Pattern Classification using Artificial Neural Networks

THESIS SUBMITTED IN PARTIAL FULFILLMENT FOR

THE DEGREE OF

Bachelor of Technology

in

Computer Science and Engineering

By

Priyanka Mehtani

Roll no:107CS050

Archita Priya

Roll no:107CS051

Under the Guidance of

Prof. S. K. Rath

Department of Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela-769 008, Orissa, India

National Institute of Technology
Rourkela

CERTIFICATE

This is to certify that the thesis entitled **"Pattern Classification using Artificial Neural Networks "** submitted by **Priyanka Mehtani : 107CS050** and **Archita Priya : 107CS051** in the partial fulfillment of the requirement for the degree of Bachelor of Technology in Computer Science Engineering, National Institute of Technology, Rourkela , is being carried out under my supervision. To the best of my knowledge the matter embodied in the thesis has not been submitted to any other university/institute for the award of any degree or diploma.

Prof. S.K.Rath

Date : Department of Computer Science and Engineering

National Institute of Technology

ACKNOWLEDGEMENT

We wish to express our sincere and heartfelt gratitude towards our guide **Prof.S.K.Rath**, Computer Science Engineering Department, for his guidance, sympathy, inspiration and above all help during the duration of our project.

We would also like to thank all the professors of the department of Computer Science and Engineering, National Institute of Technology, Rourkela, for their guidance and the inspiration.

Submitted by:

Priyanka Mehtani

Roll no-107CS050

Archita Priya

Roll no-107CS051

Computer Science and Engineering

National Institute of Technology

Rourkela

Abstract

Classification is a data mining (machine learning) technique used to predict group membership for data instances. Pattern Classification involves building a function that maps the input feature space to an output space of two or more than two classes. Neural Networks (NN) are an effective tool in the field of pattern classification, using training and testing data to build a model. However, the success of the networks is highly dependent on the performance of the training process and hence the training algorithm. Many training algorithms have been proposed so far to improve the performance of neural networks. In this project, we shall make a comparative study of training feedforward neural network using the three algorithms - ***Backpropagation Algorithm, Modified Backpropagation Algorithm and Optical Backpropagation Algorithm.*** These algorithms differ only on the basis of their error functions. We shall train the neural networks using these algorithms and taking 75 instances from the iris dataset (taken from the UCI repository and then normalised) ; 25 from each class. The total number of epochs required to reach the degree of accuracy is referred to as the convergence rate. The basic criteria of comparison process are the convergence rate and the classification accuracy. To check the efficiency of the three training algorithms, graphs are plotted between **No. of Epochs vs. Mean Square Error (MSE). The training process continues till M.S.E falls to a value 0.01.** The effect of using the momentum and learning rate on the performance of algorithm are also observed. The comparison is then extended to compare the performance of multilayer feedforward network with ***Probabilistic network.***

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Data mining can be called the process of finding correlations and patterns in huge chunks of data present in large relational databases. It involves many different algorithms to analyse data. All of these algorithms attempt to fit a model to the data. The algorithms examine the data and determine a model that is closest to the characteristics of the data being examined. It is seen as an increasingly important tool to transform data into business intelligence and be made into actionable information.
The term *knowledge discovery in databases (KDD)* and *data mining* are often used interchangeably. KDD is the process of determining useful information and patterns in data. KDD is followed by Data Mining. Data mining is the process of using algorithms to extract information and patterns derived by the KDD process. The KDD process consists of the following steps selection, pre-processing, transformation, data mining and interpretation/evaluation. Basic data mining tasks are classification, regression, time series analysis, prediction, clustering, summarization, association rules and sequence discovery. Classification maps data into predefined groups or textitclasses. It comes under something known as supervised learning because the classes are determined before examining the data. All approaches to performing classification assume some knowledge of the data. Usually, a training set is used to develop the specific parameters required. The problems of prediction or classification can be solved by using neural networks.
Neural networks(NNs) are simplified models of the biological nervous systems[10]. An NN can be said to be a data processing system, consisting of a large number of simple, highly interconnected processing elements(artificial neurons), in an architecture

inspired by the structure of the cerebral cortex of the brain. The interconnected neural computing elements have the quality to learn and thereby acquire knowledge and make it available for use. NNs have found wide applications in areas such as pattern recognition, image processing, optimisation, forecasting, and control systems to name a few.

In this project, the input to the neural networks is the IRIS dataset obtained from the UCI Repository. The Iris dataset contains 3 classes of 50 instances each, where each class refers to a type of iris plant ***virginica, setosa and versicolor.*** One class is linearly separable from the other two. Every instance consists of ***4 dimensions-sepal length, sepal width, petal length and petal width.*** The values are downloaded and then normalised to obtain a matrix with values ranging from 0 to 1.

Chapter 2

LITERATURE REVIEW

The main drawback with OBP algorithm is that it does not ensure that it would convergence to absolute minimum error, despite improving the high convergence speed as compared to standard BP.In [4], a computational technique is implemented to work out network error such that the learning rate can be computed. It is done by calculating the error for each layer, based on which the differential of the error for each layer may be determined. The result obtained is the learning rate for that layer. In each iteration, the learning rate is updated according to the error of the layers. The assumption is that each layer has a varying learning rate which needs to be updated in each iteration. In BP , error is calculated as follows:

Error = $target - output$

While the error in adjusted Differential Adaptive Learning Rate Method(DALRM) will be as:

α=f (| target - output |)

Where f is an activation function that can be different in each layer.DALRM can escape from local minima and converge to absolute minimum.

Probabilistic Neural networks is a very efficient classifier. It can be used in the establishment of intelligent systems for classifying marine vessels by the acoustic radiated noise. The evaluation results of proposed method in [7] with real data shows this method is successful in classifying ships to separate categories (heavy and medium ships). In [2], ship classification based on covariance of discrete wavelet using probability Neural Network is implemented. Five ship types were choosen to be modeled: namely, an aircraft carrier, a frigate, a destroyer, a research ship (Point Sur),

and a merchant ship. In this paper the covariance matrix of discrete wavelet transform of ship image has been used for Ship Classification using Probabilistic neural networks. Pattern recognition applications, especially handwritten character recognition[3], is one of the most widely used applications of Backpropagation Neural Networks (BPNN).

PNN is also used Handwritten Character Recognition of different languages eg- Arabic character recognition in [8]. PNNs application covers: classification brain tissues in multiple sclerosis, classification image textures, classification of liver tumors, EEG pattern classification, cloud classification, power disturbance recognition and classification and allocation of the load profiles to consumers etc.[1]

Chapter 3

CONCEPTS

3.1 Data Mining

Data mining is often defined as finding hidden information in database. Data mining access of a database differs from the traditional access in several ways:

- **Query:** Query might not be accurately stated or well formed. The data miner might not even be exactly sure of what he wants to see.
- **Data:** The data accessed is usually a different version from that of the original operational database. The data have to be cleansed and modified to better support the mining process.

- **Output:** The output of the data mining query probably is not a subset of the database. Instead, it is the output of some analysis of the contents of the database.

Data mining algorithms can be characterised as consisting of 3 parts:

- *Model :* The purpose of the algorithm is to fit a model to the data.
- *Preference :* Some criteria must be used to fit one model over another.
- *Search :* All algorithms require some technique to search the data.

Each model created can be either predictive or descriptive in nature. A *predictive model* makes a prediction about values of data using known results from different

data. Predictive model data mining tasks include classification, regression, time series analysis and prediction. A *descriptive model* identifies patterns or relationships in data. Clustering, summarization, association rules and sequence discovery are usually viewed as descriptive in nature.

3.2 Artificial Neural Networks

Classification is a data mining (machine learning) technique used to predict group membership for data instances. To simplify the problems of prediction or classification, neural networks are being introduced. Neural networks are simplified models of the biological neuron system. It is a massively parallel distributed processing system made up of highly interconnected neural computing elements that have the ability to learn and thereby acquire knowledge and make it available for use. Various learning mechanisms exist to enable the NN acquire knowledge. NN architectures have been classified into various types based on their learning mechanisms and other features. This learning process is referred to as training and the ability to solve a problem using the knowledge acquired as inference. NNs are simplified imitations of the central nervous system[10], and therefore, have been inspired by the kind of computing performed by the human brain. The structural constituents of a human brain termed neurons are the entities, which perform computations such as cognition, logical inference, pattern recognition and so on. Hence the technology, which has been built on a simplified imitation of computing by neurons of a brain, has been termed Artificial Neural Systems (ANS) technology or Artificial Neural Networks (ANN) or simply neural networks. Other names for this technology are *Connectionist Networks, Neuro Computers, Parallel Distributed Processors* etc. Also, neurons are can also be referred to as *neurodes,Processing Elements (PEs),* and *nodes.*

3.2.1 Properties of Neural Networks

1. The NNs display mapping capabilities, i.e. , they can map input patterns to their associated output patterns.[10]

2. The NNs learn by examples. Thus, NN architectures can be trained with known examples of a problem before they are tested for their inference capability on

unknown instances of the problem. They can, therefore, identify new objects previously untrained.

3. The NNs possess the ability to generalise. Thus, they can predict new outcomes from the past trends.

4. The NNs are robust systems and are fault tolerant. They can, therefore, recall full patterns from incomplete, partial or noisy patterns.

5. The NNs can process information in parallel, at high speed, and in a distributed manner.

3.2.2 Neural Network Characteristics

The word network in Neural Network refers to the interconnection between neurons present in various layers of a system. Every system is basically a 3 layered system, which are Input layer, Hidden Layer and Output Layer. The input layer has input neurons which transfer data via synapses to the hidden layer, and similarly the hidden layer transfers this data to the output layer via more synapses. The synapses stores values called weights which helps them to manipulate the input and output to various layers. An ANN can be defined based on the following three characteristics:

1. The Architecture: The number of layers and the no. of nodes in each of the layers

2. The learning mechanism which has been applied for updating the weights of the connections.

3. The activation functions used in various layers.

3.2.2.1 Architecture

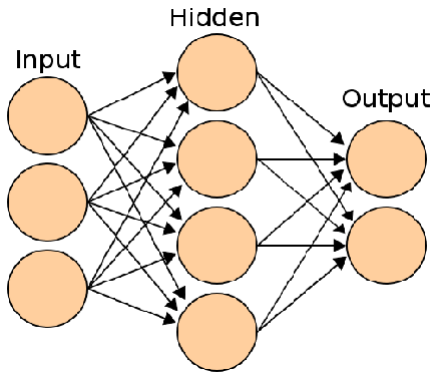


Figure 3.1: *Architecture of Neural Network*

Neural Network (NN) can be represented using a directed graph G, an ordered 2-tuple (V,E) consisting of a set V of vertices and E of edges with verticesV= $\{1, 2, , n\}$ and arcs A= { < i,j > | i ≥1,j ≤ n } ,having the following restrictions:

- V is partitioned into a set of input nodes, V_I, hidden nodes, V_H, and output nodes,V_O.
- The vertices are also partitioned into layers.
- Any arc $<i, j>$ must have node i in layer h-1 and node j in layer h.
- Arc $<i, j>$ is labeled with a numeric value w_{ij}.
- Node i is labeled with a function f_i.

When each edge is assigned an orientation, the graph is directed and is called a *directed graph* or a *diagraph. A feed forward network* has directed acyclic graph. Diagraphs are important in neural network theory since signals in NN systems are restricted to flow

in particular directions. The vertices of the graph represent neurons(input\output) and the edges, the synaptic links. The edges are labelled by the weights attached to the synaptic links. The different classes of the networks are:

(a) Single Layer Feedforward Network

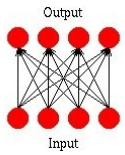


Figure 3.2: *Single Layer FeedForward Network*

This type of network consists of two layers, namely the *input layer* and the *output layer.* The input layer neurons receive the input signals and the output layer neurons receive the output signals. The synaptic links carrying the weights connect every input neuron to the output neuron but not vice-versa. Such a network is said to be feedforward. Despite the two layers, the network is termed single layer since it is the output layer, alone which performs the computation. The input layer merely transmits the signals to the output layer. Hence, the name, single layer feedforward network.

(b) Multilayer Feedforward Network

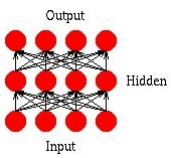


Figure 3.3: *Multi Layer FeedForward Network*

This network is made of multiple layers. It possesses an input and an output layer and also has one or more intermediary layers called hidden layers. The computational units of the hidden layer are known as the hidden neurons or hidden units. The hid-

den layer aids in performing useful intermediary computations before directing the input to the output layer. The input layer neurons are linked to the hidden layer neurons and the weights on these links are referred to as input-hidden layer weights. Similarly, the hidden layer neurons are linked to the output layer neurons and the corresponding weights are referred to as hidden-output layer weights.

(c) Recurrent Networks

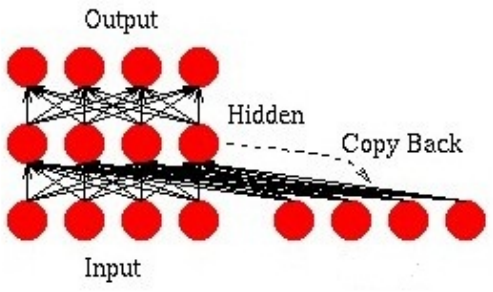


Figure 3.4: *Simple Recurrent Network*

These networks differ from feedforward network architectures in the sense that there is atleast one feedback loop. Thus, in these networks there could exist one layer with feedback connections. There could also be neurons with self-feedback links, i.e., the output of a neuron is fed back into itself as input.

3.2.2.2 Learning Methods

Learning methods[10] in NNs can be classified into 3 basic types:

- **Supervised Learning:** In this, every input pattern that is used to train the network is associated with a target or the desired output pattern. A teacher is assumed to be present during the learning process when a comparison is made between the networks computed output and the correct expected output, to determine the error. Tasks that fall under this category are Pattern Recognition and Regression.

- **Unsupervised Learning:** In this learning method, the target output is not presented to the network. The system learns of its own by discovering and adapting to structural features in the input patterns as if there is no teacher to present the desired patterns. Tasks that fall under this category includes Clustering, Compression and Filtering.

- **Reinforced Learning:** In this method, the teacher though available, does not present the expected answer but only indicates if the computed output is correct or incorrect. The information provided helps the network in its learning process. But, reinforced learning is not one of the popular forms of learning.

3.2.2.3 Activation Functions

- They are used to limit the output of a neuron in a neural network to a certain value. Output may be in range [-1,1] or [0,1]

- An activation function for a back-propagation net should be continuous, differentiable, and monotonically non-decreasing[8].

- The various kinds of activation functions[11] are:

Linear:

$$f_i(S) = c\,S$$

Threshold or Step:

$$f_i(S) = \left\{ \begin{array}{ll} 1 & if S > T \\ 0 & otherwise \end{array} \right\}$$

Ramp:

$$f_i(S) = \left\{ \begin{array}{ll} 1 & if S > T_2 \\ \frac{S-T_1}{T_2-T_1} & if T_1 \leq S \leq T_2 \\ 0 & if S < T_1 \end{array} \right\}$$

Sigmoid:

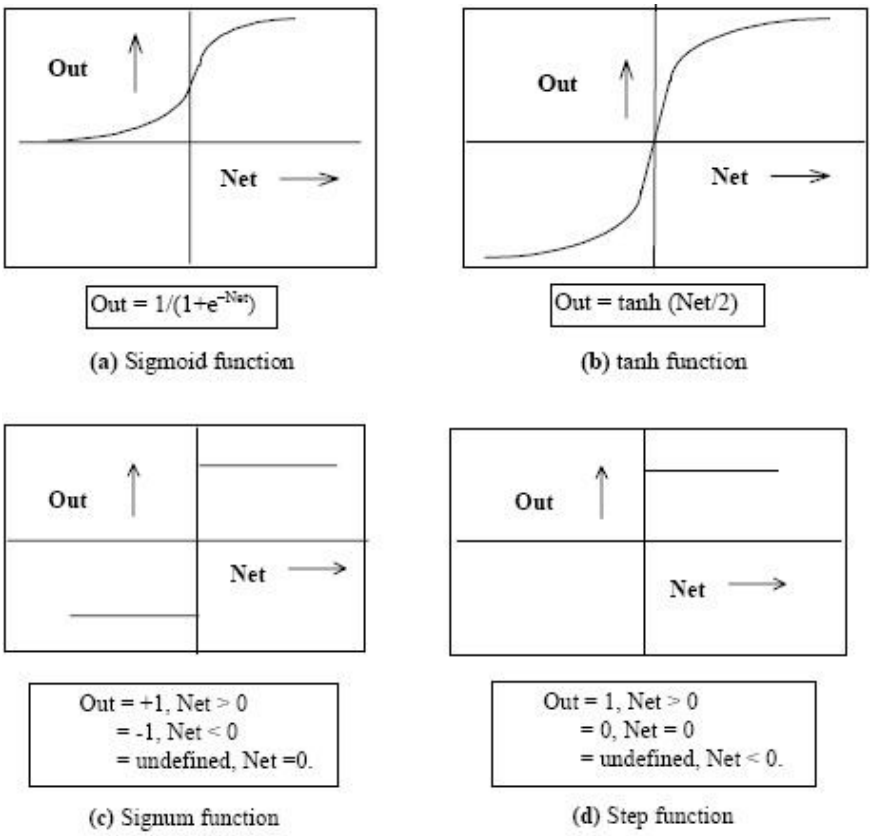
$$f_i(S) = \frac{1}{(1+e^{-c\,S})}$$

Hyperbolic Tangent:

$$f_i(S) = \frac{(1-e^{-S})}{(1+e^{-c\,S})}$$

Gaussian:

$$f_i(S) = e^{\frac{-S^2}{v}}$$



Common non-linear functions used for synaptic inhibition. Soft non-linearity: (a) Sigmoid and (b) tanh; Hard non-linearity: (c) Signum and (d) Step.

Figure 3.5: *Graphs for Activation Functions*

3.3 Classification Using Feedforward Networks

Classification has been performed using 3 algorithms: Backpropagation Algorithm, Modified Backpropagation and Optical backpropagation.

3.3.1 Backpropagation Algorithm

Backpropagation is a method of training multilayer artificial neural networks which uses the procedure of supervised learning. Supervised algorithms are error-based learning algorithms which utilise an external reference signal (teacher) and generate an error signal by comparing the reference with the obtained output. Based on error signal, neural network modifies its synaptic connection weights to improve the system performance. In this scheme, it is always assumed that the desired answer is known "a priori"

We consider the 3-layer network with input layer having 'l' nodes, hidden layer having 'm' nodes, and an output layer having 'n' nodes. We consider sigmoidal functions for activation functions for the hidden and output layers and linear activation function for input layer.

The algorithm[9] involves the following steps:

Step 1: The inputs and outputs are column normalised with respect to the maximum values. It is proven fact that the neural networks work in a better way if the range of the input and outputs is between 0 to 1. For each training pair, we assume there are 'l' inputs given by $\{I\}_I$ (l $\times$ 1) and 'n' outputs $\{O\}_O$ (n $\times$ 1) in a normalised form.

Step 2: The number of neurons present in the hidden layer are assumed to lie between l<m>2l.

Step 3: [V] represents the weights of synapses which connect input neurons and hidden neurons and [W] is used to represent weights of synapses that connect hidden neurons and output neurons. The weights are initialised to random values from -1 to 1. For general problems, λ is assumed as 1 and the threshold values are taken as zero.

$$[V]_0 = [\text{random weights}]$$
$$[W]_0 = [\text{random weights}]$$
$$[V]_0 = [W]_0 = [0]$$

Step 4: One set of inputs and outputs is presented for the training data. The pattern to the input layer $\{I\}_I$ is presented as inputs to the input layer. The output of the input layer may be evaluated by using linear activation function,as

$$\{\mathrm{O}\}_I = \{I\}_I$$
$$\mathrm{l} \times 1 \; \mathrm{l} \times 1$$

Step 5: The inputs to the hidden layer are computed by multiplying corresponding weights of synapses as

$$\{I\}_H = [V]_T \; \{O\}_I$$
$$\mathrm{m} \times 1 \; \mathrm{m} \times \mathrm{l} \; \mathrm{l} \times 1$$

Step 6: Assume that the hidden layer units calculate the output using the sigmoidal function as

$$\{\mathrm{O}\}_\mathrm{H} = \left\{ \begin{array}{c} \cdot \\ \cdot \\ \dfrac{1}{(1 + e^{-\mathrm{I}_{\mathrm{Hi}}})} \\ \cdot \\ \cdot \end{array} \right\}$$

Step 7: The inputs to the output layer are computed by multiplying corresponding weights of synapses as

$$\{I\}_O = [W]_T \; \{O\}_H$$
$$\mathrm{n} \times 1 \; \mathrm{n} \times \mathrm{m} \; \mathrm{m} \times 1$$

Step 8: Assume that the output layer units calculuate the output using sigmoidal function as

The above is the network output.

$$\{O\}_O = \left\{ \begin{array}{c} \cdot \\ \cdot \\ \dfrac{1}{(1+e^{-I_{Oj}})} \\ \cdot \\ \cdot \end{array} \right\}$$

Step 9: The error and the difference between the network output and the desired output is calculated for the ith training set as

$$E^p = \frac{(\Sigma\,(T_j - O_{oj})^2)^{1/2}}{n}$$

Step 10: $\{d\}$ is calculated as

$$\{d\} = \left\{ \begin{array}{c} \cdot \\ \cdot \\ (T_k - O_{ok})\,O_{ok}\,(1 - O_{ok}) \\ \cdot \\ \cdot \end{array} \right\}$$

Step 11: [Y] matrix is calculated as

$$[Y] = \{O\}_H \; <d>$$

$$m \times n \quad m \times 1 \quad 1 \times n$$

Step 12: Find

$$[\Delta W]^{t+1} = \alpha[\Delta W]^{t} + \eta[Y]$$
$$m \times n \qquad m \times n \qquad m \times n$$

where α = momentum

η = learning rate

Step 13: Find

$$\{e\} = [W]\{d\}$$
$$m \times 1 \quad m \times n \quad n \times 1$$

$$\{d^*\} = \begin{Bmatrix} \cdot \\ \cdot \\ e_i(O_{Hi})(1 - O_{Hi}) \\ \cdot \\ \cdot \end{Bmatrix}$$
$$m \times 1 \qquad m \times 1$$

Calculate [X] matrix as

$$[X] = \{O\}_I < d^* > = \{I\}_I < d^* >$$
$$l \times m \quad l \times 1 \quad l \times m \qquad l \times 1 \quad l \times m$$

Step 14: Find

$$[\Delta V]^{t+1} = [V]^t + \eta[X]$$
$$l \times m \qquad l \times m \qquad l \times m$$

Step 15: Find

$$[V]^{t+1} = [V]^t + [\Delta V]^{t+1}$$
$$[W]^{t+1} = [W]^t + [\Delta W]^{t+1}$$

Step 16: Calculate the error rate as

$$\text{Error rate} = \frac{\sum E_p}{nset}$$

Step 17: Repeat steps 4-16 until the error rate convergence is less than the tolerance value.

3.3.2 Modified Backpropagation Algorithm

Although backpropagation is a very popular learning method, a drawback is its slow computing speed. In Modified backpropagation, a logarithmic error function is utilized instead of the quadratic error function[5]. It yields remarkable reduction in learning time and prevents the learning to get stuck in undesirable local minina.
The algorithm is the same as BP. The difference lies in the calculation of the error function.
The error function is calculated as: $E^p = \text{G}$
Where $\text{G}=[O_{ok^*} \ln(T_k/O_{ok}) + (1 - T_k) \ln\{(1 - T_k)/(1 - O_{ok})\}]$
Assuming T_k (target) lies in [0,1] and $0.\ln(0)=0$ and O_{ok} is the output.

3.3.3 Optical Backpropagation Algorithm

The Optical Back-Propagation (OBP) algorithm[6] is designed to overcome some of the problems associated with standard BP training using nonlinear function. One of

the important properties of this algorithm is that it can escape from local minima with high speed of convergence during the training period. The convergence speed of the learning process can be improved by adjusting the error, which will be transmitted backward from the output layer to each unit in the intermediate layer.

In BP, the error at a single output unit is defined as:

$\delta^o Jk = (Y_{Jk}$ - $O_{Jk})$

where the subscript "J "refers to the jth training vector, and "K" refers to the kth output unit. In this case, Y_{Jk} is the desired output value, and O_{Jk} is the actual output from kth unit.

δ_{Jk} will propagate backward to update the output-layer weights and the hidden-layer weights.

While the error at a single output unit in adjusted OBP will be as:

$$New\delta^{\circ}Jk = (1 + e^{(YJk - OJk)2}) \quad , if\ (Y_{Jk} - O_{Jk}) >= zero\ .$$

$$New\delta^{\circ}Jk = -(1 + e^{(YJk - OJk)2}) \quad , if\ (Y_{Jk} - O_{Jk}) < zero\ .$$

The steps of an OBP:

1. Apply the input example to the input units.

$X_j = (X_{j1}, X_{j2}, X_{j3}, .., X_{jN})_t$

2. Calculate the net-input values to the hidden layer units.

$Net^h jl = (i.W^h li.X_{ji})$

3. Calculate the outputs from the hidden layer.

$i_{jl} = f^h l(net^h jl)$

4. Calculate the net-input values to the output layer units.

$net^o jk = (W^o kl.I_{jl})$

5. Calculate the outputs from the output units.

$O_{jk} = f^o l(net^o jk)$

6. Calculate the error term for the output units, pk.

$$\delta^{o}_{ik} = (1 + e^{(Y_{jk} - O_{jk})^{*}2}) \bullet f^{o'}_{k}(net^{o}_{ik}) \text{ when } (Y_{ik} - O_{ik}) >= 0$$

$$\delta^{o}_{ik} = -(1 + e^{(Y_{jk} - O_{jk})^{*}2}) \bullet f^{o'}_{k}(net^{o}_{ik}) \text{ when } (Y_{ik} - O_{ik}) < 0$$

Where, $f^ok(net^ojk) = f^ok(net^ojk\ ).(1f^ok(net^ojk))$

7. Calculate the error term for the hidden units, through applying Newδ^ojk, also.

$\delta^h jl = f^h l(net^h jl).(\delta^o jk.W^o kl)$

Notice that the error terms on the hidden units are calculated before the connection weights to the output-layer units have been updated.

8. Update weights on the output layer.

$W^o kl(t+1) = W^o kl(t) + (\eta.New\delta^o jk.i_{jl})$

9. Update weights on the hidden layer.

$W^h li(t+1) = W^h li(t) + (\eta.New\delta^h pl.X_i)$

10. Repeat steps from step 1 to step 9 until the error $(Y_{jk}O_{jk})$ is acceptably small for each training vector pairs.

The proposed algorithm as classical BP is stopped when the squares of the differences between the actual and target values summed over the output units and all patterns are acceptably small.

Where,

net^ojk - net input to the kth output unit.

W^okl -weight on the connection from the lth hidden unit to kth output unit .

O_{jk} -actual output for the kth output unit .

Y_{jk} -desired output for the kth output unit .

f -(Sigmoid) activation function.

f -derivative of activation function.

$\delta^o jk$ -signal error term for the kth output unit.

$\delta^h pj$ -signal error term for the jth hidden unit

3.4 Classification Using Probabilistic Neural Networks

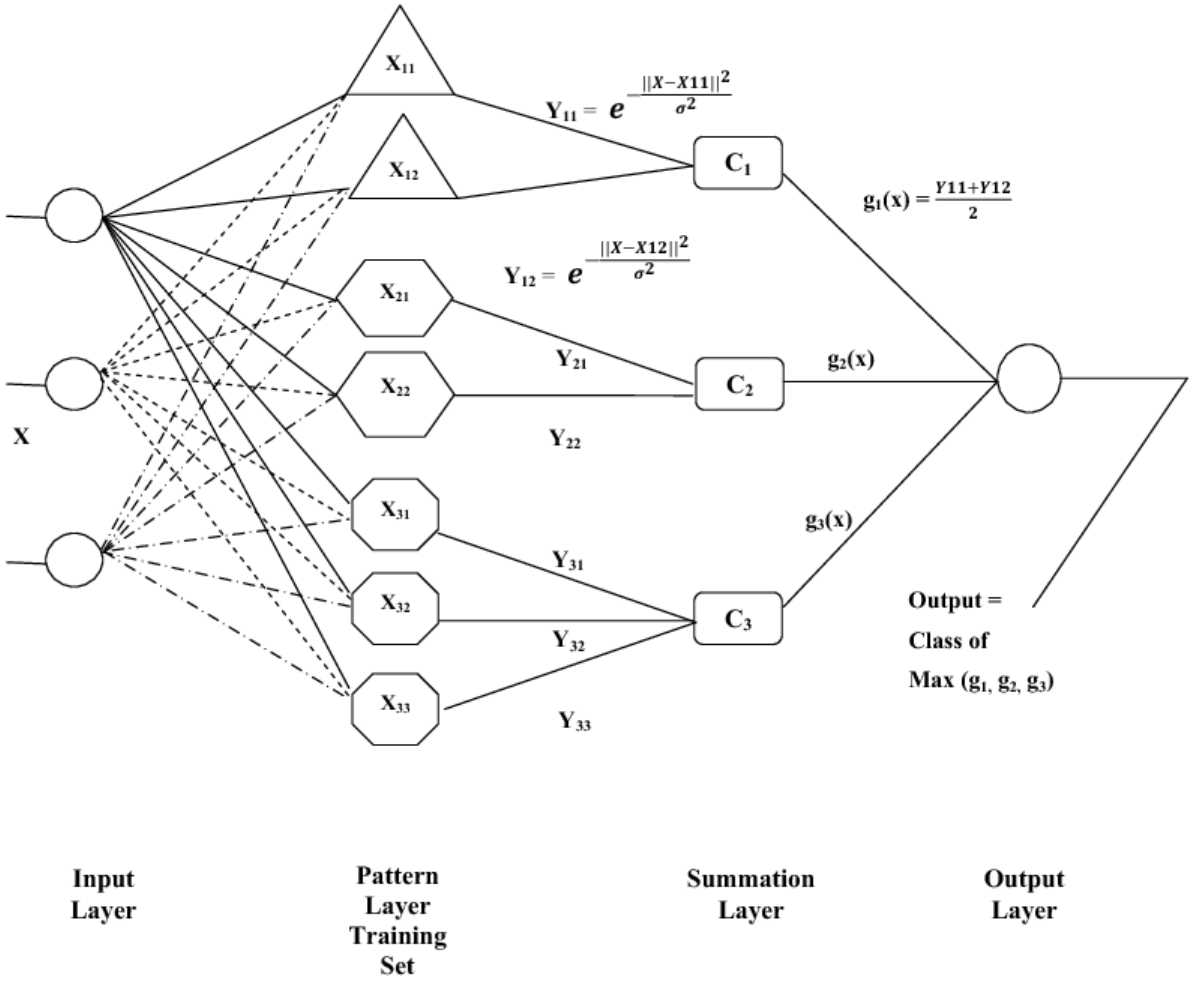


Figure 3.6: *Training and Testing in Probabilistic Neural Networks*

A Probabilistic Neural Network (PNN) is defined as an implementation of statistical algorithm called Kernel discriminate analysis in which the operations are organized into multilayered feed forward network[1]. It has four layers: input layer, pattern layer, summation layer and output layer. A PNN is mainly a classifier since

it can map any input pattern to a number of classifications.
The main advantages of using PNN are :

- Fast training process: The order of magnitude is much higher as compared to backpropagation
- An inherently parallel structure,
- Guaranteed to converge to an optimal classifier as the size of the representative training set increases and
- Training samples can be added or removed without extensive retraining.

On the other hand, the main disadvantages are:

- It is not as general as back-propagation.
- Large memory requirements.
- Slow execution of the network.
- It requires a representative training set even more so than other types of NN's.

The learning rate for a PNN is much higher than many neural networks models such as backpropagation.

Chapter 4

SIMULATION RESULTS AND GRAPHS

The simulation process is carried on a computer having Dual Core processor with speed 1.73 GHz and 2 GB of RAM.The MATLAB version used is R2010a.

Implementation Details:

- The Iris dataset (downloaded from the UCI repository, *www.ics.uci.edu*, which is a 150×4 matrix, is taken as the input data. Out of these 150 instances, 75 instances were used for training and 75 for testing.
- Under supervised learning, the target of the first 25 instances is taken as 0, for the next 25 instances as 0.5 and for the last 25 instances as 1.
- The network architecture taken was 4×3×1,i.e, the input layer has 4 nodes, the hidden layer has 3 nodes and the output layer has 1 node.
- The tolerance value taken taken was 0.01.

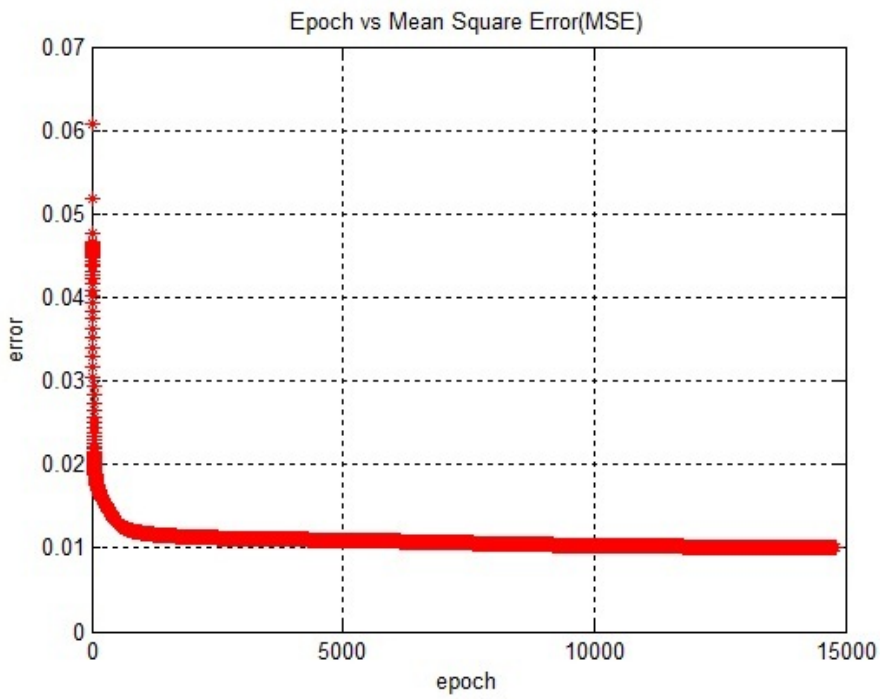


Figure 4.1: *Epoch vs. MSE for BP at α=0.3 and η=0.1*

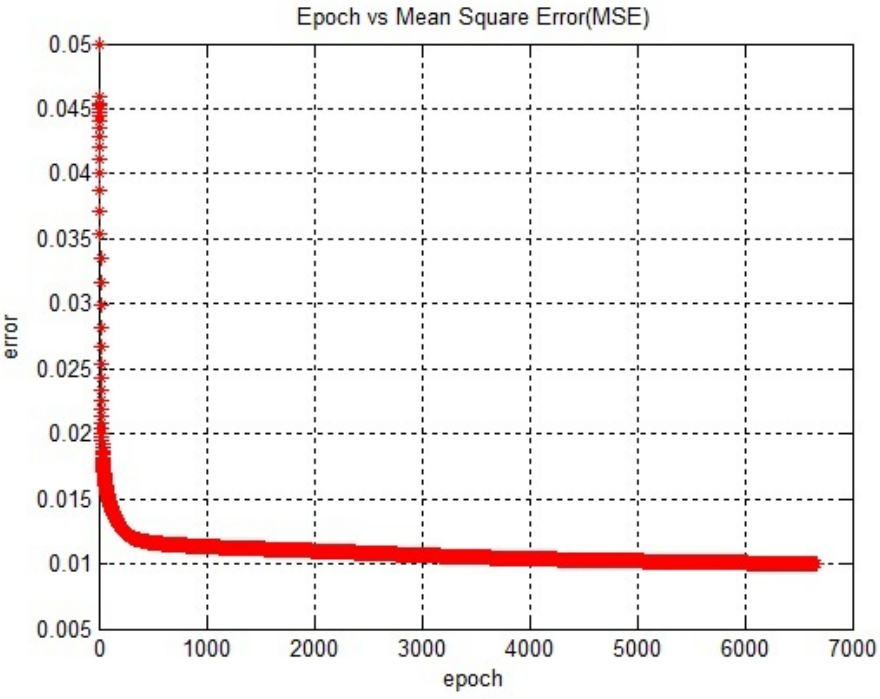


Figure 4.2: *Epoch vs. MSE for BP at α=0.2 and η=0.2*

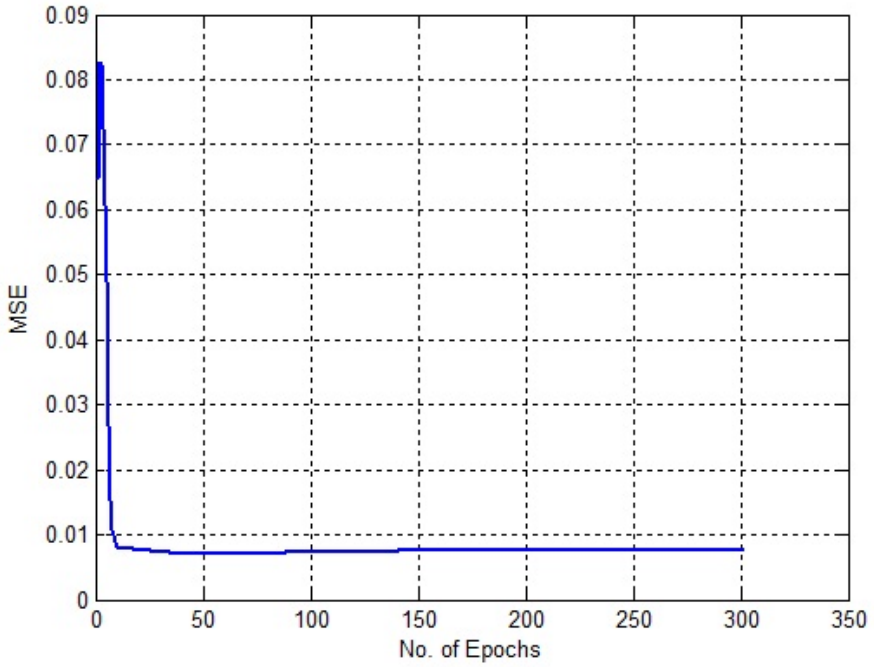


Figure 4.3: *Epoch vs. MSE for MBP at α=0.4 and η=0.6*

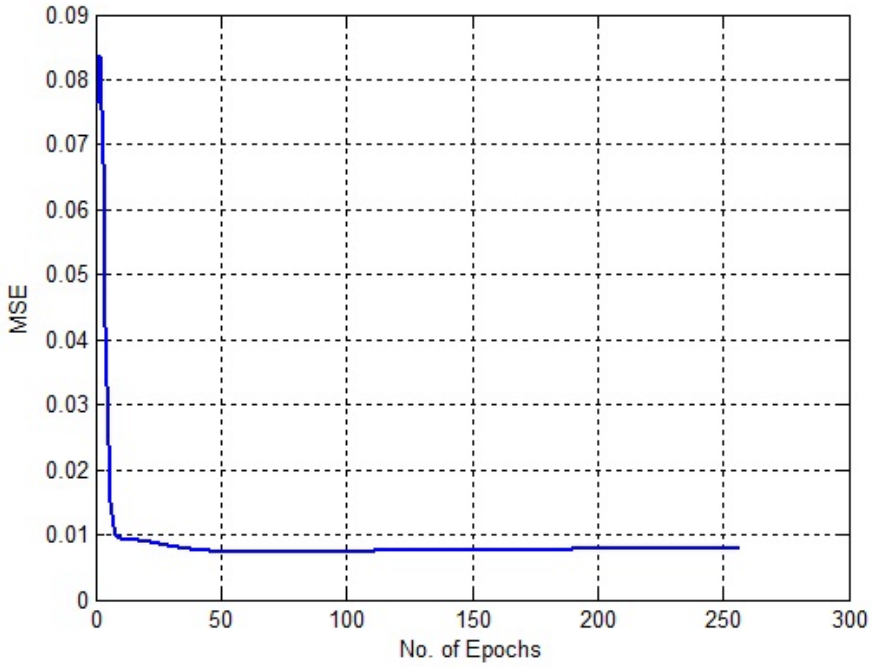


Figure 4.4: *Epoch vs. MSE for MBP at α=0.3 and η=0.8*

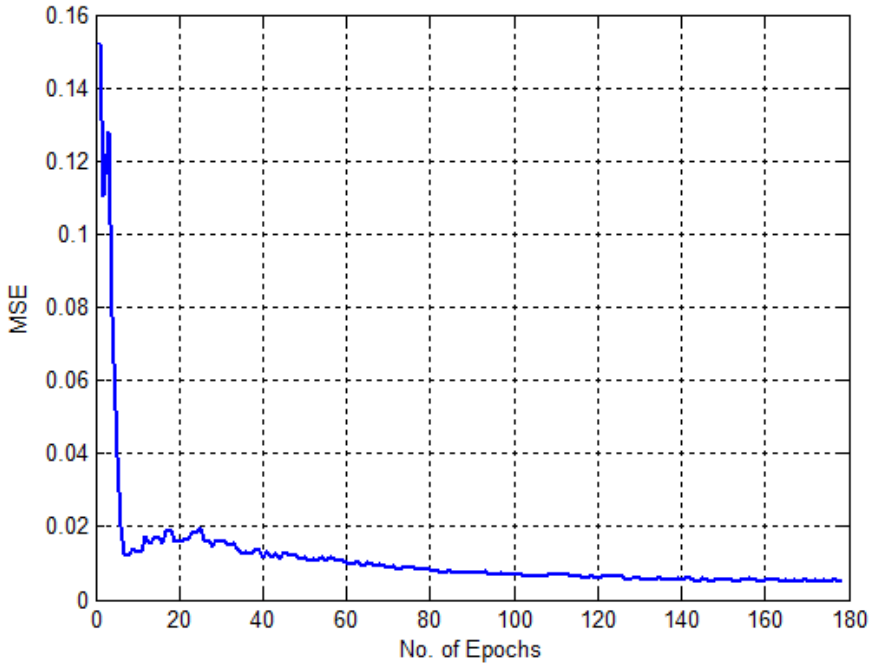


Figure 4.5: *Epoch vs. MSE for OBP at α=0.1 and η=0.1*

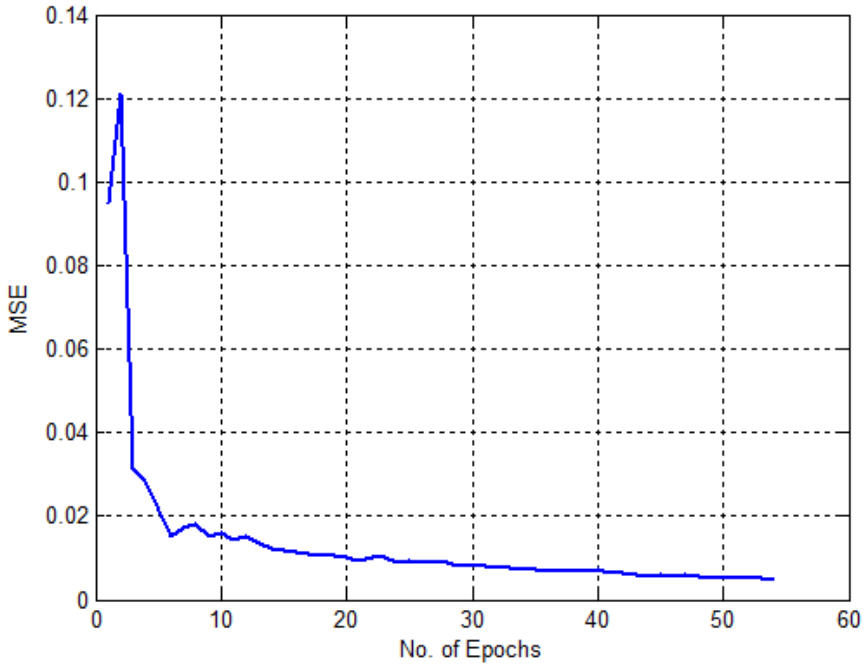


Figure 4.6: *Epoch vs. MSE for OBP at α=0.3 and η=0.9*

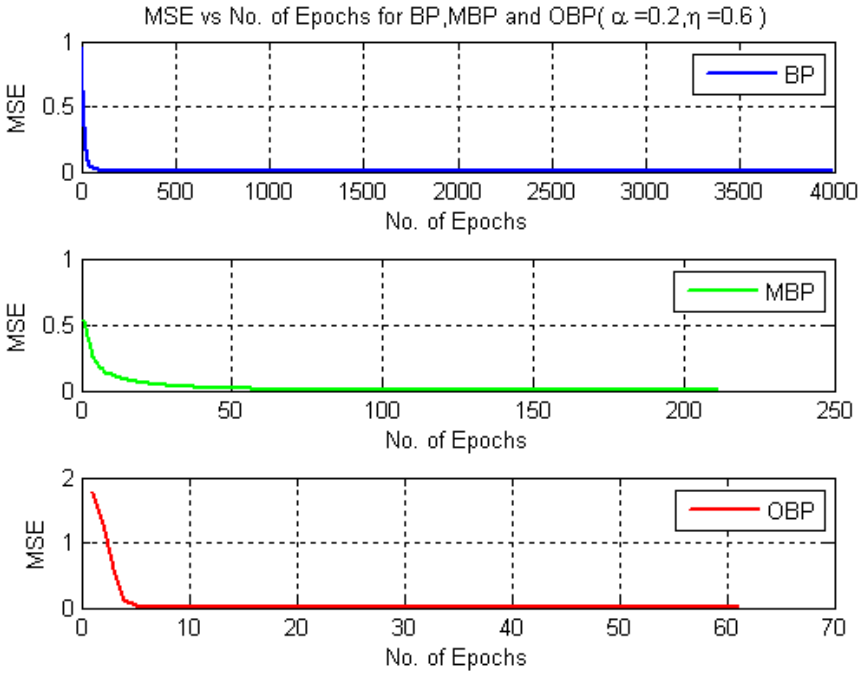


Figure 4.7: *Comparison of Training Speed of BP, MBP and OBP at α=0.2 , η = 0.6*

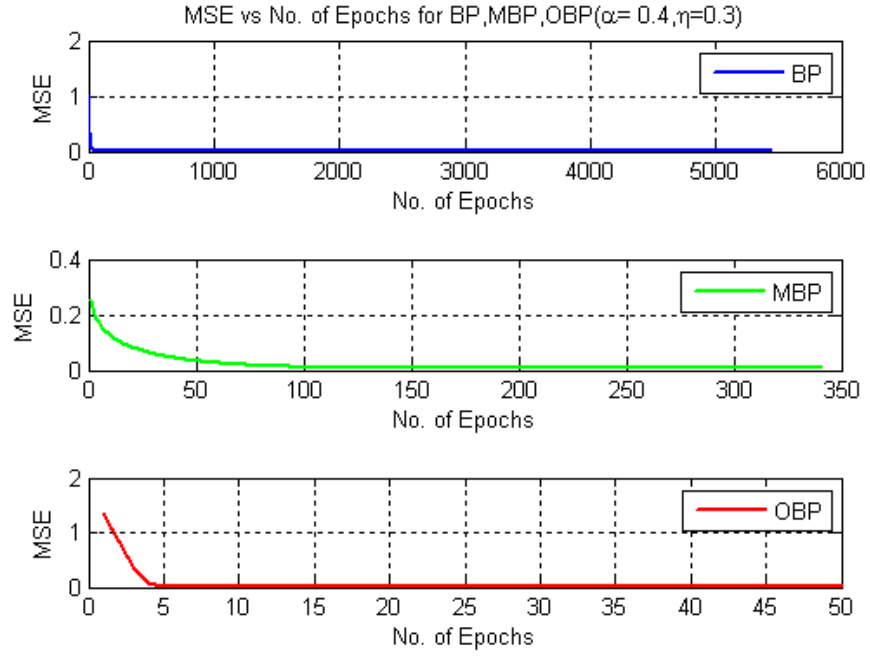


Figure 4.8: *Comparison of Training Speed of BP, MBP and OBP at α=0.3 , η = 0.4*

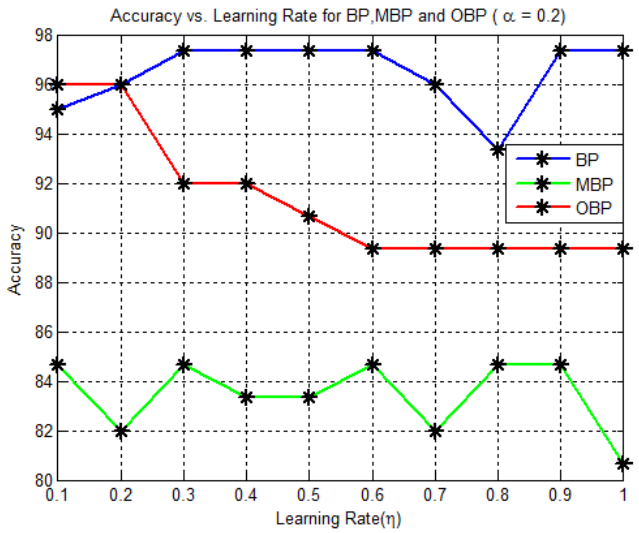


Figure 4.9: *Comparison of Accuracy of BP, MBP and OBP at α=0.2 for varying η*

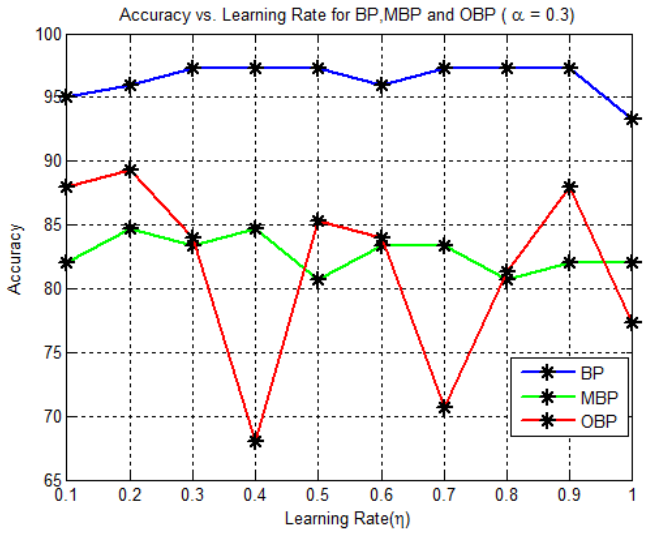


Figure 4.10: *Comparison of Accuracy of BP, MBP and OBP at α=0.3 for varying η*

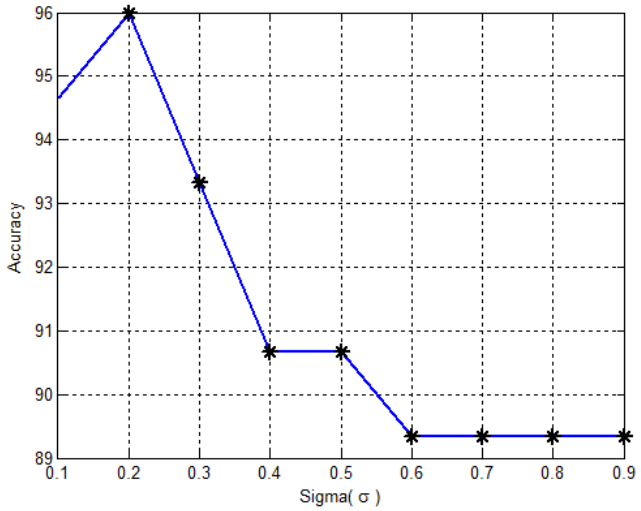


Figure 4.11: *Accuracy(%) vs. σ(sigma) for Probabilistic Neural Network*

Table 4.1: Table for no. of Epochs for different α and η of BackPropagation Algorithm

S. No.	**Momentum (α)**	**Learning Rate (η)**	**No. of Epochs in BP**
1	0.3	0.6	7200
2	0	0.1	17427
3	0.1	0.1	16552
4	0.3	0.1	14804
5	0.4	0.1	9808
6	0.5	0.1	20483
7	0.1	0.2	8585
8	0.2	0.2	6662
9	0.4	0.2	6274
10	0.2	0.3	5058
11	0.3	0.3	5337
12	0.4	0.3	5434
13	0	0.4	4838
14	0.1	0.4	3704
15	0.2	0.4	3615
16	0.3	0.4	5209
17	0.1	0.5	3092
18	0.2	0.5	3386
19	0.3	0.5	4239
20	0	0.6	4689
21	0.1	0.6	2689
22	0.2	0.6	3988
23	0	0.7	3220
24	0.1	0.7	2759
25	0.3	0.8	2565
26	0	0.8	2711
27	0.1	1	2848

Table 4.2: Table for Errors after 20000 Epochs for Backpropagation Algorithm

$\frac{\alpha\rightarrow}{\eta\downarrow}$	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
0.1	0.0098	0.0095	0.0095	0.0097	0.0098	0.0099	0.0101	0.0104	0.0098	0.0091
0.2	0.0094	0.0092	0.0090	0.0098	0.0098	0.0101	0.0107	0.0109	0.0101	0.0098
0.3	0.0093	0.0091	0.0099	0.0098	0.0099	0.0103	0.0107	0.0115	0.0102	0.0103
0.4	0.0095	0.0093	0.0099	0.0099	0.0101	0.0105	0.0110	0.0118	0.0111	0.0101
0.5	0.0094	0.0094	0.0095	0.0099	0.0103	0.0107	0.0113	0.0123	0.0112	0.0098
0.6	0.0095	0.0091	0.0092	0.0101	0.0105	0.0109	0.0115	0.0127	0.0114	0.0099
0.7	0.0098	0.0101	0.0098	0.0102	0.0106	0.0110	0.0117	0.0130	0.0123	0.0131
0.8	0.0093	0.0105	0.0099	0.0111	0.0107	0.0112	0.0120	0.0133	0.0136	0.0137
0.9	0.0097	0.0107	0.0111	0.0104	0.0109	0.0113	0.0121	0.0135	0.0142	0.0130
1	0.0099	0.0101	0.0105	0.0107	0.0109	0.0114	0.0124	0.0138	0.0110	0.01

Table 4.3: Table forAccuracy Testing of Backpropagation Algorithm

S.No.	**Momentum(α)**	**Learning Rate(η)**	**MSE**	**Accuracy (%)**
1	0.2	0.3	0.0091	97.33
2	0.2	0.4	0.0093	97.33
3	0.2	0.5	0.0094	97.33
4	0.2	0.6	0.0091	96.5
5	0.2	0.7	0.0101	97.33
6	0.2	0.8	0.0105	97.33
7	0.2	0.9	0.0107	97.33
8	0.3	0.2	0.0090	97.33
9	0.3	0.3	0.0099	97.22
10	0.3	0.4	0.099	96
11	0.3	0.5	0.0095	95.5
12	0.3	0.6	0.0092	96
13	0.3	0.7	0.0098	97.33
14	0.3	0.8	0.099	97.33
15	0.3	0.9	0.0101	97.33
16	0.4	0.3	0.0098	96
17	0.4	0.4	0.0099	97.33
18	0.4	0.5	0.0099	97.33
19	0.4	0.6	0.0101	96.5
20	0.4	0.7	0.0102	97.33
21	0.4	0.8	0.0111	96.33
22	0.4	0.9	0.0104	97.33

Table 4.4: Table for no. of Epochs for different α and η of Modified BackPropagation Algorithm

$\frac{\alpha\rightarrow}{\eta\downarrow}$	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
0.1	279	210	198	215	199	315	407	258	215	310
0.2	281	234	230	273	219	281	341	211	213	228
0.3	291	231	307	341	284	186	257	288	238	254
0.4	248	363	228	272	231	263	186	318	244	239
0.5	219	289	248	259	237	242	239	211	229	308
0.6	225	211	253	301	231	258	264	283	311	326
0.7	228	224	256	173	254	189	293	175	210	272
0.8	246	253	256	241	239	234	248	273	291	288
0.9	383	272	301	258	204	273	194	281	290	321
1	260	309	243	231	217	247	283	211	278	292

Table 4.5: Table forAccuracy Testing of Modified Backpropagation Algorithm

$\frac{\alpha\rightarrow}{\eta\downarrow}$	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
0.1	79.33	84.67	82	84.67	84.67	66.67	66.67	83.33	83.33	83.33
0.2	84.67	82	84.67	66.67	82	83.33	83.33	82	84.67	84.67
0.3	83.33	84.67	83.33	83.33	83.33	83.33	84.67	82	82	83.33
0.4	79.33	83.33	84.67	84.67	84.67	84.67	80.67	79.33	82	83.33
0.5	74.67	83.33	80.67	84.67	83.33	84.67	83.33	83.33	79.33	82
0.6	83.33	84.67	83.33	82	80.67	80.67	82	82	79.33	80.67
0.7	80.67	82	83.33	82	80.67	80.67	82	82	79.33	80.67
0.8	80.67	84.67	80.67	82	82	82	82	82	80.67	83.33
0.9	84.67	84.67	82	82	82	82	82	79.33	82	83.33
1	84.67	80.67	82	82	82	80.67	82	82	80.67	82

Table 4.6: Table for no. of Epochs for different α and η of Optical BackPropagation Algorithm

$\frac{\alpha\rightarrow}{\eta\downarrow}$	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
0.1	178	154	68	71	54	59	44	47	143	64
0.2	80	128	61	66	47	48	44	61	43	51
0.3	161	86	47	50	171	52	65	42	171	43
0.4	108	71	197	59	64	65	58	54	43	41
0.5	69	65	42	51	64	114	60	45	40	45
0.6	65	60	46	124	55	50	49	40	103	47
0.7	58	57	109	42	62	53	70	96	37	47
0.8	56	52	62	43	46	53	64	54	88	40
0.9	52	50	54	88	41	43	46	43	39	61
1	49	49	47	78	43	79	61	45	148	40

Table 4.7: Table forAccuracy Testing of Optical Backpropagation Algorithm

$\frac{\alpha\rightarrow}{\eta\downarrow}$	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**	**1**
0.1	94.67	96	88	88	89.33	84	85.33	89.33	69.33	89.33
0.2	86.67	96	89.33	84	88	88	89.33	69.33	85.33	84
0.3	88	92	84	85.33	81.33	85.33	89.33	88	77.33	89.33
0.4	82.67	92	68	85.33	82.67	82.67	76	85.33	90.67	89.33
0.5	89.33	90.67	85.33	82.67	69.33	92	66.67	89.33	85.33	86.67
0.6	90.67	89.33	84	66.67	82.67	78.67	88	86.67	68	85.33
0.7	89.33	89.33	70.67	85.33	66.67	85.33	70.67	70.67	85.33	85.33
0.8	89.33	89.33	81.33	84	84	76	88	92	69.33	90.67
0.9	89.33	89.33	88	66.67	85.33	66.67	86.67	66.67	90.67	88
1	89.33	89.33	77.33	68	82.67	66.67	80	88.67	66.67	77.33

Table 4.8: Table for Accuracy of Probabilistic Neural Networks for different values of σ(sigma)

S.No	σ	**Accuracy**(%)
1	0.1	94.67
2	0.2	96
3	0.3	93.33
4	0.4	90.67
5	0.5	90.67
6	0.6	89.33
7	0.7	89.33
8	0.8	89.33
9	0.9	89.33

Chapter 5

DISCUSSION

From the above graphs we observe that Backpropagation Algorithm gives the best accuracy but its learning speed is very slow. The no. of epochs required to train the neural network range from 2565 to 20000 and the accuracy ranges from 96% to 97.33%,i.e, the trade-off between speed and accuracy is quite low.

The Modified Backpropagation Algorithm, on the other hand, takes average time for learning but its accuracy is low. The no. of epochs required to train the neural network range from 173 to 407 and the accuracy ranges from 66.67% to 84.67%,i.e, the trade-off between speed and accuracy is average.

The Optical Backpropagation Algorithm has a very fast learning speed and gives good accuracy. The no. of epochs required to train the neural network range from 37 to 178 and the accuracy ranges from 66.67% to 96%,i.e, the trade-off between speed and accuracy is good.

Probabilistic Neural Network is very fast in terms of learning speed and gives good accuracy of 96% at σ =0.2.

Chapter 6

CONCLUSION

From the above results, graphs and discussion, it is concluded that Probabilistic Neural Network (PNN) is faster than the traditional Backpropagation Algorithm, Modified Backpropagation Algorithm and Optical Backpropation Algorithm in terms of learning speed and gave a good accuracy,i.e, has the best trade-off between speed and accuracy. So, for faster and accurate classification, Probabilistic Neural Networks can be used in many pattern classification problems.

REFERENCES:

1. Ibrahiem M.M.EI Emary and S. Ramakrishnan, "On the application of various probabilistic neural netorks in solving different pattern classification problems", World Applied Sciences Journal, 4(6):772 780, 2008.

2. Leila Fallah Araghi, Hamid Khaloozade , Mohammad Reza Arvan , "Ship Identification using probabilistic neural networks",Proceedings of the International Multi-Conference of Engineers and Computer Scientists 2009 Vol II , IMECS 2009, March 18 - 20, 2009, Hong Kong.

3. Walid A. Salameh, Mohammed A. OtairArab Princess Summaya, "Online Handwritten Character Recognition Using an Optical Backpropagation Neural Network", Issues in Informing Science and Information Technology, 787-795, 2005.

4. Saeid Iranmanesh, M. Amin Mahdavi , "A Differential Adaptive Learning Rate Method for Back-Propagation Neural Networks",World Academy of Science, Engineering and Technology 50 2009.

5. K.Matsuoka and J.YI, "backpropagation based on the logarithmic error function and elimination of local minima",pp. 1117-1122

6. Mohammed A. Otair and Walid A. Salameh , "Speeding Up Back-Propagation Neural Networks", Proceedings of the 2005 Informing Science and IT Education Joint Conference.

7. M Farrokhrooz, H.Keivani, H.Rastegar , "Marine Vessels Classification using Probabilistic Neural Networks", 2005 IEEE.

8. Khalaf khatatneh, Ibrahiem M.M El Emary and Basem Al- Rifai , "Probabilistic Artificial Neural Network For Recognizing the Arabic Hand Written Characters", Journal of Computer Science 2 (12): 879-884, 2006 ISSN 1549-3636.

9.Satish Kumar , "Neural Networks A Classroom Approach", TATA McGraw-Hill-ISBN 0-07-048292-6, pg-61,164-195,188..

10. S.Rajasekaran, G.A.Vijayalakshmi Pai , "Neural Network, Fuzzy Logic and Genetic Algorithm", Prentice Hall of India, pg-13-20.

11. Dr. M.H.Dunham, "Traffic Engineering with AIMD in MPLS networks," Data Mining, Introductory and Advanced Topics", Prentice Hall, 2002

12. D.E. Rumelhart, G.E. Hinton, and R.J. Williams , "Learning internal representation by error propagation", pages 318-362, 1986.

13. C.M. Bishop , "Neural networks for pattern recogntion", Oxford,1995.

14. R.O. Duda P.E.Hart, and D.G. Stork, "Pattern Classification ", Wiley Interscience Pulication, 2nd Edition ,2001.