ISSN:0975-9646
Naveen Kumar Sharma et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 2 (2) , 2011, 932-945

Performance Evaluation Analysis of MLP & DG-RBF Feed Forward Neural Networks for Pattern Classification of Handwritten English Curve Scripts

Naveen Kumar Sharma[1], S R Pande[2] and Manu Pratap Singh[3*]
1. *CET-IILM-AHL, Knowledge Park II, Greater Noida, India,*
2. *Department of Computer Science, SSES Amti's Science College, Congressnagar, Nagpur, India*
3. *Department of Computer Science, ICIS, Dr. B. R. A. University, Khandari, Agra, India*

***Abstract*- The purpose of this study is to evaluate the performance analysis of multilayer feed forward neural networks trained with back propagation algorithm & descent gradient Radial basis function network for the pattern classification of hand written curve script. This analysis has been done for handwritten text of three letters and for the individual English vowels. This analysis in the performance has been evaluated for the five different samples of handwritten English vowels and handwritten text of the three letters. Evaluation process is executed upon raw data in binary form and data based on extracted features (tangent values & density value) for each word & vowels. These characters are presented to the neural network for the training. Adjusting the connection strength and network parameters perform the training process in the neural network. The results of 3600 experiments indicate that the feed forward MLP performs accurately and exhaustively with imposed DG-RBF method.**

***Key words*-** Pattern Classification, Feed forward neural network, Back propagation algorithm, Radial Basis Function Neural Network.

I. INTRODUCTION

An artificial neural network (ANN) is a well-established technique for creating the artificial intelligence in the machine. This is an attempt to simulate the human behavior in the machine for the various pattern recognition tasks [1]. Neural networks consist of computer programmable objects called as neurons. These neurons are programmed to perform a simple mathematical function or to process a small portion of data. A neuron is interconnected with other neurons with the connection strength known as weight. These weights of the neural network are adjustable in nature to adept the behavior of input pattern information. Thus, by adjusting the weights of the network, the behavior of the neural network can be altered and controlled. This mechanism in neural network system is known as learning.

Neural networks have been used in a number of applications such as pattern recognition & classification [2, 3, 4, 5], remote sensing [6], dynamic modeling and medicine [7]. The increasing popularity of the neural networks is partly due to their ability to learn and generalization. Particularly, feed forward neural network makes no prior assumption about the statistics of input data and can construct complex decision boundaries [8]. This property makes neural networks, an attractive tool to many pattern classification problems such as hand written curve scripts [9, 10, 11].

The pattern classification of handwritten text and numerals has been a domain of great importance for the researchers due to the large availability of data and to make the processes work faster [12 – 14]. The employment of few systems like Optical character recognition [15] for Printed text and OMR [16] sheets for examination forms have reduced the cost of operation and also time to process this data. Such systems require large knowledge base and intelligent technology that can function correctly even when distorted/ modified input is present to it. For this purpose Artificial Neural Networks have been used to implement such systems.

Since last 5 decades various methods and technologies have been developed to use Artificial Neural Networks (ANN) in the domain of handwriting recognition [17]. The increasing popularity of the neural networks is partly due to their ability to learn and generalization in particular, the feed forward neural network refers to prior assumption about the statistics of input data and can construct complex decision boundaries [18].

Many systems have been developed by researchers for handwriting recognition using multi layer Perceptron (MLP) [19-22], Hidden Markov models and feed forward neural networks [23, 24, 25-27]. The literature is supplied with high precision recognition systems for already segmented handwritten numerals and characters [28-29]. However, research into the recognition of characters extracted from cursive and touching handwriting does not have the same measure of achievement [30]. One of the major problems faced when dealing with segmented, handwritten English character recognition is the indistinctness and illegibility of the characters. The accurate recognition of segmented characters is very important in the context of segmentation based, word recognition [31].

There are different types of architectures and designs for the neural networks, but here we discuss the most common one, i.e. feed forward manner. In a feed forward neural network the nodes are organized into layers; each "stacked" on each other. The neural network consists of an input layer of nodes, one or more hidden layers, and an output layer [32]. Each node in the layer has one corresponding node in the next layer, thus creating the stacking effect. The input layer's nodes consists with output functions those deliver data to the first hidden layers nodes. The hidden layer(s) is the processing layer, where all of the actual computation takes place. Each node in a hidden layer computes a sum based on its input from the previous layer (either the input layer or another hidden layer). The sum is then "compacted" by an output function (sigmoid function), which changes the sum down to more a limited and manageable range. The output sum from the hidden layers is passed to the output layer, which exhibits the final network result. Feed-forward networks may contain any number of hidden layers, but only one input and one output layer. A single-hidden layer network can learn any set of training data that a network with multiple layers can learn [33]. However, a single hidden layer may take longer to train.

There are numerous algorithms have been proposed to improve the back propagation learning algorithm. Since, the error surface may have several flat regions; the back propagation algorithm with fixed learning rate may be inefficient. In order to overcome with these problems, Vogel et. al. [34] and Jacobs [35] proposed a number of useful heuristic methods, including the dynamic change of the learning rate by a fixed factor and momentum based on the observation of the error signals. Yu et. al. proposed dynamic optimization methods of the learning rate using derivative Information [36]. Several other variations of back propagation algorithms based on second order methods have been proposed [37-41]. Artificial Neural Networks (ANNs) have played a wonderful role in achieving remunerable results. The one essential subset of ANN is Radial Basis Function Network and its generalization. The important aspect of the RBFN is the distinction between the techniques of updating the first and the second layers weights. Various techniques have been proposed in the literature for optimizing the Radial Basis functions such as unsupervised methods like selection of subsets of data points [42], orthogonal least square method [43], clustering algorithm [44], Gaussian mixture models [45] and with the supervised learning method [31].

The RBF network has one hidden layer of Gaussian functions, which are combined linearly by the output nodes. In early stage, the parameters of RBF networks were usually estimated in two phases: Gaussian parameter estimation by clustering and weight learning by error minimization. Since the clustering procedure does not consider the divisibility of patterns, the Gaussian parameters learned this way do not lead to good classification performance. A substantial improvement is to adjust all the parameters simultaneously by error minimization [46]. This makes the RBF network competitive with the MLP in classification accuracy.

In this paper, we consider two neural networks architectures (NN1 & NN2). The NN1 is trained with the conventional back propagation learning algorithm with incorporation of momentum terms & Doug's Momentum descent term [47]. The NN2 network architecture has been implemented with the Radial basis function [48] in the single hidden layer. This network incorporates the steepest gradient descent for weight updates. The performance of these two network architectures has been analyzed against three types of input patterns. These input patterns are in the form of binary image matrix, tangent values matrix for the handwritten curve scripts of three letters and the density values for the handwritten English vowels. The networks analyzed to figure out the network that exhibits higher performance results with greater efficiency. Every network is assessed based on the rate of convergence and speed of determination of the convergence weights for the every pattern. The experiments are conducted with 600 samples of English words of three characters and five sets of handwritten characters of English vowels. The significant improvement in the performance of the network has been achieved for the pattern classification of the handwritten vowels as well as for the recognition of handwritten words.

The next section presents the implementation of the neural network architecture with Radial basis function. The simulation design and algorithmic steps of the problem are represented in section 3. The experimental results and discussion are presented in section 4. Section 5 contents the conclusion of this paper and the future research directions.

II. IMPLEMENTATION OF THE RADIAL BASIS FUNCTION

There are various methods for classification problems [49] like the handwritten English characters recognition and each of them has pros and cons. The problem specification in handwritten recognition using ANN is basically the optimization problem. Methods of nonlinear optimization in ANNs have been studied for hundreds of years, and there is a huge literature on the subject in fields such as numerical analysis, operation research and statistical computing [50 – 51]. The multi layer neural networks (MLP) architecture has used for the constraint free non linear optimization for the various optimization problem. The pattern classification is a good example of this optimization. The MLP usually suffers with the convergence problem of local error surface and the use of second order gradient descent term in weight update has significantly improved the performance of MLP [52] but still there is no guarantee to find the global optimum. Global optimization for neural nets is especially difficult because the number of distinct local optima can be astronomical. Another important consideration in the choice of optimization algorithms is that neural nets are often ill-condition [53], especially when there are many hidden units. The algorithms that use only first-order information, such as steepest descent and standard back-propagation are notoriously slow for ill-conditioned problems. Generally speaking, the more use an algorithm makes of second-order information, the better it will behave under ill-conditioning. The following methods are listed in order of increasing use of second-order information: conjugate gradients, quasi-Newton, Gauss-Newton and Newton-Raphson. Due to the ill-conditioned nature of ANN used for handwriting recognition; it has also been proposed to evaluate the performance of the proposed network with the introduction of gradient descent of RBF. The Gradient Descent RBF methods have been proven to be most effective and fast convergence methods in the problem domain of supervised learning. In this section, we investigate a network structure related to the multi layer feed forward neural network (FFNN), implemented using the Radial Basis Function (RBF-MLP) which suffices the need of locally responsive neurons. If we interpret this real life phenomena in the domain of handwritten recognition then we can clearly see that local approximation based function can play important role in adjusting weights of inter mediate layers of an MLP designed for pattern recognition task and in the process of optimization of these locally responsive neurons play the role of building block of global minima.

The architecture and training methods of the RBF network are well known [54, 55, 56, 57, 58, 59] & well established. The Radial basis function network (RBFN) is a universal approximator with a solid foundation in the conventional approximation theory. The RBFN is a popular alternative to the MLP, since it has a simple structure and a much faster training process. The RBFN has its origin in performing exact interpolation of a set of data points in a multidimensional space [60]. The RBFN is having, network architecture similar to the classical regularization network [55], where the basis functions are the Green's functions of the Gram operator associated with the stabilizer. If the stabilizer exhibits radial symmetry, the basis functions are radially symmetric as well and an RBFN is obtained. From the approximation theory viewpoint, the regularization network has three following desirable properties [61, 62]:

1. It can approximate any multivariate continuous function on a compact domain to an arbitrary accuracy, given a sufficient number of units.
2. The approximation has the best-approximation property since the unknown coefficients are linear.
3. The solution is optimal in a way that it minimizes a function that measures how much it oscillates.

An RBFN is a three layer feed forward network that consists of one input layer, one hidden layer and one output layer as shown in Figure 1, each input neuron corresponds to a component of an input vector x. The hidden layer consists of K neurons and one bias neuron. Each node in the hidden layer uses an RBF denoted with $\phi(r)$, as its non-linear activation function.

The hidden layer performs a non-linear transform of the input and the output layer this layer is a linear combiner which maps the nonlinearity into a new space. The biases of the output layer neurons can be modeled by an additional neuron in the hidden layer, which has a constant activation function $\phi_0(r) = 1$. The RBFN can achieve a global optimal solution to the adjustable weights in the minimum MSE range by using the linear optimization method. Thus, for an input pattern x, the output of the jth node of the output layer can define as;

$$y_j(x) = \sum_{k=1}^{K} w_{kj}\phi_k(\|x_i - \mu_k\|) + w_{0j} \quad (1)$$

for $j = (1,2,.....,,M)$ where $y_j(x)$ is the output of the j^{th} processing element of the output layer for the RBFN , w_{kj} is the connection weight from the kth hidden unit to the jth output unit , μ_k is the prototype or centre of the kth hidden unit.

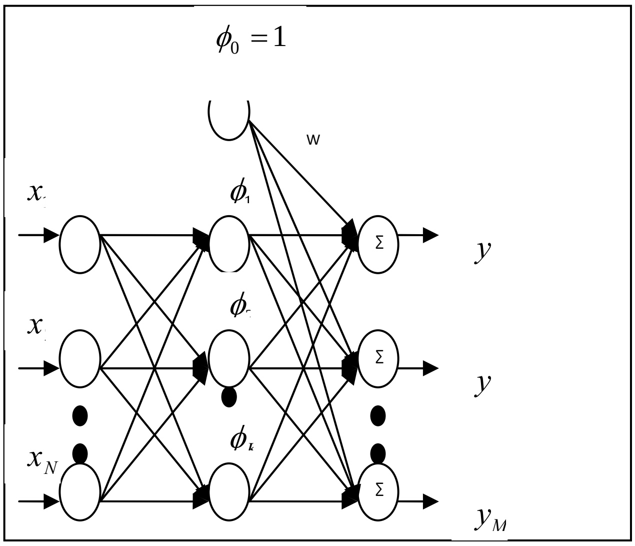


Fig. 1 Architecture of the RBFN. The input layer has N nodes; the hidden and the output layer have K and M neurons, respectively. $\phi_0(x) = 1$, corresponds to the bias.

The Radial Basis Function $\phi(.)$ is typically selected as the Gaussian function that can be represented as:

$$\phi_k(x_l) = \exp(-\frac{\|x_l - \mu_k\|^2}{2\sigma_k^2})$$

for $k = (1,2,.....,,K)$ (2)

and 1 for $k = 0$ (bias neuron)

Where x is the N- dimensional input vector, μ_k is the vector determining the centre of the basis function ϕ_k and σ_k represents the width of the neuron. The weight vector between the input layer and the kth hidden layer neuron can consider as the centre μ_k for the feed forward RBF neural network.

Hence, for a set of L pattern pairs $\{(x_l, y_l)\}$, (1) can be expressed in the matrix form as

$$Y = w^T \phi \quad (3)$$

where $W = [w_1,.............w_m]$ is a KxM weight matrix, $w_j = (w_{0j},...........w_{kj})^T$, $\phi = [\phi_0,..........\phi_k]$ is a K x L matrix, $\phi_{l,k} = [\phi_{l,1},..........\phi_{l,k}]^T$ is the output of the hidden layer for the lth sample, $\phi_{l,k} = \phi(\|x_l - c_k\|)$, $Y = [y_1, y_{2,}......y_m]$ is a M x L matrix and $y_{lj} = (y_{l1,}.....y_{lm})^T$.

The important aspect of the RBFN is the distinction between the rules of the first and second layers weights. It can be seen [47] that, the basis functions can be interpreted in a way, which allows the first layer weights (the parameters governing the basis function), to be determined by unsupervised learning. This leads to the two stage training procedure for RBFN. In the first stage the input data set $\{x^n\}$ is used to determine the parameters of the basis functions. The basis functions are then keep fixed while the second – layer weights are found in the second phase of training. There are various techniques have been proposed in the literature for optimizing the basis functions such as unsupervised methods like selection of subsets of data points [63], orthogonal least square method [64], clustering algorithm [55], Gaussian mixture models [65] and with the supervised learning method.

It has been observed [46] that the use of unsupervised techniques to determine the basis function parameters is not in general an optimal procedure so far as the subsequent supervised training is concerned. The difficulty with the unsupervised techniques arises due to the setting up of the basis functions, using density estimation on the input data and takes no consideration for the target labels associated with the data. Thus, it is obvious that to set the parameters of the basis functions for the optimal performance, the target data should include in the training procedure and it reflects the supervised training. Hence, the basis function parameters for regression can be found by treating the basis function centers and widths along with the second layer weights, as adaptive parameters to be determined by minimization of an error function. The error function has considered in equation (1) as the least mean square error (LMS). This error will minimize along the decent gradient of error surface in the weight space between hidden layer and the output layer. The same error will minimize with respect to the Gaussian basis function's parameter as defined in equation (2.2). Thus, we obtain the expressions for the derivatives of the error function with respect to the weights and basis function parameters for the set of L pattern pairs (x^l, y^l) as; where $l = 1$ to L.

$$\Delta w_{jk} = -\eta_1 \frac{\partial E^l}{\partial w_{jk}} \qquad (4)$$

$$\Delta \mu_k = -\eta_2 \frac{\partial E^l}{\partial \mu_k} \qquad (5)$$

and $$\Delta \sigma_k = -\eta_3 \frac{\partial E^l}{\partial \sigma_k} \qquad (6)$$

here, $$E^l = \frac{1}{2}\sum_{j=1}^{M}(d_j^l - y_j^l)^2$$

and $$y_j^l = \sum_{k=1}^{K} w_{jk}\phi_k(\|x^l - \mu_k^l\|) \qquad (7)$$

and $$\phi_k(\|x^l - \mu_k^l\|) = \exp(-\frac{\|x^l - \mu_k^l\|^2}{2\sigma_k^2})$$

Hence, from the equation (4) we have,

$$\Delta w_{jk} = -\eta_1 \frac{\partial E^l}{\partial w_{jk}} = -\eta_1 \frac{\partial E^l}{\partial y_j^l}.\frac{\partial y_j^l}{\partial w_{jk}} = -\eta_1 \frac{\partial E^l}{\partial y_j^l}.\phi_k\|x^l - \mu_k^l\|$$

or $$\Delta w_{jk} = -\eta_1 \frac{\partial E^l}{\partial s_j^l(y_j^l)}.\frac{\partial s_j^l(y_j^l)}{\partial y_j^l}.\exp(-\frac{(\|x^l - \mu_k^l\|)^2}{2\sigma_k^2})$$

$$= \eta_1 \sum_{j=1}^{M}(d_j^l - y_j^l).\dot{s}_j^l(y_j^l).\sum_{k=1}^{K}\exp(-\frac{(\|x^l - \mu_k^l\|)^2}{2\sigma_k^2}$$

So,

$$\Delta w_{jk} = \eta_1 \sum_{j=1}^{M}\sum_{k=1}^{K}(d_j^l - y_j^l)\dot{s}_j^l(y_j^l)\exp(-\frac{(\|x^l - \mu_k^l\|)^2}{2\sigma_k^2}) \qquad (8)$$

Now, from the equation (6) we have

$$\Delta \mu_{ki} = -\eta_2 \frac{\partial E^l}{\partial \mu_{ki}} = -\eta_2 \frac{\partial E^l}{\partial y_j^l}.\frac{\partial y_j^l}{\partial \mu_{ki}}$$

$$= -\eta_2 \frac{\partial E^l}{\partial y_j^l}.w_{jk}.\exp(-\frac{(\|x_i^l - \mu_{ki}^l\|^2)}{2\sigma_k^2}).(\frac{x_i^l - \mu_{ki}^l}{\sigma_k^2})$$

Or

$$\Delta \mu_{ki} =$$

$$\eta_2 \sum_{j=1}^{M}\sum_{k=1}^{K}(d_j^l - y_j^l)\dot{s}_j^l(y_j^l)w_{jk}.\exp(-\frac{(\|x_i^l - \mu_{ki}^l\|)^2}{2\sigma_k^2})$$

$$(\frac{x_i^l - \mu_{ki}^l}{\sigma_k^2}) \qquad (9)$$

Now, from the equation (6) we have

$$\Delta \sigma_k = -\eta_3 \frac{\partial E^l}{\partial \sigma_k} = -\eta_3 \frac{\partial E^l}{\partial y_j^l}.\frac{\partial y_j^l}{\partial \sigma_k}$$

$$= -\eta_3 \frac{\partial E^l}{\partial y_j^l}.w_{jk}\ \exp(-\frac{(\|x_i^l - \mu_{ki}^l\|)^2}{2\sigma_k^2})\frac{\|x_i^l - \mu_{ki}^l\|^2}{\sigma_k^3}$$

or,

$$\Delta \sigma_k = \eta_3 \sum_{j=1}^{M}\sum_{k=1}^{K}(d_j^l - y_j^l).\dot{s}_j^l(y_j^l).w_{jk}.$$

$$\exp(-\frac{(\|x_i^l - \mu_{ki}^l\|)^2}{2\sigma_k^2})\frac{\|x_i^l - \mu_{ki}^l\|^2}{\sigma_k^3} \qquad (10)$$

So that, we have from equations (8), (9) & (10) the expressions for change in weight vector & basis function parameters to accomplish the learning in supervised way. The adjustment of the basis function parameters with supervised learning represents a non-linear optimization problem, which will typically be computationally intensive and may be prove to finding local minima of the error function. Thus, for reasonable well-localized RBF, an input will generate a significant activation in a small region and the opportunity of getting stuck at a local minimum is small. Hence, the training of the network for L pattern pair i.e. (x^l, y^l) will accomplish in iterative manner with the modification of weight vector and basis function parameters corresponding to each presented pattern vector. The parameters of the network at the m^th step of iteration can express as;

$$w_{jk}(m)=w_{jk}(m-1)+\eta_1\sum_{j=1}^{M}\sum_{k=1}^{K}(d_j^l-y_j^l)\dot{s}_j^l(y_j^l).\exp\left(-\frac{(\|x_i^l-\mu_{ki}^l\|)^2}{2\sigma_k^2}\right) \quad (11)$$

$$\mu_{ki}(m)=\mu_{ki}(m-1)+\eta_2\sum_{j=1}^{M}\sum_{k=1}^{K}(d_j^l-y_j^l)\dot{s}_j^l(y_j^l).w_{jk}.\phi_k(x_i^l).\left(\frac{x_i^l-\mu_{ki}^l}{\sigma_k^2}\right) \quad (12)$$

$$\sigma_k(m)=\sigma_k(m-1)+\eta_3\sum_{j=1}^{M}\sum_{k=1}^{K}(d_j^l-y_j^l)\dot{s}_j^l(y_j^l).w_{jk}.\phi_k(x_i^l).\frac{\|x_i^l-\mu_{ki}^l\|^2}{\sigma_k^3} \quad (13)$$

where η_1, η_2 & η_3 are the coefficient of learning rate.

The discussed gradient decent approach for implementation of RBFNNs system is incremental learning algorithm in which the parameters update for each example (x^l, y^l). The RBFNNs trained by the gradient-decent method is capable of providing the equivalent or better performance compared to that of the multi layer feed forward neural network trained with the back propagation. The gradient decent method is slow in convergence since it cannot efficiently use the locally tuned representation of the hidden layer units. When the hidden unit receptive fields, controlled by the width σ_k are narrow for a given input only a few of the total number of hidden units will be activated and hence only these units need to be updated. Thus, there is no guarantee that the RBFNN remains localized after the supervised learning [55]. As a result the computational advantage of locality is not utilized. Indeed, in numerical simulations it is found that the subset of the basis functions may evolve to have very broad responses. It has been realized that some of the main advantages of the radial basis function network, is fast two-stage training and interpretability of the hidden unit representation.

Hence, among the neural network models, RBF network seems to be quit effective for pattern recognition task such as handwritten character recognition. Since it is extremely flexible to accommodate various and minute variations in data. Now, in the following subsection we are presenting the simulation designed and implementation details of redial basis function worked as a classifier for the handwritten English vowels and recognition of handwritten English curve scripts of three letters.

III. SIMULATION DESIGN AND IMPLEMENTATION DETAILS

The experiments described in this section were designed to evaluate the performance of feed forward neural network when evolved with the back propagation algorithm for MLP & RBF network with decent gradient method. To accomplish this task two neural networks are considered namely NN1 and NN2. NN1 is the conventional MLP and NN2 is the MLP with RBF imposed in its learning sequence. The problem domain has been divided into two parts. In the first part the task associated to the neural networks in both experiments was to accomplish the training of the handwritten English language vowels in order to generate the appropriate classification. For this, first we obtained the scanned image of five different types of samples of handwritten English language vowels as shown in Figure 2. After collecting these samples, we partitioned an English vowel image in to four equal parts and calculated the density of the pixels, which belong to the central of gravities of these partitioned images of an English vowel. Like this, we will get 4 densities from an image of handwritten English language vowel, which we use to provide the input to the feed forward neural network. We use this procedure of generating input for a feed forward neural network with each sample of English vowel scanned images.

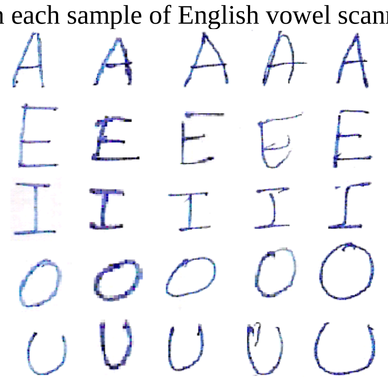
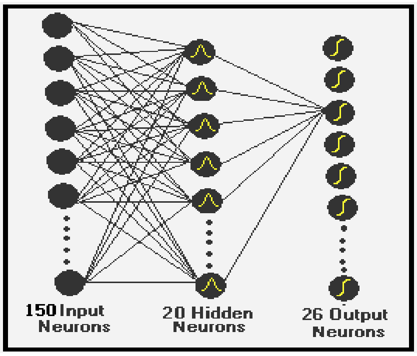


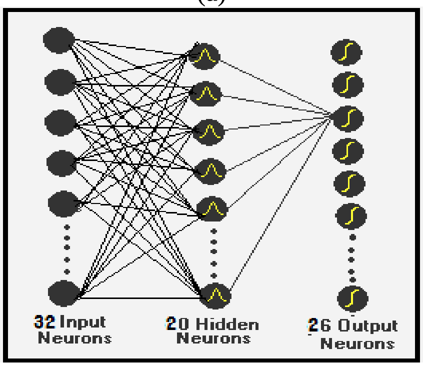
Fig 2 Scanned images of five different samples of handwritten English language vowels.

In the second part the complete set of experiments can be sub-divided into two segments. In the first segment input vector is of the dimensions 150x3, and for the second segment of experiment, the input vector is of 32x1. For the first segment of experiments, all three systems are simulated using a neural network system that consist of 150 neurons in input layer, 20 neurons in hidden layer and 26 output neurons. Each network has 150 input neurons that are equivalent to the input character's size as we have resized every character into a binary matrix of size 15x10. Character's image is achieved by applying the segmentation technique [52]. The distinguishing factors among FF-MLP (NN1) and RBF-MLP (NN2) are that in the case of FF-MLP the network contains Log-Sigmoid transfer functions whereas RBF-MLP contains RBF transfer function as shown in Fig. 3.

For the second set of experiments all network arrangements are having 32 input neurons, 20 hidden neurons and 26 output neurons. The transfer functions used to propagate the weight update among the layers are same as discussed above. The 32 input neurons correspond to 32 tangent values of each word's image. The 26 output neurons correspond to 26 letters of English alphabet. The number of hidden neurons is directly proportional to the system resources. The bigger the number more the resources are required. The number of neurons in hidden layers is kept 20 for optimal results. The 625 word samples are gathered form 51 subjects of different ages including male and female for the input patterns. After the preprocessing module 600 input samples were considered for training. Each sample was presented to the network 6 times (2 types of input patterns for each sample multiplied by 3 network architectures) thus 3600 experiments have been conducted.



(a)



(b)

Fig. 3 Network architecture of MLP with RBF in Hidden Layer for (a) First set of experiments where binary input in form of 150x3 matrix is used. And (b) Second set of experiments where tangent values based input in the form of 32x1 matrix is used

A. *Experiments*

As we have mentioned that two sets of experiments were executed. In each of the sets same type of network architecture was used. The problem domain has been divided in two phases. In the first phase the performance of both the network architectures have evaluated for the classification of handwritten English vowels and in the second phase the performance has been evaluated for the recognition of handwritten curve scripts of three letters. The parameters used for both experiments in both the problem domains are described in Table 1 and 2.

TABLE I
PARAMETERS USED FOR BACK PROPAGATION ALGORITHM

Parameter	**Value**
Back propagation learning Rate (η)	0.1
Momentum Term (α)	0.9
Doug's Momentum Term $\left(\frac{1}{1-(\alpha)}\right)$	$\left(\frac{1}{1-(\alpha)}\right)$
Adaption Rate (K)	3.0
Minimum Error Exist in the Network $(MAXE)$	0.00001
Initial weights and biased term values	Randomly Generated Values Between 0 and 1

TABLE II
PARAMETERS USED FOR DECENT GRADIENT -RBF ALGORITHM.

Parameter	**Value**
Back propagation learning Rate (η)	0.1
Momentum Term (α)	0.9
Doug's Momentum Term $\left(\frac{1}{1-(\alpha)}\right)$	$\left(\frac{1}{1-(\alpha)}\right)$
Adaption Rate (K)	3.0
Spread parameter σ	1.0
Mean of inputs(c)	Between maximum & minimum values
Minimum Error Exist in the Network $(MAXE)$	0.00001
Initial weights and biased term values	Randomly Generated Values Between 0 & 1

B. *RBF Implementation in the Neural Network Architecture for the first phase*

The first neural network (NN1) structural design was based on feed forward multilayer generalized perceptron. Four input units have been used, with one hidden layers of six numbers of neurons and five numbers of neurons in output layer. The second neural network (NN2) structural design was also based on a completely connected feed forward multilayer generalized perceptron. But four input units have been used, with single hidden layer of six neurons and five neurons in output layer. The NN1 network is employing the sigmoid function for generating the output signal from the processing elements of all the hidden layers and output layer. The NN2 is using the same sigmoid function for the processing elements of output layer, but the Gaussian form of radial basis function is used for the hidden layer elements. The hidden layers were employed to investigate the effects with back propagation and decent gradient-RBF would have on the hyper plane. The MLP network has a single output layer with the following activation and output functions for the pattern vector (x_1, y_1)

$$y_j^l = \sum_{k=0}^{K} w_{kj} s_h^l (q_h^l) \qquad (14)$$

and, $s_j^l(y_j^l) = f[y_j^l] = f[\sum_{k=0}^{K} w_{kj} s_h^l (q_h^l)]$ (15)

for $j = (1,2,.....,,M)$ and $l = (1,2,....,L)$,

where function $f[y_i^l]$ can define as,

$$s_j^l(y_j^l) = \frac{1}{1+e^{-Ky_j^l}} \qquad (16)$$

Now, similarly, the output and activation value for the neurons of hidden layers and input layer can be written as,

$$q_h^l = \sum_{i=0}^{N} w_{ik} x_i$$

and, $s_h^l(q_h^l) = f[q_h^l] = f[\sum_{i=0}^{N} w_{ik} x_i]$,

for $h = (1,2,......,K)$ (17)

$$s_h^l(q_h^l) = (\frac{1}{1+e^{-Kq_h^l}}) \qquad (18)$$

In the Back propagation learning algorithm the change in weights are being done according to the calculated error in the network, after each, iteration of training. The change in weights and error in the network can be calculated as,

$$\Delta w_{kj}(t+1) = -\eta \sum_{k=1}^{K} \frac{\partial E^l}{\partial w_{kj}} + \alpha \Delta w_{kj}(t) + \frac{1}{1-(\alpha \Delta w_{kj}(t))} \qquad (19)$$

$$\Delta w_{ik}(t+1) = -\eta \sum_{i=1}^{N} \frac{\partial E^l}{\partial w_{ik}} + \alpha \Delta w_{ik}(t) + \frac{1}{1-(\alpha \Delta w_{ik}(t))} \qquad (20)$$

$$E^l = \frac{1}{2} \sum_{l=1}^{L} \left(d^l - s^l(y^l) \right)^2 \qquad (21)$$

Where $\left(d^l - s^l(y^l) \right)^2$ is the squared difference between the actual output value and the target output value of output layer for pattern l. Here, we have used the doug's momentum term [47] with momentum descent term for calculating the change in weights in eqn. (19) & (20). Doug's momentum descent is similar to standard momentum descent with the exception that the pre-momentum weight step vector is bounded so that its length cannot exceed 1 (one). After the momentum is added, the length of the resulting weight change vector can grow as high as 1 / (1 - momentum). This change allows stable behavior with much higher initial learning rates, resulting in less need to adjust the learning rate as training progresses.

Now, in the decent gradient learning for the RBF network the change in weights and basis function parameters can be computed as;

$$\Delta w_{jk}(t+1) = \eta_1 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l).\dot{s}_j^l(y_j^l) \exp(-\frac{\left\| x_i^l - \mu_{ki}^l \right\|^2}{2\sigma_k}) + \alpha_1 \Delta w_{jk}(t) + \frac{1}{1-(\alpha_1 \Delta w_{jk}(t))} \qquad (22)$$

$$\Delta \mu_{ki}(t+1) = \eta_2 \sum \sum (d_j^l - y_j^l).\dot{s}_j^l(y_j^l).w_{jk}.\exp(-\frac{\left\| x_i^l - \mu_{ki}^l \right\|^2}{2\sigma_k^2}).(\frac{x_i^l - \mu_{ki}^l}{\sigma_k^2}) + \alpha_2 \Delta \mu_{ki}(t) + \frac{1}{1-(\alpha_2 \Delta \mu_{ki}(t))} \qquad (23)$$

and

$$\Delta \sigma_k(t+1) = \eta_3 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l).\dot{s}_j^l(y_j^l).w_{jk}.\exp(-\frac{\left\| x_i^l - \mu_{ki}^l \right\|^2}{2\sigma_k^2}).(\frac{x_i^l - \mu_{ki}^l}{\sigma_k^3}) + \alpha_3 \Delta \sigma_k(t) + \frac{1}{1-(\alpha_3 \Delta \sigma_k(t))} \qquad (24)$$

Here, again we are using the Doug's Momentum Term with momentum decent term for calculating the change in weights and basis function parameters. The redial basis function network has the single output and hidden layer with the following output functions for the pattern vector (x^l, y^l).

$$y_j^l = \sum_{k=1}^{K} w_{jk}\phi_k(\|x_i^l - \mu_{ki}^l\|) \quad (25)$$

and

$$s_j^l(y_j^l) = f[y_j^l] = f[\sum_{k=1}^{K} w_{jk}\phi_k(\|x_i^l - \mu_{ki}^l\|)] \quad (26)$$

for $j = (1,2,....,M)$ & $i = (1,2,...,N)$

Where function $f[y_j^l]$ can define as;

$$s_j^l(y_j^l) = \frac{1}{1+e^{-ky_j^l}}$$

and $$\phi_k(\|x_i^l - \mu_{ki}^l\|) = \exp(-\frac{\|x_i^l - \mu_{ki}^l\|^2}{2\sigma_k^2}) \quad (27)$$

C. RBF Implementation in the Neural Network Architecture for the Second phase

In this phase of the problem domain we consider both test and training patterns. Handwriting word samples from 51 people of different ages and genders mostly university and school students, were collected. Each writer was asked to write 15 sample characters in blocks made in the request form. Each of the 51 forms was scanned and word's images were extracted.

The input patterns for first set of experiments are prepared based on method discussed in [52]. A brief of this method is:-

1) Segment each word's image using vertical segmentation technique.
2) Reshape each character's image in to 15x10 binary matrix.
3) Resize the image into 150x1 matrix.
4) Repeat 2 and 3 for all three characters and club together in 150x3 matrix to form a sample.
5) Repeat 1-4 for all words to create 600 samples.

The input sample for one word from above referred technique can be depicted as in Fig. 4.

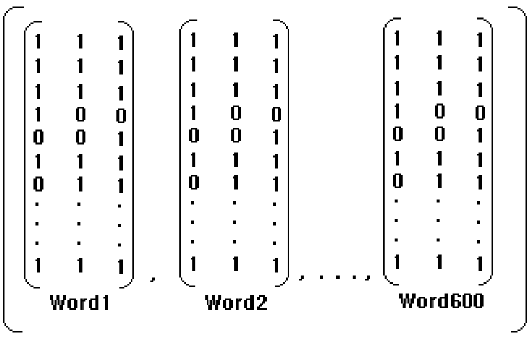


Fig 4 Input sample for the first set of experiments.

Word's image is binarized and resized to 280x560. Each image is then divided into 32 equal sub-images of the size 70x70. For better understanding the (2, 2) nd sub-division is shown in the following Fig. 5.

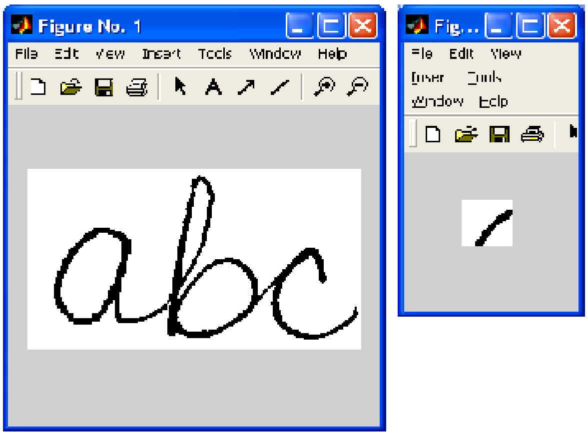


Fig 5 Dividing the character's image into 32 equal parts.

After this division, the binary sub-image is vertically (Column Based) traced to fit a liner curve. During the vertical reading of this sub-image we find black pixels, for these black pixels row numbers are summed and divided by total number of black pixels in a particular column.

$$AvgRowNum_i = (1/n) * \sum R_{ij} \quad (28)$$

here i is the column number being traced, R_{ij} is the row number in this column that contains the black pixel. This way we achieve the average row number against the column number being traced. The process is explained in the Fig. 6 shown below.

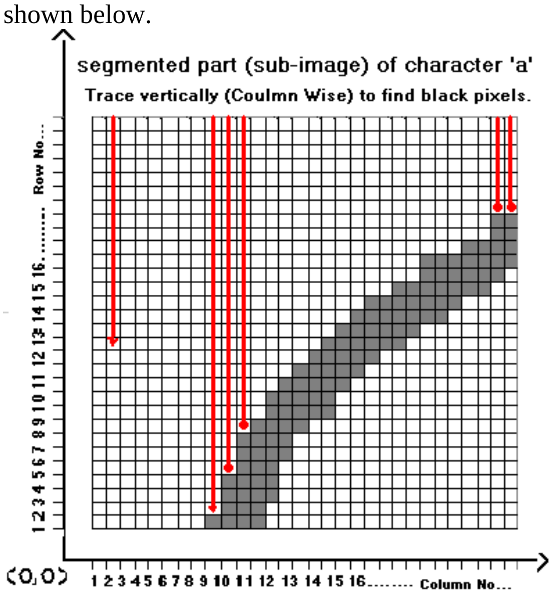


Fig. 6 The Column-Based tracing of a binary sub-image. This figure is just a pictorial presentation to show the process of vertical tracing.

The output of this vertical (i.e. column based) tracing is a set of ordered pairs of (Col, AvgRowNum). "Col" represents the number of column number where at-least one black pixel is found. AvgRowNum is, as already explained in previous section is the simple average of all row numbers where the black pixel is found. This data is useful to draw a straight line on the segmented image of the handwritten character. This output set can be shown in following Table 3. This table should be read from left to right (row wise) to get knowledge about data.

TABLE III
THE RESULT TABLE OF COLUMN BASED TRACING OF SEGMENTED CHARACTER WHERE “COLUMN NO” REPRESENTS COLUMN NUMBER WHERE BLACK PIXEL IS FOUND AND CORRESPONDING CALCULATED AVGROWNUMBER.

Column No	24	25	26	27	28	29	30	31
AvgRowNum	2	4.5	5	6	6.5	8	8.5	9
Column No	32	33	34	35	36	37	38	39
AvgRowNum	9	10.5	12.5	14.5	16	17.5	18	19.5
Column No	40	41	42	43	44	45	46	47
AvgRowNum	19.5	21	22.5	24.5	26	26.5	27.5	29.5
Column No	48	49	50	51	52	53	54	55
AvgRowNum	30.5	30.5	31.5	33	35	36	36.5	38.5
Column No	56	57	58	59	60	61	62	63
AvgRowNum	39	40	40	40.5	42	42	43.5	44.5
Column No	64	65	66	67	68	69	70	
AvgRowNum	45	46.5	47	47	47.5	48	48	

After the vertical tracing of the image a liner curve is fitted on this data, and tangent of this liner curve i.e. line is computed. This process is followed for all 32 segments.

For each of the character images, the following algorithm was applied to get 32 tangent values:-

Algorithm #1 : Pattern Preparation

Input: Character’s Image

1. Resize the image ***(I)*** into 280 X 560 pixels, using the K-Nearest Neighborhood algorithm.
2. Divide the image ***(Iresized)*** into 32 equal sub-images using standard IMCROP method of MATLAB. As shown in Figure 5.
3. Covert the sub-image ***(ISub1)*** into binary form ***(IBin1)***
4. Trace ***IBin1*** vertically (Column - wise) to get the black pixel.
    4.1 Add row number to SumRowCount and Increment *RowCount* by one if a black pixel is found. And store corresponding Column Number in an Array *ArrCoulmnNum.*
    4.2 Calculate the average *RowNum* by dividing *SumRowCount* with *RowCount* when row number is reached to max i.e. 70 and *RowCount* is non-zero.
    4.3 Store non zero values of *RowNum* in an array *ArrAvgRowNum* as shown in Table **I**.
5. Repeat step 4 till the all columns are read.
6. Fit a liner polynomial on (*ArrAvgRowNum, ArrCoulmnNum).*
7. For sample test Column Numbers 0:5:70 get the values from polynomial. From these values calculate Tangent value for this polynomial. Using the formula :-
   Tangent T = (Y2-Y1)/(X2-X1)
8. Repeat steps 3-7 for each of 32 sub-images and arrange the obtained tangent values in a matrix of size 32x1.

Output: 32 tangent values corresponding to the character’s image

The above mentioned algorithm when applied to this sub-image we retrieve both tangent value and corresponding liner curve. The graphical representation of the computed data from Table III and liner curve is shown in following diagram from MATLAB. The tangent of the liner curve is computed from the straight line as shown in Fig. 7.

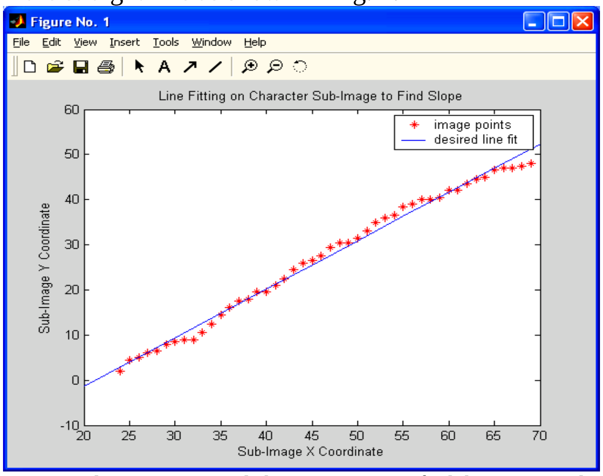


Fig 7 Fitting a line on segmented character image to find the tangent value the red colored values are taken from Table 3 and blue line is derived from the fitted polynomial.

The tangent values achieved from above mentioned algorithm for each word of the sample collection form. Each input sample comprises of 32 tangent values for each word. This is best depicted in following matrix where Wd1_Tan1 to Wd1_Tan32 are tangent values for First Word, Wd2_Tan1 to Wd2_tan32 are tangent values for second word and so on for all 600 words.

TABLE IV
INPUT SAMPLE PREPARED FROM TANGENT VALUES OF INDIVIDUAL WORDS.

Wd1_Tan1	Wd2_Tan1	Wd3_Tan1	•••	Wd600_Tan1
Wd1_Tan2	Wd2_Tan2	Wd3_Tan2	•••	Wd600_Tan2
Wd1_Tan3	Wd2_Tan3	Wd3_Tan3	•••	Wd600_Tan3
Wd1_Tan4	Wd2_Tan4	Wd3_Tan4	•••	Wd600_Tan4
Wd1_Tan5	Wd2_Tan5	Wd3_Tan5	•••	Wd600_Tan5
⋮	⋮	⋮		
Wd1_Tan32	Wd2_Tan32	Wd3_Tan32		Wd600_Tan32

The simulation program, which we have been developed in MATLAB 6.5, for testing these two networks for handwritten English language vowels classification problem and the recognition of handwritten curve scripts of three letters, generates initial weights randomly through its random generator but the same set of weights have been used for both the network architectures. So the epochs for the algorithms will be different every time with the same network structure and the same training data set.

IV. RESULTS AND DISCUSSION

A. Results for First phase of problem domain (English Vowels)

The result presented in this section are demonstrating the large significant difference exist between the performance of

BPNN and DG-RBF for handwritten English language vowels classification problem. The simulated results for this problem have been considered for the 5 trails with both algorithms up to maximum limit of 50000 iterations. All the results of 5 trials contain five different types of handwritten samples for each English vowels character. The training has been performed in such a way that repetition of same input sample for a character cannot be happen simultaneously, i.e. if we have trained our network with a input sample of a character then next training cannot be happen with the other input sample of the same character. This input sample will appear for training after other sample of other characters training. We have considered the mean of performance with their best case of convergence for all the trails. The comparative results are presented in Table V and the Fig. 7 is representing the comparison charts designed on the basis of values available in the table.

TABLE V
RESULTS FOR CLASSIFICATION OF HANDWRITTEN ENGLISH VOWELS USING BACK PROPAGATION FOR MLP AND DECENT GRADIENT WITH RBF NETWORK

Characters	Back propagation Epochs				
	Sample 1	Sample 2	Sample 3	Sample 4	Sample 5
A	558	0.4	0.4	0.4	0.4
E	3183	0.4	0.4	0.4	0.4
I	9800	0.4	0.4	0.4	0.4
O	1391	0.4	0.4	564	0.4
U	367	1848	0.4	2369	0.4
Characters	**DG-RBF Epochs**				
	Sample 1	**Sample 2**	**Sample 3**	**Sample 4**	**Sample 5**
A	1421	1972	1005	3836	6419
E	1641	124	1020	747	647
I	2416	1808	3239	860	76
O	6054	82	92	3352	3503
U	3494	475	13819	839	2924

It can observe from the results of table and graph that BPNN has converged conversing approximately for the 20 percent cases but the RBFN has converged for 75 percent cases.

The table is also showing some real numbers. These entries represents the error exit in the network after executing the simulation program up to 50000 iterations i.e. up to 50000 iterations the algorithm could not converge for a sample of a hand written English language vowels into the feed forward neural network.

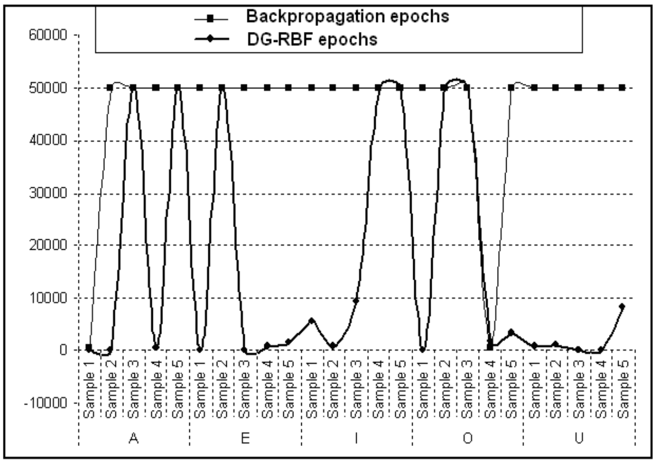


Fig 8 The Comparison Chart for Handwritten English Vowels Classification Epochs for two learning algorithms.

***B** Results for Second phase of problem domain (Three letter curve scripts)*

In this phase of the problem two sets of experiments were executed. In each of the sets same type of network architecture was used. In the first experiment set we have used conventional MLP and this network was trained and evaluated with two types of input patterns (tangent values and binary matrix). This process was executed three times for all two types of networks (MLP i.e. NN1, RBF-MLP i.e. NN2). In total the number of experiments conducted for training of all of the networks was 3600 including 1200-experiments for each type of network. The testing of the performance of networks was done in 300 experiments. Training and testing samples of both kinds (Binary format 150X3 and tangent values 32X1) for a particular word; when presented to the Networks; yielded 6 sets of data. The performance of the particular network has been evaluated based on the comparison done for same samples of data with other networks. For testing of the network, 5 test samples were created by randomly selecting character images from any of the 600 samples. Thus the network was trained with 600 different sets of input patterns. The following tables (table 6 and 7) contain epoch average of 10 iterations for 600 samples, thus only 60 readings have been mentioned. Sample 1 depicts average epoch value for sample 1 to sample 10; Sample 2 depicts average epoch value for sample 11 to sample 20 and so on.

Each sample has been presented to two networks. First set of experiments are conducted with training and test patterns formed as binary format of 150x3. The number of iterations (epochs) required by each network to learn the particular sample was captured and an average of such 10 values is summarized in Table VI. This table data is used to compare the learning and convergence performance of each network. Epoch data captured for this set of experiments are shown in Table VI.

TABLE VI
EPOCHS OR NUMBER OF NETWORK ITERATIONS FOR THE TWO NETWORK SETTINGS RBF-MLP AND FEED FORWARD MLP USING BINARY INPUT SAMPLES OF 150X3 SIZES.

Samples	Epochs with MLP	Epochs With RBF-MLP	Samples	Epochs With MLP	Epochs With RBF-MLP
Sample 1	117	5	Sample 31	50000	11755
Sample 2	248	23	Sample 32	50000	12891
Sample 3	392	28	Sample 33	50000	14378
Sample 4	600	31	Sample 34	50000	17603
Sample 5	856	59	Sample 35	50000	19379
Sample 6	1303	71	Sample 36	50000	21118
Sample 7	2651	108	Sample 37	50000	25375
Sample 8	3899	163	Sample 38	50000	33426
Sample 9	5527	177	Sample 39	50000	36568
Sample 10	7605	202	Sample 40	50000	37624
Sample 11	8124	229	Sample 41	50000	40275
Sample 12	13642	256	Sample 42	50000	41811
Sample 13	14200	291	Sample 43	50000	43899
Sample 14	16693	337	Sample 44	50000	44053
Sample 15	17834	375	Sample 45	50000	45276
Sample 16	18005	490	Sample 46	50000	46831
Sample 17	19766	736	Sample 47	50000	50000
Sample 18	20763	934	Sample 48	50000	48643
Sample 19	21830	1016	Sample 49	50000	49079
Sample 20	26991	1419	Sample 50	50000	50000
Sample 21	28421	1620	Sample 51	50000	50000
Sample 22	32095	1883	Sample 52	50000	50000
Sample 23	34298	2037	Sample 53	50000	50000
Sample 24	35109	2790	Sample 54	50000	50000
Sample 25	39254	3188	Sample 55	50000	50000
Sample 26	48913	3632	Sample 56	50000	50000
Sample 27	50000	4194	Sample 57	50000	50000
Sample 28	50000	6278	Sample 58	50000	50000
Sample 29	50000	8115	Sample 59	50000	50000
Sample 30	50000	9023	Sample 60	50000	50000

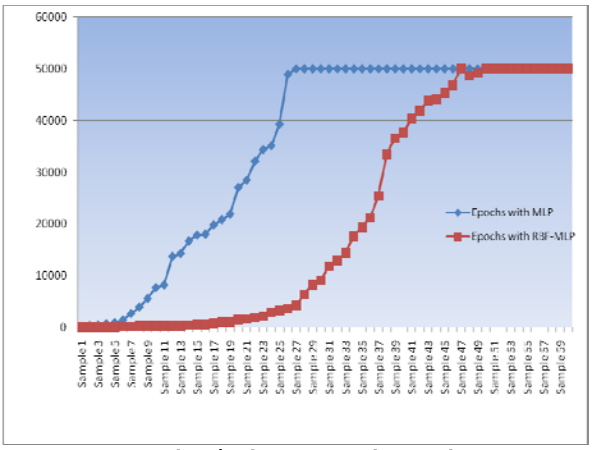


Fig. 9 Comparison chart for the two networks using binary input samples of 150x3 sizes.

The presence of 50000 in above mentioned table shows that the maximum number of epochs has been reached but the network did not converge to the desired output; due to the error exists in the network. The graphical representation for both networks in terms of epochs has been displayed in Fig. 9.

Presences of maximum epoch values in this graph represent the powerlessness of network to learn the behavior in specified limitation of iterations. Only 267 samples were learnt by the MLP and RBF-MLP could make up to 493 samples.

Same set of networks was also examined for its performance against the other type of training and test samples. The samples used for this experiment execution phase was in the form of 32 tangent values for each word. Thus 600 words make 32X600 sample sizes. The number of iterations or epochs used to learn the behavior was captured for each word then simple average has been calculated. The results for the performance of these two networks for this data have been presented in the Table VII.

TABLE VII
Epochs or number of iterations for the two networks settings with tangent values for each input sample of 32x1 sizes

Samples	Epochs with MLP	Epochs With RBF-MLP	Samples	Epochs With MLP	Epochs With RBF-MLP
Sample 1	32	7	Sample 31	29312	5316
Sample 2	39	9	Sample 32	31957	5874
Sample 3	53	15	Sample 33	36568	6689
Sample 4	64	22	Sample 34	49465	6701
Sample 5	88	36	Sample 35	50000	8596
Sample 6	142	41	Sample 36	50000	8952
Sample 7	179	43	Sample 37	50000	10527
Sample 8	204	70	Sample 38	50000	10803
Sample 9	221	84	Sample 39	50000	11059
Sample 10	260	120	Sample 40	50000	12603
Sample 11	355	193	Sample 41	50000	14545
Sample 12	598	247	Sample 42	50000	14379
Sample 13	602	309	Sample 43	50000	16911
Sample 14	785	386	Sample 44	50000	18379
Sample 15	936	443	Sample 45	50000	19206
Sample 16	1251	742	Sample 46	50000	19700
Sample 17	1677	813	Sample 47	50000	24485
Sample 18	2054	934	Sample 48	50000	26592
Sample 19	2348	998	Sample 49	50000	31007
Sample 20	2603	1219	Sample 50	50000	37601
Sample 21	3184	1344	Sample 51	50000	40155
Sample 22	3812	1475	Sample 52	50000	46952
Sample 23	6059	1529	Sample 53	50000	50000
Sample 24	6875	1942	Sample 54	50000	50000
Sample 25	7426	2061	Sample 55	50000	50000
Sample 26	8513	2413	Sample 56	50000	50000
Sample 27	9624	2455	Sample 57	50000	50000
Sample 28	12198	2570	Sample 58	50000	50000
Sample 29	17274	3485	Sample 59	50000	50000
Sample 30	20166	5499	Sample 60	50000	50000

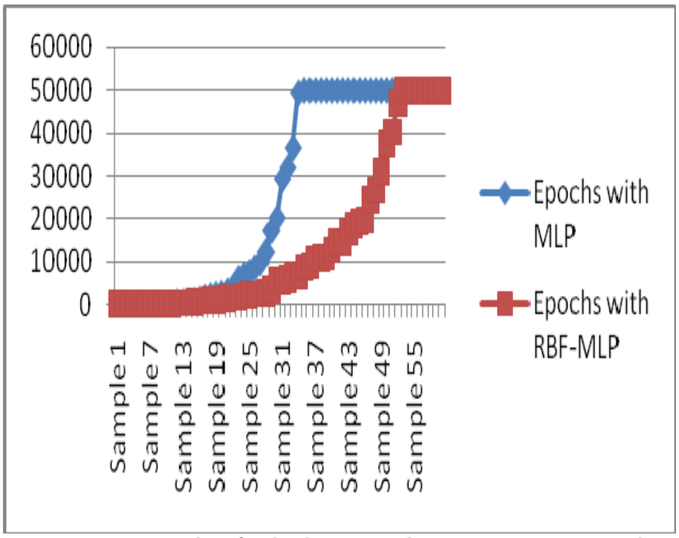


Fig. 10 Comparison chart for the three network settings using tangent values for each input sample of 32x1 sizes

For this set of experiments also, maximum number of epochs assigned to the networks is 50000. We can see in Table 7 that error still exists in the network, because of the presence of 50000 in above mentioned table again. But at the same time, it is clearly evident that in this set of experiments, all networks have performed better as they have converged in fewer epochs. The graphical representation for both networks in terms of epochs has been displayed in Fig. 10.

The networks have achieved better performance in this case of experiments. It is evident from the data captured in table 7 that all networks have learnt more samples in less iteration. As compared to table 6, this time FF-MLP has learnt 348 samples; whereas, RBF-MLP learnt 524 samples.

V. CONCLUSIONS

. We have considered the two problems in this research work. The first problem is for the classification of handwritten English vowels and the second problem is for the recognition of handwritten curve scripts of three letters with the two techniques of feature extractions. The results described in this paper indicate that, for the handwritten English language vowels classification problem, feed forward neural network trained with back propagation algorithm does not perform better in comparison to feed forward neural network trained with decent gradient with RBF. We found that, in each and every case, the DG-RBF network gives better results for the classification of English vowels, in comparison to the back propagation for the MLP network. It has been also observed that the RBF network has also stuck in local minima of error for some of the cases. The reason for this observation is quite obvious, because there is no guarantee that RBFNN remains localized after the supervised learning and the adjustment of the basis function parameters with the supervised learning represents a non-linear optimization, which may lead to the local minimum of the error function. But the considered RBF neural network is well localized and it provides that an input is generating a significant activation in a small region. So that, the opportunity is getting stuck at local minima is small. Thus the number of cases for DG-RBFNN to trap in local minimum is very low. The direct application of DG-RBF to the handwritten character classification has been explored in this research. The aim is to introduce an alternative approach to solve the handwritten character classification problem. The results from the experiments conducted are quite encouraging and reflect the importance of radial basis function for the optimal classification to the given problem.

The experimental results for the second problem also confirm that the DG-RBF network results in high performance in terms of recognition rate and classification accuracy, at the same time completely eliminating the substitution error. The presented result demonstrate that, within the simulation framework presented above, large significant difference exists between the performance of Backpropagation feed-forward neural network and BG-RBF for handwritten English words recognition problem. The results described in this paper indicate that the handwritten English language words classification problem, feed-forward neural network trained with Backpropagation algorithm does not perform better in comparison of feed-forward neural network trained with DG-RBF. The performance of DGRBF is efficient and accurate in all the simulations. The higher speed of convergence in the DG-RBF training process suggests that this architecture may not be fascinated in the false minima of the error surface. It may also minimize the possibilities of misclassification for any unknown testing input pattern.

Nevertheless, more work need to be done especially on the tests for large complex handwritten characters. Some future works should also be explored. For instances, current work is showing the performance of DG-RBF over back propagation for MLP network in classification process up to handwritten English language vowels and the recognition of handwritten curve scripts of three letters but we can proceed further to use this idea for more complex handwritten character recognition problems in which the character are not of the same size and may be not having the same rotation angle. The conjugate decent for the weights between hidden layer and output layer and for the parameters of basis function can also be calculated to increase the performance of the network for convergence as the extension work.

REFERENCES

[1] M. Riedmiller, "*Rprop - Description and Implementation Details Technical Report*", University of Karlsruhe: W-76128 Karlsruhe (1994).

[2] K. Fukushima and N. Wake, "*Handwritten alphanumeric character recognition by the neocognitron*," IEEE Trans. on Neural Networks, 2(3) 355-365 (1991).

[3] Manish Mangal and Manu Pratap Singh, "*Analysis of Classification for the Multidimensional Parity-Bit-Checking Problem with Hybrid Evolutionary Feed-forward Neural Network*." Neurocomputing, Elsevier Science, (70) 1511-1524 (2007).

[4] D. Aha and R. Bankert, "*Cloud classification using error-correcting output codes*", Artificial Intelligence Applications: Natural Resources, Agriculture, and Environmental Science, 11(1) 13-28 (1997).

[5] Y.L. Murphey, Y. Luo, "*Feature extraction for a multiple pattern classification neural network system*", IEEE International Conference on Pattern Recognition, (2002).

[6] L. Bruzzone, D. F. Prieto and S. B. Serpico, "*A neural-statistical approach to multitemporal and multisource remote-sensing image classification*", IEEE Trans. Geosci. Remote Sensing, (37) 1350-1359 (1999).

[7] J.N. Hwang, S.Y. Kung, M. Niranjan, and J.C. Principe, "*The past, present, and future of neural networks for signal processing*", IEEE Signal Processing Magazine, 14(6) 28-48 (1997).

[8] C. Lee and D. A. Landgrebe, "*Decision boundary feature extraction for neural networks*," IEEE Trans. Neural Networks, 8(1) 75–83 (1997).

[9] C. Apte, et al., "*Automated Learning of Decision Rules for Text Categorization*", ACM Transactions for Information Systems, 12 233-251 (1994).

[10] Manish Mangal and Manu Pratap Singh, " *Handwritten English Vowels Recognition using Hybrid evolutionary Feed-Forward Neural Network*", Malaysian Journal of Computer Science, 19(2) 169 –187 (2006).

[11] Y. Even-Zohar and D. Roth, "*A sequential model for multi class classification*", In EMNLP-2001, the SIGDAT. Conference on Empirical Methods in Natural Language Processing, 10-19 (2001).

[12] S. N. Srihari, "*Recognition Of Handwritten and Machine printed text for postal address Interpretation*", Patterns Recognition Letters, 14 (1993).

[13] T. Gergely, K. Krislof and Csaba, "*Analogical preprocessing and segmentation algorithm for offline hand writing recognition*", Journal of Circuits Systems and Computers, 12 (6) 783-804 (2003).

[14] S. Procter, J. Illingworth, F. Mokhtarian. "*Cursive Handwriting Recognition using Hidden Markov Models and a lexicon driven level building algorithm*", IEEE Proc Vision, Image and Signal Processing, 35 332-367 (2000).

[15] P. Schneider. "*Computer assisted spelling normalization of 18th century English*". Language and Computers, 36 (2001).

[16] F. Rossant and I. Bloc, "*Robust and adaptive OMR system including Fuzzy modeling, fusion of musical rules, and possible error detection*", *EURASIP J. Appl. Signal Process.* 1 160-160 (2007).

[17] R. Plamondon and S. N. Srihari. "*On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey*". IEEE Trans. Pattern Anal. Mach. Intell. (2000).

[18] M. Ofer, "*Decision Region Connectivity Analysis: A Method for Analyzing High-Dimensional Classifiers*", Machine Learning, 48 (2002).

[19] V. Alessandro, P. Michael, "*Combining Online and Offline Handwriting Recognition*", *ICDAR,* Seventh International Conference on Document Analysis and Recognition (ICDAR'03) 2 844 (2003).

[20] R. Palacios and A. Gupta, "*A System for Processing Handwritten Bank Checks Automatically*", MIT Sloan, (2002).

[21] N. Gorski, V. Anisimov, E. Augustin, O. Baret, D. Price, J. Simon, "*A2iA Check Reader: A family of bank check recognition systems*", ICDAR -99. Proceedings of the Fifth International Conference on Document Analysis and Recognition, 523 – 526 (1999).

[22] S. Lee, "*Off-Line Recognition of Totally Unconstrained Handwritten Numerals Using Multilayer Cluster Neural Network*", IEEE Trans. Pattern Anal. Mach. Intelligence 18 (6) 648-652 (1996).

[23] J. Fierrez, J. Ortega-Garcia, D. Ramos, and J. Gonzalez-Rodriguez. "*HMM-based on-line signature verification: Feature extraction and*

signature modeling", Pattern Recognition Letter, 28 (16) 2325-2334 (2007).

[24] D. Xi, S. W. Lee, "*Extraction of reference lines and items from form document images with complicated background*", Pattern Recognition 38 (2) 289-305 (2005).

[25] M. H. Shaheed, "*Feedforward neural network based non-linear dynamic modeling of a TRMS using RPROP algorithm*", Aircraft Engineering and Aerospace Technology, 77 (1)13-22 (2005).

[26] A. E. Hassan, J. Wu and R. C. Holt, "*Visualizing Historical Data Using Spectrographs*" In Proceedings of the 11th IEEE international Software Metrics Symposium METRICS, IEEE Computer Society, Washington, (2005).

[27] C. Y. Suen, and R. Legault, C. Nadal, M. Cheriet and L. Lam, "*Building a New Generation of Handwriting Recognition Systems*", Pattern Recognition Letters 14 305-315 (1993).

[28] S. B. Cho, "*Neural-Network Classifiers for Recognizing Totally Unconstrained Handwritten Numerals*", IEEE Trans. on Neural Networks, 8 43-53 (1997).

[29] H. Yamada and Y. Nakano, "*Cursive Handwritten Word Recognition Using Multiple Segmentation Determined by Contour Analysis*", IEICE Transactions on Information and Systems, 79 (E) 464-470 (1996).

[30] F. Kimura, N. Kayahara, Y. Miyake and M. Shridhar, "*Machine and Human Recognition of Segmented Characters from Handwritten Words*", 4th International Conference on Document Analysis and Recognition (ICDAR '97), Ulm, Germany, 866-869 (1997).

[31] P. D. Gader, M. Mohamed and J. H. Chiang, "*Handwritten Word Recognition with Character and Inter-Character Neural Networks*", IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics, 27 158-164 (1997).

[32] Martin M. Anthony, Peter Bartlett, "*Learning in Neural Networks: Theoretical Foundations*", Cambridge University Press, New York, NY, (1999).

[33] M. Wright, "*Designing neural networks commands skill and savvy*", EDN, 36(25), 86-87 (1991).

[34] T. P. Vogl, J. K. Mangis, W. T. Zink, A. K. Rigler, D. L. Alkon "Accelerating the Convergence of the Back Propagation Method", Biol. Cybernetics, 59 257-263 (1988)

[35] R. A. Jacobs, "*Increased Rates of Convergence through Learning Rate Adaptation*", Neural Networks, 1 295-307 (1988).

[36] X.H. Yu, G.A. Chen, and S.X. Cheng, "*Dynamic learning rate optimization of the backpropagation algorithm"*, IEEE Trans. Neural Network, 6 669 - 677 (1995).

[37] Stanislaw Osowski, Piotr Bojarczak, Maciej Stodolski, "*Fast Second Order Learning Algorithm for Feedforward Multilayer Neural Networks and its Applications*", Neural Networks, 9(9) 1583-1596 (1996).

[38] R. Battiti , "*First- and Second-Order Methods for Learning: Between Steepest Descent and Newton's Method*". Computing: archives for informatics and numerical computation, 4(2) 141-166 (1992).

[39] S. Becker, & Y. Le Cun, "*Improving the convergence of the back-propagation learning with second order methods*", In D. S. Touretzky, G. E. Hinton, & T. J. Sejnowski (Eds.), Proceedings of the Connectionist Models Summer School. San Mateo, CA: Morgan Kaufmann, 29-37(1988).

[40] M. F. Moller, "A *Scaled Conjugate Gradient Algorithm for Fast Supervised learning*", Neural Networks, 6 525-533(1993).

[41] M.T. Hagan, and M. Menhaj, "*Training feed-forward networks with the Marquardt algorithm*", *IEEE Transactions on Neural Networks*, 5(6) 989-993(1999).

[42] P. Petra and A. Chid, "*Empirical evaluation of feature subset selection based on a real-world data set*", Engineering Applications of Artificial Intelligence, 17(3) 285-288 (2004).

[43] M. Bierlaire, M. Thémans, "*Dealing with singularities in nonlinear unconstrained optimization*", European Journal of Operational Research, 196(1) 33-42 (2009).

[44] L. Jianhua and B. Laleh, "*An Effective Clustering Algorithm for Mixed-size Placement*", Proc. of ISPD, Austin, (2007).

[45] N. Shental, A. Bar-hillel and D. Weinshall, "*Computing Gaussian Mixture Models with. EM using Side-Information*", in Proc. of Int. Conference on Machine Learning, ICML-03, Washington DC, (2003).

[46] C. M. Bishop. "*Neural Networks for Pattern Recognition*". Oxford University Press, (1995).

[47] P. Christenson, A. Maurer & G. E. Miner, "*handwritten recognition by neural network*", http:// csci.mrs.umn.edu/ UMMCSciWiki/ pub/ CSci4555s04/ InsertTeamNameHere/ handwritten. pdf (2005).

[48] M. J. D. Powell, "*Radial basis functions for multivariable interpolation: A review*", in *Algorithms for Approximation of Functions and Data*, J. C. Mason, M. G. Cox, Eds. Oxford, U.K.: Oxford Univ. Press, 143-167 (1987).

[49] Guobin Ou, Yi Lu Murphey: "*Multi-class pattern classification using neural networks*". Pattern Recognition, 40 (1) 4-18 (2007).

[50] D. P. Bertsekas, "*Nonlinear Programming, Belmont, MA: Athena Scientific*", ISBN 1-886529-14-0 (1995).

[51] D. P. Bertsekas and J. N. Tsitsiklis, "*Neuro-Dynamic Programming*", Belmont, MA: Athena Scientific, ISBN 1-886529-10-8 (1996).

[52] V. S. Dhaka and M. P. Singh, "*Handwritten character recognition using modified gradient descent technique of neural networks and representation of conjugate descent for training patterns*", International Journal of Engineering, Iran, 22 (2) 145-158 (2009).

[53] S. Saarinen, R. Bramley and G. Cybenko, "*Ill-conditioning in neural network training problems*", Siam J. of Scientific Computing, 14 693-714 (1993).

[54] J. Moody and C.J. Darken. "*Fast learning in networks of locally-tuned processing units*", Neural Computation, 1(2) 281-294 (1989).

[55] T. Poggio, and F. Girosi, "*Regularization algorithms for learning that are equivalent to multilayer networks*", *Science*, 247 978-982 (1990b).

[56] M. T. Musavi , W. Ahmed , K. H. Chan , K. B. Faris , *D. M. Hummels, "On the training of radial basis function classifiers*", Neural Networks, 5(4) 595-603 (1992).

[57] D. Wettschereck, T. G. Dietterich, "*Improving the performance of radial basis function networks by learning center locations*". In J. E. Moody, S. J. Hanson, and R. P. Lippmann, (Eds.) *Advances in Neural Information Processing Systems,* San Mateo, CA: Morgan Kaufmann, *4* 1133-1140(1992).

[58] W. P. Vogt, "*Dictionary of Statistics and Methodology: A Nontechnical Guide for the Social Sciences",* Thousand Oaks: Sage (1993).

[59] S. Haykin, "*Neural Networks",* Macmillan College Publishing Company, Inc, New York (1994).

[60] D. S. Broomhead and D. Lowe, "*Multivariable functional interpolation and adaptive networks*", Complex Syst, 2 321-355(1988).

[61] T. Poggio, F. Girosi, "*Networks for approximation and learning*", Proc IEEE, 78(9) 1481–1497(1990).

[62] F. Girosi, T. Poggio, "*Networks and the best approximation property*", Biol Cybern 63 169–176 (1990).

[63] M.A. Kraaijveld and R.P.W. Duin, "*Generalization capabilities of minimal kernel-based networks*", *Proc. Int. Joint Conf. on Neural Networks (Seattle, WA, July 8-12)*, IEEE, Piscataway, U.S.A., I-843 - I-848(1991).

[64] S. Chen, C. F. N. Cowan, and P. M. Grant, "*Orthogonal least squares learning algorithm for radial basis function networks*", *IEEE Transactions on Neural Networks*, ISSN 1045-9227, 2 (2) 302-309(1991).

[65] C. M. Bishop, "*Novelty detection and neural network validation*", IEEE Proceedings: Vision, Image and Signal Processing. Special issue on applications of neural networks, 141 (4) 217–222(1994b).