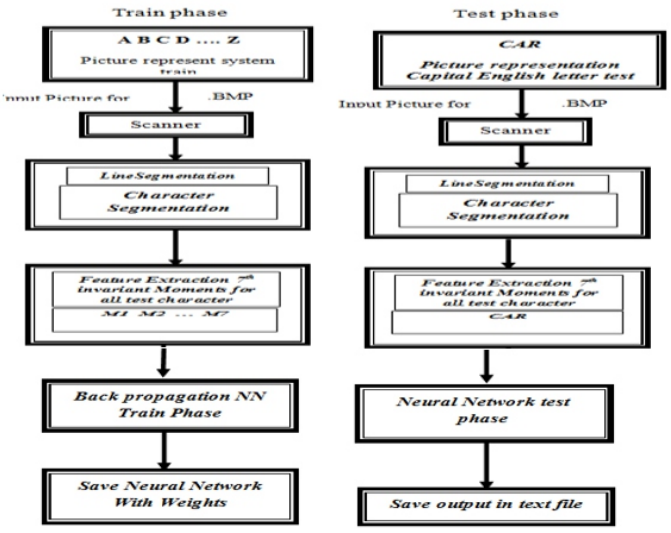
Hidden layer is in charge of modification to the input pattern via weighted connections.

ALGORITHMS: EXPERIMENTAL COMPARISON
Resilient back-propagation, Gradient descent and Bayesian regularization are utilized to observe performance analysis of neural network. Being one of the most popular adaptive learning rates training algorithm, resilient back propagation is adopted as reference algorithm to compare with others. In this paper, we apply the multilayer perceptron for training of the network with the above mentioned algorithms for comparison. Performance and regression plots are analyzed [21].

The conditions for stopping the training are as follows:
- Maximum Epochs.
- Maximum time exceeded.
- Minimum performance goal.
- Minimum Gradient.
- Maximum Validation Fail Times.

Mean square error is calculated in each epoch in order to average the squared difference between outputs and targets. The lower MSE values are the better the performance is, thus zero MSE means no error. Regression analysis shows the relation of outputs and targets.

The simulation steps are presented in Fig. 11.



Figure(7) : General graph of the simulation

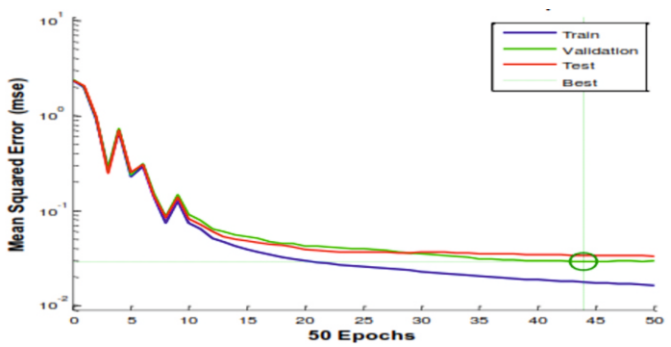


Figure (8) Resilient Back propagation Learning Curve

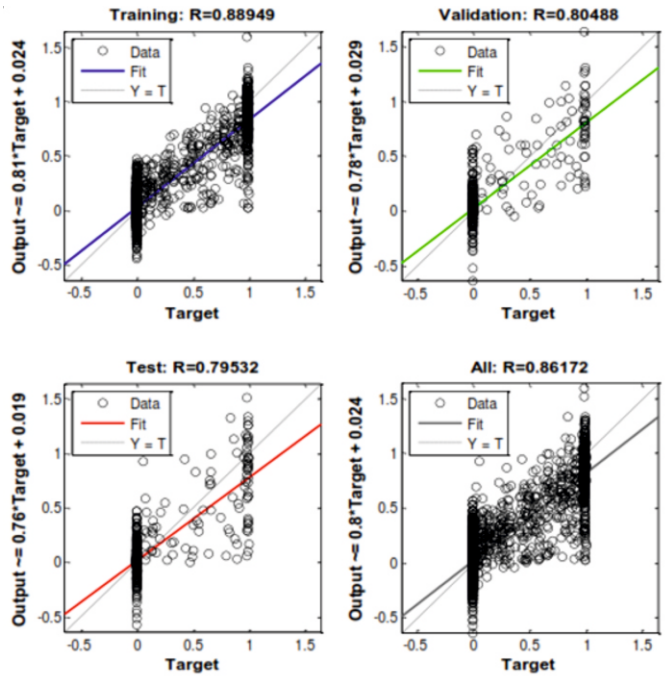


Figure (9) Resilient Back Propagation RegressionPlot

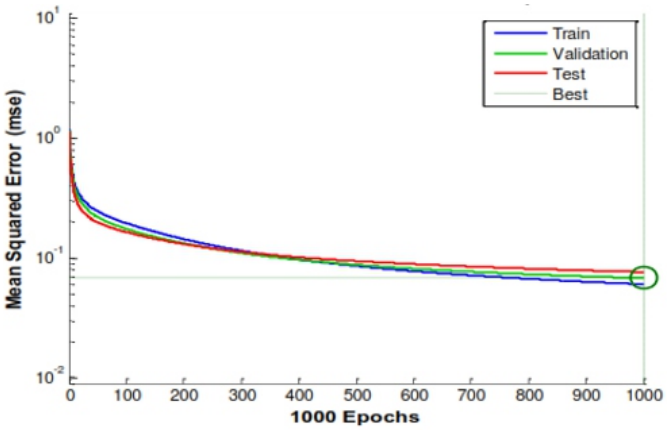


Figure (10) Gradient Algorithm Learning Curve

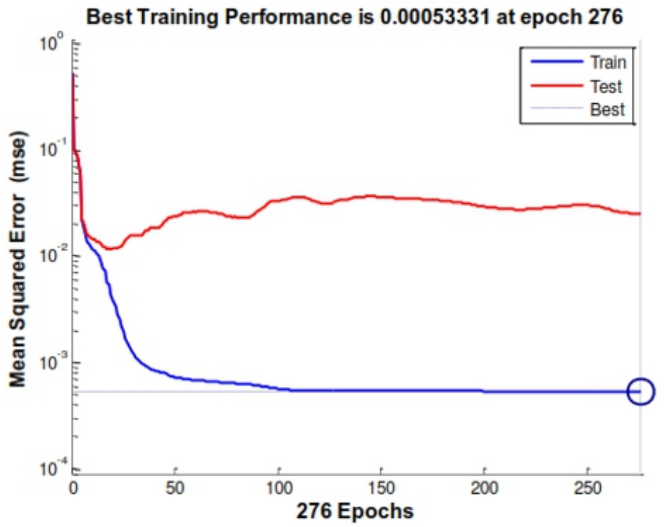


Figure (11) Bayesian Regularization Algorithm Learning Curve

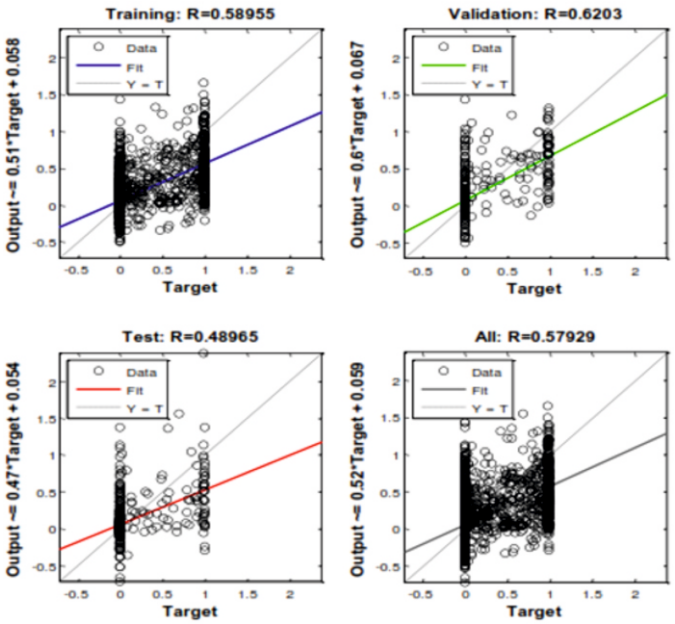


Figure (12) Gradient Descent Algorithm Regression Plot

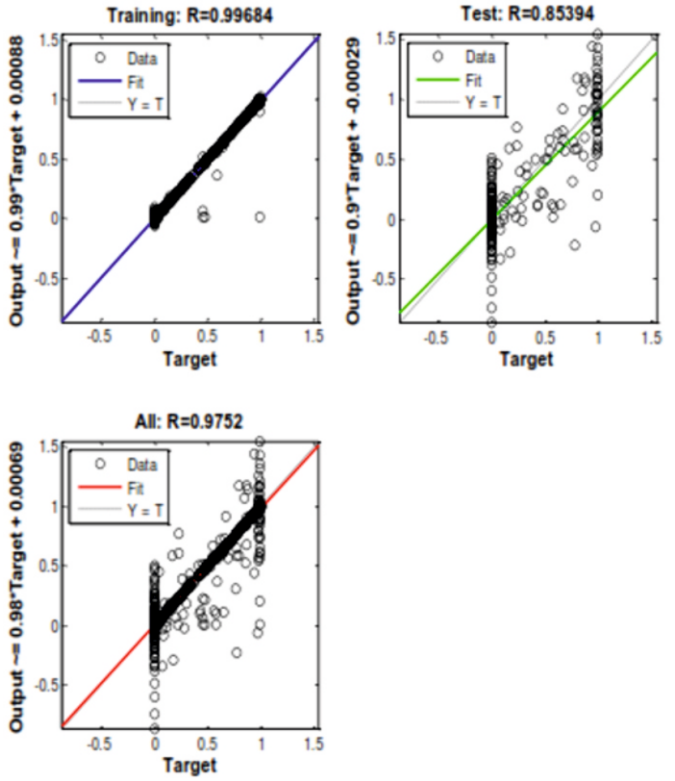


Figure (13) Bavesian Regularization Plot

5.2.1 Accuracy of the three algorithms:

Whenever a neural network is trained, the solution to the problem varies because we have divided the data differently into training, validation and test groups. Also the initial weights and bias values vary. To ensure that a neural network of good accuracy has been detected, we retrain the network on the same input data for several times to see the accuracy. We have also performed the same; the accuracy of the simulated algorithms is tested by repeating the algorithms for 15 times. The target error was set to 0.09. When all other parameters of the training maintain their default, the Bayesian Regularization algorithm gives good accuracy on the given data. Secondly, the resilient back propagation algorithm gives good accuracy and finally the gradient descent accuracy is shown relative to achieving the target value.

Table 2: Comparative Analysis Result

Algorithm	Accuracy
Resilient Back Propagation	80.36%
Gradient Descent	70.66%
Bayesian Regularization	98.69%

5.2.2 Results

Throughout this work we conducted an analysis and compared the performance of neural networks through implementing resilient back propagation, Bayesian Regularization and gradient descent algorithms. Feed-forward neural network is trained by the above-mentioned algorithms; its performance is also evaluated. MNIST database is used to conduct experiments whose results show that learning and test curves are identical. Regression plots of three algorithms are also compared to reveal the relation between outputs and targets. Based on our set parameters and results, it is concluded that resilient back propagation converges much faster than gradient descent and Bayesian regularization. Based on the comparison results in Table 2, it is evident that Bayesian Regularization and Resilient Back Propagation are capable of giving a good result for character recognition. However, Bayesian Regularization provides higher accuracy percentage than both the other techniques. Furthermore, it is capable of completing the training much faster compared to gradient descent.

6- CONCLUSION

A range of aspects of artificial neural network are tackled within this work where we conducted a study comparing the human brain principle to the artificial neural network. Consequently, it is maintained that in spite of the similarity between the artificial neural network principle and the human nervous system, the human brain is more apt to and capable of adapting to the learning process than the artificial neural network.

Artificial neural network fields provide a variety of applications ranging from business intelligence to pattern recognition. In chapter 5, handwritten system recognition was built with the main application being the ability to generate from scanned hard copy of a document a corresponding soft and editable copy.

The simulation results indicate that Bayesian Regularization provides accuracy percentages higher than both the other techniques. Furthermore, it is capable of completing the training much faster compared to gradient descent.

REFERENCES:

1. M. Hajek, " Neural Networks ", thesis, 2005.
2. R.Rojas, " Neural Networks A systematic Introduction", Springer-Verlag, Berlin, 1996.
3. Panduranga, P.P. ; KLS Gogte Inst. of Technol., Belgaum, Rao, D.H. Deshpande, A.Fault "Tolerance Analysis of Neural Networks for Pattern Recognition"2000.
4. S. Kullback, " Information Theory and Statistics", Dover, 1968.
5. AL-Dulaimi, Buthaina ;Ali, Hamza, "Enhanced Traveling Salesman Problem Solving by Genetic Algorithm Technique (TSPGA)",world Academy of Science, Engineering and Technology ,Page 296.(2008).
6. p. Forsland, " A neural Network Based Brain – computer Interface for Classification of Movement Related EEG", Thesis, Linkoping, December, 2003.
7. R.Gencay and M. Qi, " Princing and Heading Derivative Securities with Neural Networks: Bayesian Regularization, Early stopping, and Bagging", IEEE Transaction on Neural Network , Vol.12, No. 4, July, 2001.
8. D. Graupe, " PRINCIPLES OF ARTIFICIAL NEURAL NETWORKS ", 2nd Edition, ADVANCED SERIES IN CIRCUITS AND SYSTEMS, Vol. 6, University of lllinois, Chicago, USA.
9. S.Hiregoundar, Manjunath.K, K. S. Patil, " A survey: Research Summary on Neural Network ", IJRET: International Journal of Research in Engineering and Technology, Vol.03, Issue 03, May, 2014.
10. U. N. Lerner, "Hybrid Bayesian Networks for Reasoning about Complex Systems", Computer science department, Stanford University, 2002.
11. P. W. Kasteleyn, "Dimer Statistics and Phase Transitions", Journal of Mathematical Physics, 4(2):287{293,1963.}
12. V K Jian, New Delhi 2009 , "Information Technology issue and challenges", ISBN: 978-81-7446-706-5.
13. Y. L. Loh, E. W. Carlson, and M. Y. J. Tan, "Bond-propagation algorithm for thermodynamic functions ingeneral two-dimensional Ising models", Physical Review B, 76(1):014404, 2007.
14. G. Kitagawa, "The Two-Filter Formula for Smoothing and an implementation of the Gaussian-sum smoother", Annals of the Institute of Statistical Mathematics, 46(4):605{623, 1994.}
15. AL- Naima F, Al-Timemy A, "Resileint Back Propagation Algorithm for Breast Biospy Classification Based on Artificial Neural Network", (2010), Computational Intelligence and Modern Heuristics.
16. M. Dharlingam and R. Amalraj, " ARTIFICIAL NEURAL NETWORK ARCHITECTURE FOR SOLVING THE DOUBLE DUMMY BRIDGE PROBLEM IN CONTRACT BRIDGE", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 12, December 2013.
17. Gonzalez, R. C. et al., "Digital Image Processing", second edition. (addison- wesly problishing company, Inc.), 1987.
18. C-J. Kim and C. R. Nelson, "State-Space models with regime switching", MIT Press, 1999.
19. D. Barber, " Bayesian Reasoning and Machine Learning", 2010.
20. Raudys, Sarunas, 2001 Statistical and Neural Classifiers "an integrated approach to design",(Text Categorization by Back-propagation Network (0975 – 8887), 2010).
21. Halim Sh., Ahmed A., Noh N., Safudin M., and Ahmed R. "A comparative study between Standard Back Propagation and Resilient Propagation on Snake Identification Accuracy". IEEE, Malaysia, 2011.