CONJUGATE DESCENT OF GRADIENT DESCENT RADIAL BASIS FUNCTION FOR GENERALIZATION OF FEED FORWARD NEURAL NETWORK

[1] **Shivpuje Nitin Parappa**
Shri Venkateshwara University
Gajraula , Amroha(Uttar Pradesh)
India
(e-mail : shivpujenitin@gmail.com)

[2] **DR. Manu Pratap Singh**
Institute of Computer & Information Science
Dr. B.R.Ambedkar University, Agra -282002
Uttar Pradesh, India
(e-mail: manu_p_singh@hotmail.com)

Abstract

Multilayer feed-forward neural network trained with conventional gradient descent method of generalized delta learning rule is not conveniently exhibit the generalized behavior for the pattern mapping task or in pattern classification. This inadequacy of the generalized delta learning rule also exhibits in its conjugate descent or second derivative. The objective of this study is to analyze the performance of feed forward neural network for performing the generalized pattern classification task for the automated word recognition. The performance of feed forward neural network for this generalization is evaluated with gradient descent radial basis function feed-forward neural network architecture. Here in this paper we are proposing a novel method to improve the performance of Multi Layer feed-forward neural network for generalized pattern recognition task using second order gradient descent of Radial Basis Function i.e. Radial Basis Function and its Conjugate Descent.

The results of 1800 experiments indicate that the Multi Layer feed-forward neural network performs accurately and exhaustively with imposed Radial Basis Function and its Conjugate Descent. The results indicate that the performance of the Multi Layer feed-forward neural network is much efficient and improved convergence with the Radial basis function network and from its Conjugate Descent.

Keywords : Descent Gradient, Radial Basis Function, Generalized pattern classification, Pattern Recognition, Multi layer feed forward neural networks.

1 INTRODUCTION

A multi-layer feed forward neural network exhibits the good generalization and good approximation capabilities. The generalized behavior of the multilayer feed forward neural network shows the interpolative behavior for the given test pattern. The conventional learning rule like backpropagation or generalized delta learning rule trained the multilayer feed forward neural network to capture generalized implicit relationship between input and output pattern pairs those are used during the training process. There are various applications like pattern mapping, pattern classification, optimization and estimation where the efficient learning by the back-propagation algorithm is used to train the multilayer feed forward neural network for many practical applications. The back-propagation algorithm calculates the weight changes of artificial neural networks, and a common approach is to use a two-term algorithm consisting of a learning rate (LR) and a momentum factor (MF). The major drawbacks of the two-term BP learning algorithm are the problems of local minima and slow convergence speeds, which limit the scope for real-time applications. A local minimum is defined as a point such that all points in a neighborhood have an error value greater than or equal to the error value in that point [1, 2].

It is found that the trained neural networks have been used in a number of applications such as pattern mapping & classification [3, 4, 5], non-linear optimization [6], remote sensing [7], dynamic modeling and medicine [8]. 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 [9]. This property makes neural networks, an attractive tool to many pattern classification problems such as hand written curve scripts [10, 11, 12].

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 [13]. 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 [14]. However, a single hidden layer may take longer to train.

In neural networks, the choice of learning algorithm, network topology, weight and bias initialization and input pattern representation are important factors for the network performance in order to accomplish the learning. In particular, the choice of learning algorithm determines the rate of convergence, computational cost and the optimality of the solution. The multi layer feed forward is one of the most widely used neural network architecture. The learning process for the feed forward network can consider as the minimization of the specified error (E) that depends on all the free parameters of the network. The most commonly adopted error function is the least mean square error. In the feed forward neural network with J processing units in the output layer and for the l^{th} pattern, the LMS is given by;

$$E^l = \frac{1}{2}\sum_{j=1}^{M}(d_j^l - y_j^l)^2 \qquad (1.1)$$

Here $l = 1$ to L (total number of input-output pattern pairs of training set) and d_j^l and y_j^l are the desired and actual outputs corresponding to the l^{th} input pattern.

Hence, due to the non-linear nature of E, the minimization of the error function is typically carried out by iterative techniques [15]. Among the various learning algorithms, the back propagation algorithm [16] is one of the most important and widely used algorithms and has been successfully applied in many fields. It is based on the steepest descent gradient and has the advantage of being less computationally expensive. However, the conventional back propagation learning algorithm suffers from short coming, such as slow convergence rate and fixed learning rate. Furthermore it can be stuck to a local minimum of the error.

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 [17] and Jacobs [18] 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 [19]. Several other variations of back propagation algorithms based on second order methods have been proposed [20-24]. This method generally converges to minima more rapidly than the method based solely on gradient decent method. However, they require an additional storage and the inversion of the second-order derivatives of the error function with respect to the weights. The storage requirement and computational cost, increases with the square of the number of weights. Consequently, if a large number of weights are required, the application of the second order methods may be expensive.

It can realize that the generalization or approximation is basically the optimization problems. The multilayer feed forward neural network with backpropagated algorithm can perform the unconstraint non-linear optimization. It can also realize that the generalize pattern classification task is a non-linear 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, operations research, and statistical computing [25, 26]. Masters [27] has a good elementary discussion of conjugate gradient and Levenberg-Marquardt algorithms in the context of NNs. There is no single best method for nonlinear optimization. One needs to choose a method based on the characteristics of the problem to be solved. Objective functions that are not continuously differentiable are more difficult to optimize. For continuous objective functions that lack derivatives on certain manifolds, such as ramp activation functions (which lack derivatives at the top and bottom of the ramp) and the least-absolute-value error function (which lacks derivatives for cases with zero error), sub-gradient methods can be used. For objective functions with discontinuities, such as threshold activation functions and the misclassification-count error function, Nelder-Mead simplex algorithm [52] and various secant methods can be used. However, these methods may be very slow for large networks, and it is better to use continuously differentiable objective functions when possible.

Artificial Neural Networks (ANNs) have shown the efficient performance for the non-linear unconstraint optimization problems. The pattern classification for the handwritten curve script with its learning and generalization is an important nonlinear optimization activity. Multilayer feed-forward neural network with backpropagation learning rule and its various variants have been applied for this problem but due to ill-posing and non-convergence problem the performance is not found adequate. The Multilayer feed-forward neural network is not intended specifically to solve a pattern classification or pattern mapping task, as both require generalization based on closeness property in classification and smoothness property in mapping respectively. Thus, a Multilayer feed-forward neural network trained with backpropagation learning is neither designed to exploit the property of closeness for generalizing a classification task, nor is it designed to exploit the property of smoothness to generalize a function approximation task. It is designed mainly to provide discrimination between patterns belonging to different classes. Therefore, mere manipulation of the structure of a neural network and learning of Multilayer feed-forward neural network are not likely to achieve the generalization required for a given problem. There is no guarantee of obtaining the desired result. This is because; the network is not designed specifically to address the generalization problem. Moreover, it is not generally possible to analyze a Multilayer feed-forward neural network to understand the task each layer is performing. In fact, if the given problem is known to be a classification problem based on the closeness of data in the input vectors, then specific architectures can be evolved to achieve generalization. Such architectures tend to be much simpler than a general Multilayer feed-forward neural network, and training also is likely to be simpler than the backpropagation learning. Therefore, it is possible to improve the generalization capability using regularization which involves imposing some smoothness & closeness constraints explicitly on the pattern classification and pattern mapping function.

Let we assume that the training set $(a_l, b_l), l = 1,2,.....L.$ data consists of pairs of input-output vectors represented by For a classification task, b_l is an *N*-dimensional vector of a_l zeros and ones, with a 1 in the J^{th} position if the input vector belongs to the J^{th} class. This is considered as the hard classification. There may have several input vectors, which are close to each other, and hence may have the same associated with b_l them. In many situations, it may be desirable to have the *N*-dimensional output vector to represent an estimate of the probability distribution of the classes for the given input. That is, the J^{th} component of b_l corresponds to the probability that the input vector belongs to the class *j*. In this case the sum of all the components in b_l will add up to **1.** The input vector a_l could be as an *M*-dimensional vector of ones and zeros or a vector of real numbers. In the function estimation or pattern mapping the output vector b_l is an *N*-dimensional vector of real values. The function estimation can also be viewed as a nonparametric regression problem, as we are trying to determine a network that realizes the best fit function for the given input-output pairs of data. Thus, in both the cases i.e. generalize pattern classification and pattern mapping the learned network should generalize well, which means that the network should give the correct classification for a new (test) data input in the case of a classification task and a reasonable approximation to the true function value for a new (test) data input in the case of function approximation. The one essential subset of artificial neural network which has proved to be more adequate for regularization and interpolation 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 [28], orthogonal least square method [29], clustering algorithm [30], Gaussian mixture models [31] and with the supervised learning method [32].

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 [32]. This makes the RBF network competitive with the multilayer perceptron in classification accuracy. It has been observed [33] that the use of unsupervised techniques to determine the Radial 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 Radial Basis functions using density estimation of the input data and takes no consideration for the target levels associated with the data. Thus, it is obvious that to set the parameters of the Radial Basis functions for the optimal performance the target data should be included in the training procedure and it reflects the supervised training.

Hence, the Radial Basis function parameters for regression can be found by the training the Radial Basis function centers and widths along with the second layer weights as adaptive parameters to be determined by minimization of an error function. Although, both Radial basis feed forward neural network (RBFN) and multilayer neural feed forward neural network have shown appreciable generalization performance, but at the same time we can see that in these two networks Feed Forward mechanism of neural network and weight update function (Radial Basis) of RBFN are two key role players for the success of these systems. So In this work we have tried to create a system by taking the above specified qualities of both networks together.

The multilayer neural feed forward neural network 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 multilayer neural feed forward neural network [34]. Also, all of the above methods find local optima i.e. they are not guaranteed to find a 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-conditioned [34], especially when there are many hidden units. The algorithms that use only first-order information, such as steepest descent and standard backpropagation 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 [35]. Due to the ill-conditioned nature of multilayer feed forward neural network trained with backpropagation algorithm used for handwriting recognition, it has also been proposed to evaluate the performance of the proposed network with the introduction of Conjugate Descent of RBF. The Conjugate Gradient Descent methods have been proven to be most effective and fast convergence methods in the problem domain of supervised learning.

In this research work we consider three neural networks architectures (NN1, NN2 and NN3). The first architecture NN1 is trained and tested with the conventional back propagation learning algorithm with the incorporation of Doug's Momentum descent term [36]. The NN2 network architecture has been put into operation with the Radial Basis Function [37] in the hidden layer. This network incorporates the Steepest Gradient Descent for weight updates. The third architecture uses NN2 as base network but includes more restrictions in terms of weight update rule. NN3 introduces the Conjugate Descent of Radial Basis Function. The performance of all these three networks has been judged for input patterns of the handwritten curve scripts of three words of English language. These input patterns are considered in the form of binary image matrix. 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. So these two things have been the key-focus points of this work. The experiments are conducted with 600 samples of English words of 3 characters. The significant improvement in the generalized pattern classification of handwritten words is achieved. Hence, among the neural network models, Radial Basis function network seems to be quite effective for generalize pattern classification task such as handwritten character recognition.

2 Gradient descent Radial basis Function

The architecture and training methods of the RBF network are well known [37, 38, 39, 40, 41, 42, 43] & 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 simpler 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 [44, 45]. The RBFN is having, network architecture similar to the classical regularization network, 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 viewpoint of approximation theory, the regularization network has three following desirable properties [46, 47]:

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 the sense that it minimizes a functional 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.

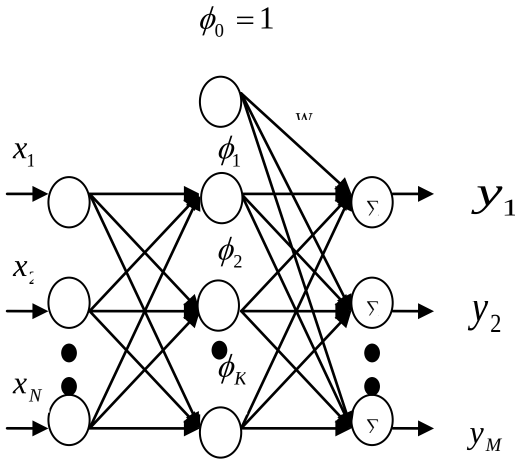


Figure 1: Architecture of the RBFN.

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 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 mean square error (MSE) range by using the linear optimization method. Thus, for an input pattern x, the output of the j^th node of the output layer can define as;

$$y_j(x) = \sum_{k=1}^{K} w_{kj}\phi_k(\|x_i - \mu_k\|) + w_{0j} \qquad (2.1)$$

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

And 1 for $k = 0$ (bias neuron) (2.2)

Where x is 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

μ_k

vector between the input layer and the *k*th hidden layer neuron can consider as the centre for the feed forward RBF neural network.

Hence, for a set of L pattern $\{(x_l, y_l)\}$ pairs, (2.1) can be expressed in the matrix form as

$$Y = w^T \phi \qquad (2.3)$$

where $W = [w_1, ...w_m]$ a $K\,x\,M$ weight matrix

$w_j = (w_{0j}, ...w_{kj})^T$, , $\phi = [\phi_0, ...\phi_k]$

$\phi_{l,k} = [\phi_{l,1}, ...\phi_{l,k}]^T$

is a $K\,x\,L$ is matrix, the output of the hidden layer for $\phi_{l,k} = \phi(\|x_l - c_k\|)$ $Y = [y_1, y_2, ...y_m]$ den the l^{th}, sample,

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 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 [48], orthogonal least square method [49], clustering algorithm, Gaussian mixture models [50] and with the supervised learning method.

It has been observed [51] 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 (1.1) 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 (x^l, y^l) pairs as; where $l = 1$ to L.

$$\Delta w_{jk} = -\eta_1 \frac{\partial E^l}{\partial w_{jk}} \qquad (2.4)$$

$$\Delta \mu_k = -\eta_2 \frac{\partial E^l}{\partial \mu_k} \qquad (2.5)$$

$$\Delta \sigma_k = -\eta_3 \frac{\partial E^l}{\partial \sigma_k} \qquad (2.6)$$

And

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

Here,

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

And, $\phi_k (\|x^l - \mu_k^l\|) = \exp(-\frac{\|x^l - \mu_k^l\|^2}{2\sigma_k^2})$ (2.7)

Hence, from the equation (2.4) we have,

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

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

or

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

$$\Delta w_{jk} = \eta_1 \sum_{j=1}^{M} \sum_{k=1}^{K} (d_j^l - y_j^l) \dot{s}_j^l (y_j^l) \exp(-\frac{(\|x^l - \mu_k^l\|)^2}{2\sigma_k^2}) \quad (2.8)$$

So,that *Now, fromthe equation (2.6) we have*

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

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

or $\Delta\mu_{ki} =$

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

$$(\frac{x_i^l - \mu_{ki}^l}{\sigma_k^2}) \qquad (2.9)$$

Again from the equation (2.6) we have

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

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

Or,

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

$$(-\frac{(\left\| x_i^l - \mu_{ki}^l \right\|)^2}{2\sigma_k^2}) \frac{\left\| x_i^l - \mu_{ki}^l \right\|^2}{\sigma_k^3}$$

(2.10)

So that, we have from equations (2.8), (2.9) & (2.10) 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. 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 mth step of iteration can express as;

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

(2.11)

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

(2.12)

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

(2.13) $\eta_1, \eta_2 \ \& \ \eta_3$

Here are the coefficients of learning rate.

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

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

(2.14)

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

(2.15)

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

And

(2.16)

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) of the training set.

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 [44]. 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 like conjugate descent of Radial basis function may evolve to have very broad responses.

3. Conjugate Descent of Radial Basis function

It has been observed and discussed , that the multilayer feedforward neural networks suffers with the problem of finding global minima when it is ill-conditioned and the Radial Basis Function network also has no guarantee from the skiing of local minimum. Hence here we are proposing the use of Conjugate Descent of Radial Basis Function. When we apply characteristics of a minimum to the weight update and change in weight from equation (2.14), we find following trajectory graph:

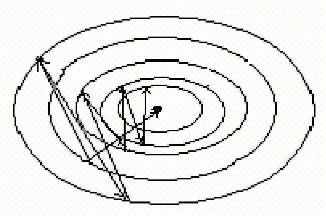


Figure 2: Direction of Error minimization in FF-MLP (Steepest Gradient)

It is clearly evident that the graph is having zigzag motion. The solution to the zigzag problem the steepest descent method suffers from, is to use as much of the direction of the previous step as possible. We can express this as the new search direction being a mixture of the new gradient direction and the previous search direction as:

$$\overline{d}_{new} = -grad(E(w_{new})) + \beta * \overline{d}_{previous} \tag{3.1}$$

Or, in weight update terms we can represent this theorem as:-

$$W_{ik}(t+1) = W_{ik}(t) + \eta \Delta W_{ik}(t) + \alpha \Delta W_{ik}(t-1) + \beta \Delta W_{ik}(t-2) \tag{3.2}$$

Here β is known as the conjugate gradient parameter.

The conjugate gradient descent method chooses the new direction so as not to undo the minimization achieved by the previous step; i.e., the new weight change should not change (to first order) the component of the gradient along the previous direction.

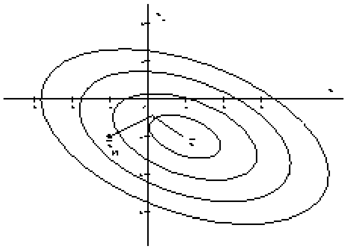


Figure 3: Direction of Error minimization in Conjugate RBF-MLP.

The result of this Conjugate Descent is still a zigzag path, but in another space (scaled and rotated according to the Eigen values). However, in this path, each orthogonal component of the remaining error is reduced to zero in turn, one component at a time, by performing a line search for the minimum along the corresponding vector. For an *N*-dimensional problem, the method converges in *N* steps.

There are various methods those suggests to compute the term β. We have used the Polak-Ribière formula as:

$$\beta = \frac{(grad(E(\overline{w}_{new})) - grad(E(\overline{w}_{previous}))) * grad(E(\overline{w}_{new}))}{(grad(E(\overline{w}_{previous})))^2} \tag{3.4}$$

Now, let us compute the gradient values for set of training and test cycles. The Radial Basis function parameters corresponding to each presented pattern vector at the parameters of the network at the kth step of iteration can be evaluated based on the gradient of the error surface and minima of the local error. The minima can only be verified by computing the second order derivations of error function with respect to standard deviation and width of center vectors.

$$\frac{\delta^2\vec{E}}{\delta{\mu_i}^2} = \sum_{k,p} w_{ki} * (o_k^p - y_k^p) * e^{-\frac{|\vec{x}^p - \vec{\mu}_i|^2}{2\sigma_i^2}} * \left\{\frac{|\vec{x}^p - \vec{\mu}_i|}{\sigma_i^2} * \frac{\vec{x}^p - \vec{\mu}_i}{\sigma_i^2} - \frac{1}{\sigma_i^2}\right\} \tag{3.5}$$

And from equation (2.10) we get-

$$\frac{\delta^2\vec{E}}{\delta{\sigma_i}^2} = \sum_{k,p} w_{ki} * (o_k^p - y_k^p) * \frac{|\vec{x}^p - \vec{\mu}_i|}{2} * e^{-\frac{|\vec{x}^p - \vec{\mu}_i|^2}{2\sigma_i^2}} * \left\{\frac{|\vec{x}^p - \vec{\mu}_i|^2}{\sigma_i^2} - \frac{3}{\sigma_i^4}\right\} \tag{3.6}$$

$$\frac{\delta^2\vec{E}}{\delta\sigma_i\delta\mu_i} = \sum_{k,p} w_{ki} * (o_k^p - y_k^p) * \frac{\vec{x}^p - \vec{\mu}_i}{\sigma_i^3} * e^{-\frac{|\vec{x}^p - \vec{\mu}_i|^2}{2\sigma_i^2}} * \left\{\frac{|\vec{x}^p - \vec{\mu}_i|^2}{\sigma_i^2} - 2\right\} \tag{3.7}$$

The interdependency of mean μ, weight *w* and standard deviation σ is an important thing to study the Error minimization process, so that we can have following equations:-

$$\frac{\delta^2\vec{E}}{\delta\sigma_i\delta w_i} = \sum_{k,p}(o_k^p - y_k^p) * e^{-\frac{|\vec{x}^p - \vec{\mu}_i|^2}{2\sigma_i^2}} * \frac{|\vec{x}^p - \vec{\mu}_i|^2}{\sigma_i^3} \tag{3.8}$$

$$\frac{\delta^2 \vec{E}}{\delta \mu_i \delta w_i} = \sum_{k,p}(o_k^p - y_k^p) * e^{-\frac{|\vec{x}^p - \vec{\mu}_i|^2}{2\sigma_i^2}} * \frac{\vec{x}^p - \vec{\mu}_i}{\sigma_i^2} \tag{3.9}$$

$$\frac{\delta^2 \vec{E}}{\delta {w_i}^2} = \sum_p(o_k^p - y_k^p) * e^{-\frac{|\vec{x}^p - \vec{\mu}_i|^2}{2\sigma_i^2}} + \sum_p e^{-\frac{|\vec{x}^p - \vec{\mu}_i|^2}{\sigma_i^2}} \tag{3.10}$$

The discussed Conjugate gradient descent approach for implementation of NN3 system is incremental learning algorithm in which the parameters are updated at each example (xl, yl) of training set. The training with the Conjugate decent gradient method is capable to provide the equivalent or better performance compared to that of the multi layer feed forward neural network trained with the back propagation. The Conjugate gradient decent method is speedy in convergence since it can competently use the locally tuned representation of the hidden layer units. When the hidden unit receptive fields, controlled by the width σ_k are narrowed for a given input; only a few of the total number of hidden units will be activated and only these units need to be updated. It has been realized that some of the main advantages of the Conjugate Descent Radial basis function, is the fast two stage training and interpretability of the hidden unit representation.

The conjugate gradient descent method has been applied to train the neural network architecture and the significant increase in speed and convergence is considered. Interesting is the possibility that a network trained with this method may converge to quite a different point in the error landscape because of the different size and direction of the steps taken. Especially if the surface is very irregular or contains many local minima the conjugate gradient descent method may perform very differently from, say, the back-propagation method.

Now in the following subsection, we are presenting the simulation designed implementation details of redial basis function and its conjugate descent to accomplish the task of generalized pattern classification for handwritten curve scripts of three words of English language. The performance of RBF and its conjugate descent network is compared with the performance of multilayer feed forward neural network trained with backpropagation learning rule.

4. Simulation Design and Implementation Details

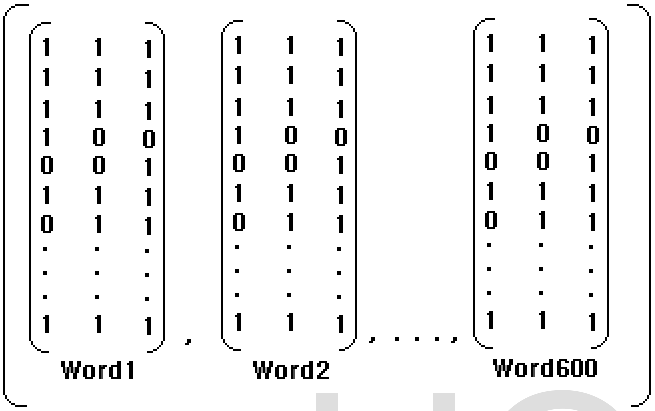
The experiments described in this section are designed to evaluate the performance of feed forward neural networks when evolved with the back propagation algorithm, RBF network and the conjugate decent of RBF network.

As we have discussed already that to accomplish the generalized pattern classification task for the handwritten curve scripts the unconstraint non-linear optimization is required. Hence to perform this three neural network architectures are considered i.e. NN1, NN2 and NN3. The NN1 evolved with multilayer feed forward neural network trained with backpropagation learning rule (FFBP), The NN2 is considered as radial basis function network (RBFN) and the NN3 is evolved with the conjugate descent of radial Basis function network (CDRBFN). The training set for these experiments are constructed with 600 samples of handwritten words of three letters. Each pattern vector of the training set is constructed with input vector of 150 x 1. Thus, each input pattern involves 150 features for each scanned image of the handwritten three letters word. All three neural networks are simulated using feed-forward neural network architecture consists of 150 neurons in input layer, 20 neurons in hidden layer and 26 output neurons. 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. 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 [53]. The distinguishing factors among FFBP (NN1) and RBFN (NN2) and CDRBFN (NN3) is that in the case of FFBP the network contains Log-Sigmoid transfer functions whereas RBFN contains RBF transfer function and Conjugate CDRBFN is optimized with the introduction of Conjugate Descent of RBF. The 625 word samples are gathered form 51 subjects of different ages including male and female for the input samples. After the preprocessing module 600 input samples were considered for training. Each sample was presented to the network 3 times. Thus 1800 experiments have been conducted.

The input patterns for experiments are prepared based on method discussed in [53]. 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:-



$$\left(\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\\1&0&0\\0&0&1\\1&1&1\\0&1&1\\\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot\\1&1&1\end{bmatrix}_{\text{Word1}},\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\\1&0&0\\0&0&1\\1&1&1\\0&1&1\\\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot\\1&1&1\end{bmatrix}_{\text{Word2}},\ldots,\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\\1&0&0\\0&0&1\\1&1&1\\0&1&1\\\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot\\1&1&1\end{bmatrix}_{\text{Word600}}\right)$$

Figure 4: Input sample for the first set of experiments.

Experiments

Three sets of experiments were executed. In each of the sets same type of network architecture was used. In the first experiment we have used multilayer feed forward neural network trained with backpropagation learning algorithm and evaluated for the generalized pattern classification for the given training set. In the second experiment we have used Radial Basis Function network for the same training set to evaluate its performance for the generalized pattern classification. In the third experiment we have used the conjugate descent of Radial basis function network for the same training set and also evaluate its performance again for the generalized pattern classification. In total the number of experiments conducted for training of all of the networks was 1800 including 1800 training cycles for each type of network. The testing of the performance of networks was done with 25 sample patterns of the test pattern set those were not used in the training set. The parameters used for both experiments are described in Table 1 and 2 as follows:

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
Initial weights and biased term values	RandomlyGenerated Values Between 0 and 1

Table 1: 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
Spread parameter σ	1.0
Mean of inputs(c)	Between maximum & minimum values
Initial weights and biased term values	Randomly Generated Values Between 0 & 1

Table 2: Parameters Used for Radial Basis function and its conjugate descent

THE NETWORKS' LEARNING AND PERFORMANCE EVALUATION:-

The system is trained with 600 samples, with following set of training parameters as presented in table 3 and 4 are used in the MATLAB simulation for the neural networks:-

Parameter Name	Value
Max Epochs	50000
Training Function	TRAINLM
Performance Function	MSE (Mean Square Error)
Training Goal	0.0001
Minimum gradient	0.00000007
Maximum Fail	60
Show Graph	100 epochs

Table 3: Network training function and Parameter values for FFBP network.

Parameter Name	Value
Max Epochs	50000
Training Function	NEWRB and NEWRBE
Performance Function	MSE (Mean Square Error)
Training Goal	0.0001
Minimum gradient	0.00000007
Maximum Fail	60
Show Graph	100 epochs

Table 4: Network training function and Parameter values for RBF and CDRBF network.

5. RESULTS AND DISCUSSION

All three networks i.e. NN1 (FFBP), NN2 (RBF) and NN3 (CDRBF) have been trained and examined with same sets of sample data. Training and testing samples of format 150x3 for a particular word, when presented to the Networks; yielded 3 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, 25 test samples were used to validate the performance of these three trained neural networks for the classification.

The networks were trained with 600 different sets of input patterns. The table 5 contains epoch average of 10 iterations for 600 samples, thus only 60 readings have been mentioned. Sample1 depicts average epoch value for sample1 to sample10; Sample2 depicts average epoch value for sample11 to sample20 and so on. Each sample has been presented to three networks. 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 5. This table data is used to compare the learning and convergence performance of each network.

Samples	Epochs with FFBP	Epochs With RBF	Epochs With CDRBF
Sample 1	117	5	3
Sample 2	248	23	15
Sample 3	392	28	19
Sample 4	600	31	22
Sample 5	856	59	27
Sample 6	1303	71	43
Sample 7	2651	108	52
Sample 8	3899	163	69
Sample 9	5527	177	74
Sample 10	7605	202	81
Sample 11	8124	229	95
Sample 12	13642	256	118
Sample 13	14200	291	134
Sample 14	16693	337	150
Sample 15	17834	375	179
Sample 16	18005	490	186
Sample 17	19766	736	208
Sample 18	20763	934	219
Sample 19	21830	1016	241
Sample 20	26991	1419	250
Sample 21	28421	1620	272
Sample 22	32095	1883	285
Sample 23	34298	2037	297
Sample 24	35109	2790	334
Sample 25	39254	3188	406
Sample 26	48913	3632	478
Sample 27	NC	4194	616
Sample 28	NC	6278	699

Samples	Epochs with FFBP	Epochs With RBF	Epochs With CDRBF
Sample 29	NC	8115	854
Sample 30	NC	9023	938
Sample 31	NC	11755	991
Sample 32	NC	12891	1402
Sample 33	NC	14378	1755
Sample 34	NC	17603	1863
Sample 35	NC	19379	2027
Sample 36	NC	21118	2123
Sample 37	NC	25375	4398
Sample 38	NC	33426	6875
Sample 39	NC	36568	7835
Sample 40	NC	37624	8247
Sample 41	NC	40275	9107
Sample 42	NC	41811	10269
Sample 43	NC	43899	15481
Sample 44	NC	44053	18757
Sample 45	NC	45276	20942
Sample 46	NC	46831	22413
Sample 47	NC	50000	25910
Sample 48	NC	48643	29037
Sample 49	NC	49079	30441
Sample 50	NC	NC	35312
Sample 51	NC	NC	41271
Sample 52	NC	NC	46896
Sample 53	NC	NC	47316
Sample 54	NC	NC	48745
Sample 55	NC	NC	49013
Sample 56	NC	NC	NC
Sample 57	NC	NC	NC
Sample 58	NC	NC	NC
Sample 59	NC	NC	NC
Sample 60	NC	NC	NC

Table 5: Epochs or Number of network iterations for the given training set

From the table 5, we can see that the maximum number of epochs assigned to the networks is 50000. The presence of NC (Not Converged) 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 all the three networks in terms of epochs has been displayed in following figure.

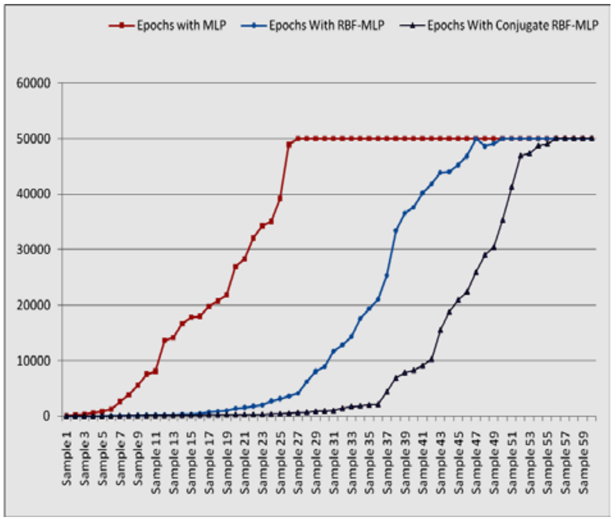


figure 5: Comparison chart of three networks with respect to number of epochs

Presences of maximum epoch values in this figure represent the powerlessness of network to learn the behavior in specified limitation of iterations. Only 267 samples were learnt by the FFBP; whereas, RBF could make up to 493 samples. On the contrary Conjugate RBF (CDRBF) has learnt the maximum i.e. 552 samples. After that the test pattern samples are presented to these networks to validate the performance of these trained neural networks. There are 25 test pattern samples. Hence out of these 25 test pattern samples the FFBP is able to classified correctly only 8 samples and remaining 17 patterns are misclassified i.e. only 32% patterns are correctly classified and remaining 68 % are misclassified. The RBF network is able to classified 15 pattern correctly and 10 test patterns are misclassified i.e. 60% patterns are correctly classified and 40% patterns are misclassified. The CDRBF network is able to classified 21 patterns correctly and 4 patterns are misclassified i.e. 84% patterns are correctly classified whereas 16% patterns are misclassified. Thus the rate of correct classification is much higher and rate of misclassification is low in CDRBF network with respect to the other neural networks. This shows the adequate generalized behavior performance of conjugate descent radial Basis Function network for pattern classification.

The experimental results confirm that the proposed method results in high performance in terms of recognition rate and classification accuracy, at the same time completely eliminating the substitution error. Hence the developed architecture is robust in the recognition of unconstrained handwritten words. We have developed three classification networks to recognize unconstrained handwritten words. The experiments those have been carried out on the training and test sets to assess the performance of each classification network; have shown that it is preferable to deal with Conjugate Descent RBF network. This is due to the amount of data learnt easily by this network as we can refer table 6, which is almost twice the amount of data learnt by feed-forward multilayer neural network trained with backpropagation algorithm.

The results demonstrates that, within the simulation framework presented above, large significant difference exists between the performances of Backpropagation feed-forward neural network and Conjugate Descent based RBF network for handwritten English words recognition problem. The results described in this paper indicate that, for the handwritten English language words classification problem, feed-forward neural network trained with Backpropagation algorithm does not perform better in comparison to feed-forward neural network trained with Conjugate Descent of RBF. The performance of Conjugate RBF-MLP is efficient and accurate in all the simulations. The higher speed of convergence in the Conjugate Descent 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.

The simulation program, which we have been developed in MATLAB 6.5, for testing these three networks for handwritten curve script of English language to accomplish the task of generalized pattern classification, generates initial weights randomly through its random generator. So the epochs for the algorithms will be different every time with the same network structure and the same training data set.

2.4 Conclusions & Future Work

The results described in this present work indicate that, for the non-linear unconstrained optimization problem like generalized classification of handwritten curve script of three words, 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 and conjugate descent of Radial basis function network. The performance of conjugate descent Radial Basis Function network is found better even than simple Radial basis Function network in term of number of training iterations and classification accuracy. We found that, in each and every case, the conjugate descent Radial Basis Function network gives better results for the classification of handwritten curve script of three words, in comparison to the decent gradient with RBF and feed forward neural network trained with back propagation algorithm. It has been also observed that the RBF network has also stuck in local minima of error for some of the cases. The conjugate descent RBF network also stuck in local minima but the probability is much lesser then simple RBF network. 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 proposed conjugate descent radial Basis function (CDRBF) 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 conjugate descent radial Basis function (CDRBF) neural network to trap in local minimum is very low.

The direct application of conjugate descent radial Basis function (CDRBF) neural network to the handwritten curve script classification has been explored in this research. The aim is to introduce as alternative approach to solve the generalized pattern classification problem. The results from the experiments conducted are quite encouraging and reflects the importance of radial basis function for the unconstraint non-linear optimization problem. Nevertheless, more work need to be done especially to evaluate the performance of proposed method on the complex handwritten curve scripts of more numbers of alphabets. Some future works should also be explored for better technique of feature extraction. The evolutionary search techniques can incorporate with conjugate decent RBF to make the performance of neural network for generalization and approximation more adequate to explore the global optimal solutions and minimize the problem of local minimum.

REFERENCES

1. I. G. Sprinkhuizen, Kuyper, E. J. W. Boers, “The local minima of the error surface of the 2-2-1 XOR network”, Annals of Mathematics and Artificial Intelligence, 25 (1-2) (1999) 107 – 136.

2. Y. H. Zweiri , L. D. Seneviratne, K. Althoefer, “Stability analysis of a three-term backpropagation algorithm”, Neural Networks, 18 (10) (2005) 1341 – 1347.

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

4. M. Mangal and M. P. Singh, “Analysis of Classification for the Multidimensional Parity-Bit-Checking Problem with Hybrid Evolutionary Feed-forward Neural Network.” Neurocomputing, Elsevier Science, 70 (2007) 1511-1524.

5. D. Aha and R. Bankert, “Cloud classification using error-correcting output codes”, Artificial Intelligence Applications: Natural Resources, Agriculture, and Environmental Science, 11(1) (1997) 13-28.

6. Y.L. Murphey, Y. Luo, “Feature extraction for a multiple pattern classification neural network system”, IEEE International Conference on Pattern Recognition, (2002).

7. 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 (1999) 1350-1359.

8. 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) (1997) 28-48.

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

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

11. M. Mangal and M. P. Singh, "Handwritten English Vowels Recognition using Hybrid evolutionary Feed-Forward Neural Network", Malaysian Journal of Computer Science, 19(2) (2006) 169 –187.

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

13. M. M. Anthony, P. Bartlett, "Learning in Neural Networks: Theoretical Foundations", Cambridge University Press, New York, NY, (1999).

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

15. J. Go, G. Han, H. Kim and C. Lee, "Multi-gradient: A New Neural Network Learning Algorithm for Pattern Classification", IEEE Trans. Geo-science and Remote Sensing, 39(5) (2001) 986-993.

16. S. Haykin, "Neural Networks", A Comprehensive Foundation, Second Edition, Prentice-Hall, Inc., New Jersey, (1999).

17. 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 (1988) 257-263.

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

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

20. S. Osowski, P. Bojarczak, M. Stodolski, "Fast Second Order Learning Algorithm for Feedforward Multilayer Neural Networks and its Applications", Neural Networks, 9(9) (1996) 1583-1596.

21. 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) (1992) 141-166.

22. 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, (1988) 29-37.

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

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

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

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

27. T. Masters, "Advanced Algorithms for Neural Networks: A C++ Sourcebook", NY: John Wiley and Sons, (1996) ISBN 0-471-10588-0

28. 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) (2004) 285-288.

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

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

31. 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).

32. C. M. Bishop, "Neural Networks for Pattern Recognition". Oxford University Press, (1995).

33. L. Bruzzone and P. Fernández, "Automatic analysis of the difference image for unsupervised change detection," IEEE Trans. Geo-science. Remote Sens., 38 (3) (2000) 1171-1182.

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

35. R. Fletcher, "Practical methods of optimization (2nd ed.)", New York: John Wiley & Sons, (1998) 113, ISBN 978-0-471-91547-8.

36. S. Shrivastava and M. P. Singh, "performance evaluation of feed-forward neural network with soft computing techniques for hand written English alphabets", Journal of Applied Soft Computing, Elsevier, 11 (2011), 156-1182.

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

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

39. T. Poggio, and F. Girosi, "Regularization algorithms

for learning that are equivalent to multilayer networks", Science, 247 (1990 b) 978-982.
40. 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) (1992) 595-603.
41. 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 (1992) 1133-1140.
42. W. P. Vogt, "Dictionary of Statistics and Methodology: A Nontechnical Guide for the Social Sciences", Thousand Oaks: Sage (1993).
43. S. Haykin, "Neural Networks", Macmillan College Publishing Company, Inc, New York, (1994).
44. 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, (1987) 143-167.
45. D. S. Broomhead and D. Lowe, "Multivariable functional interpolation and adaptive networks", Complex System, 2 (1988) 321-355.
46. T. Poggio, F. Girosi, "Networks for approximation and learning", Proc IEEE 78 (9) (1990) 1481–1497.
47. F. Girosi, T. Poggio, "Networks and the best approximation property", Biol Cybern 63 (1990) 169–176.
48. 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., (1991) I-843 - I-848.
49. 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) (1991) 302-309.
50. 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) (1994b) 217–222.
51. M. Bierlaire, M. Thémans, "Dealing with singularities in nonlinear unconstrained optimization", European Journal of Operational Research, 196 (1) (2009) 33-42.
52. K. I. M. McKinnon, "Convergence of the Nelder-Mead simplex method to a non-stationary point", SIAM Journal of Optimization, 9 (1999) 148-158.
53. 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, IJE Transaction A: Basics, 22 (2) (2009) 145-158.