Analysis of Soft Computing Technique to Minimiz the Local Minima Problem in Pattern Recognition for Hand Written English Alphabets

Saurabh Shrivastava
Department of Mathematical Sciences & Computer Applications, Bundelkhand University, Jhansi
hanu.saurabh@gmail.com

Manu Pratap Singh
Department of Computer Science, Dr. B.R. Ambedkar University, Khandari, Agra
manu_p_sing@hotmail.com

Abstract: The back-propagation algorithm suffers with the local minima error surface for a large set of problems. 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. These regions of local minima occur for combinations of the weights from the inputs to the hidden nodes such that one or both hidden nodes are saturated for at least two patterns. However, boundary points of these regions of local minima are saddle points. This paper describes the soft computing techniques for the performance evaluation of Back-propagation algorithm to recognize the hand written English alphabets. Two different architectures of neural network have been taken with five trials of each network. It has been analyzed that the conventional back-propagation algorithm suffers with the problem of non-convergence of the weights. The results of the experiments and result shows that the conventional back-propagation algorithm does not suites to solve the challenging problem most reliably and efficiently.

Keywords: Character recognition, Back-propagation algorithm, local minima error

1. Introduction

Efficient learning by the back-propagation algorithm is required 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 [8,9]. However, GA is particularly good to perform efficient searching in large and complex space to find out the global optimal solution and for the convergence. As the complexity of the search space increases, GA presents an increasingly attractive alternative to gradient based techniques such as error back-propagation algorithm [2].

Pattern recognition aims to classify data (patterns) based on either a priori knowledge or on statistical information extracted from the patterns. The patterns to be classified are usually groups of measurements or observations, defining points in an appropriate multidimensional space. Character recognition plays an important role in today's life. It can solve many complex problems in of real life. An example of character recognition is Handwritten English alphabets. The classic difficulty of being able to correctly recognize even typed optical language symbols is the complex irregularity among pictorial representations of the same character due to variations in fonts, styles and size. This irregularity undoubtedly widens when one deals with handwritten characters [1]. The aforesaid tasks can easily be accomplished by a human being without involving much efforts. The task of pattern classification & pattern mapping the supervised multilayer feed forward neural network [7] is considered with non-linear differentiable function in all processing units of output and hidden layers. The number of processing units in the input layer, corresponds to the dimensionalities of the input pattern, are linear. The number of output units corresponds to the number of district classes in the pattern classification. A method [4] has been developed so that network can be

trained to capture the mapping explicitly in the set of input-output pattern pair collected during an experiment and simultaneously expected to modal the unknown system for function from which the predictions can be made for the new or untrained set of data. The possible output pattern class would be approximately an interpolated version of the output pattern class corresponding to the input learning pattern close to the given test input pattern. This method involves the back propagation-learning rule [5] based on the principle of gradient descent along the error surface in the negative direction. This algorithm is used for the training of a supervised multi-layer feed forward neural network, so that the network could be trained to capture the missing implicit pattern and generate the classification for different features in the given set of input-output pattern pairs.

The proposed experiments demonstrate that for the given set of problems (recognition of English hand written alphabets) the back-propagation algorithm does not perform better than the other soft computing technique such as genetic algorithm in terms of the accuracy and rate of convergence. We found the significant difference in accuracy and the rate of convergence of genetic algorithm with the simple back propagation algorithm.

Section two of this paper describes the generalized approaches and principles of the algorithms applied for the problem. Section three describes the architecture and design of the network used, in terms of simulation design. Section four shows the results of the experiments. Section five describes the discussions based on the results, and finally the references.

2. Supervised feed forward Neural Networks

Feed forward neural network is a biologically inspired classification algorithm. It consists of a number of simple neuron-like processing *units*, organized in *layers*. Every unit in a layer is connected with all the units in the previous layer. These connections are not all equal; each connection may have a different strength or weight. The weights on these connections encode the knowledge of a network. The neural network consists of an input layer of nodes, one or more hidden layers, and an output layer. Each node in the layer has one corresponding node in the next layer, thus creating the stacking effect. The input layer's nodes have output functions that deliver data to the first hidden layer nodes. The hidden layer(s) are the processing layers, 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 a sigmoid function (a logistic curve), which changes the sum to a limited and manageable range. The output sum from the hidden layers is passed on to the output, which produces the final network results as shown in Fig. 1. Feed-forward networks may contain any number of hidden layers, network with a single hidden layer can learn any set of training data that a network with multiple layers can learn, depends upon the complexity of the problem [3]. However, neural network with a single hidden layer may take longer to train.

Popular working-out algorithms for feed forward neural networks such as back-propagation algorithm undergo the intrinsic complications of gradient–decent techniques-predominately local minima in the error function. GA propose an another solution to conventional techniques since they do not rely on gradient information they can sample the search space irrespective of where the existing solution is to be found, while remaining is biased toward good solutions.

A genetic algorithm has four main elements: (i) the genetic code, a concise representation for an individual solution; (ii) the population, a number of individual solutions; (iii) the fitness function, an evaluation of the usefulness of an individual; (iv) the propagation techniques, a set of methods for generating new individuals. In the genetic algorithm, first a population of individuals is generated randomly selecting different samples (or genes) th... h mutation and elitism. From these samples, the crossover operator is used to generate the combinat... of selected samples. The fitness of each individual is ... evaluated. The best among all these is selecte... or further processing. The cycle of evaluation an... ropagation continues until a satisfactory solution... he optimal solution, is found.

This paper focuses on the recognition of handwritten English alphabets in its basic form, i.e. individual character recognition. The rationale is to improve the efficiency of neural network for character recognition task. A series of tests were conducted to determine which of the two learning algorithms i.e. Back-propagation or Genetic Algorithm (GA) [6], trained a neural network faster and more efficiently. The determination of convergent weights for pattern recognition is also an important observation of this research. The simulated results are determined from the five sets of handwritten characters of English alphabets.

3. Simulation Design and Imp... Details

This section describes the ex... to evaluate the performance of f... network when evolved with two... algorithms. We have consider... architectures, first consisting 4 n... layer, 5 neurons in each hidden lay... and 5 neurons in the output laye... architecture consists 4 neurons i... neurons in each hidden layers, 2 ... neurons in the output layer.

3.1 Experiment

Five different sets of alphab... the experiments. The alphabets ar... density function by using MATLA... data. We used two different learn... propagation and Genetic Algorithm... we have taken five trials to train... For this, the scanned images of f... of handwritten English alphabets... collecting these samples, each En... was partitioned into four equal p... values of the pixels for each part ... the density values of the central ... partitioned images of the Eng... calculated. Consequently, four ... from an image of handwritte... alphabets, which were then used... feed-forward neural network. Th... to present the input pattern to th... network for each of the sample... alphabets.

3.2 Neural Network Architec...

The structural design of th... based on a fully connected fe... generalized perceptron. Four inpu... two hidden layers (with five neur... and six neurons in second archite... in the output layer. The hidde... investigate the effects of Back-p... the hyperspace. Each network h... with the following activation a...

$$A^o_k = \sum_{i=0}^{H} w_{io_k} O^h_i$$

$$f(A^o_k) = f\left(\sum_{i=0}^{H} w_{io_k} \right.$$

3. Simulation Design and Implementation Details

This section describes the experiments designed to evaluate the performance of feed-forward neural network when evolved with two different learning algorithms. We have considered two different architectures, first consisting 4 neurons in the input layer, 5 neurons in each hidden layers, 2 hidden layers and 5 neurons in the output layer, while the second architecture consists 4 neurons in the input layer, 6 neurons in each hidden layers, 2 hidden layers and 5 neurons in the output layer.

3.1 Experiment

Five different sets of alphabets are used for both the experiments. The alphabets are converted into their density function by using MATLAB program, for input data. We used two different learning algorithms (Back propagation and Genetic Algorithm). In each experiment we have taken five trials to train the neural network. For this, the scanned images of five different samples of handwritten English alphabets were obtained. After collecting these samples, each English alphabets image was partitioned into four equal parts, and the density values of the pixels for each part were calculated. Next, the density values of the central of gravities for these partitioned images of the English alphabets were calculated. Consequently, four values were obtained from an image of handwritten English language alphabets, which were then used as the input for the feed-forward neural network. This procedure was used to present the input pattern to the feed-forward neural network for each of the sample of English language alphabets.

3.2 Neural Network Architecture

The structural design of the neural network was based on a fully connected feed-forward multilayer generalized perceptron. Four input units were used, with two hidden layers (with five neurons in first architecture and six neurons in second architecture) and five neurons in the output layer. The hidden layers were used to investigate the effects of Back-propagation and GA on the hyperspace. Each network had a single output unit with the following activation and output functions.

$$A^o_k = \sum_{i=0}^{H} w_{ik} O^h_i \qquad \ldots (3.1)$$

$$\int (A^o_k) = \int \left(\sum_{i=0}^{H} w_{ik} O^h_i \right) = O^o_k \qquad \ldots (3.2)$$

where function $\int (A^o_k)$ is given as,

$$O^o_k = \frac{1}{1 + e^{-KA^o_k}} \qquad \ldots (3.3)$$

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

$$A^h_k = \sum_{i=0}^{N} w_{ik} O_i \qquad \ldots (3.4)$$

$$\text{and} \quad O^h_k = \frac{1}{1 + e^{-KA^h_k}} \qquad \ldots (3.5)$$

$$\text{and} \quad O^i_k = \int (A^i_k) = A^i_k \qquad \ldots (3.6)$$

In the Back-propagation learning algorithm, the change in weight populations was done according to the calculated error in the network after each of the iteration of training. The genetic algorithms evolve the population of weights using its operators, and select the best population of the weights that minimize the error between the desired output and the actual output of neural network system.

3.3 The Genetic Algorithm Implementation

The initial population was generated with randomly assigned values for weights and biases. The values were obtained from the random generator generating values between 0 and 1.

After the initial population of weights and biases is created, the problem domain is represented as a chromosome. For that, the set of weight values is represented as a square matrix, in which a real number corresponds to the weighted link from one neuron to another neuron. A zero value means that there is no connection between the two given neurons.

Each row of this matrix represents a group of incoming weighted links to a single neuron. In total, there are 102 weighted links between neurons and 19 biased values of neurons in the neural network. Thus, a chromosome is a collection of genes representing either a weight or a biased value. A 121-gene chromosome can be represented as indicated in Table 1where some genes correspond to a single weighted link and some correspond to biased values of neurons in the network.

A mutation operator which randomly selects a gene in a chromosome and adds a small random value

Table 1. Format of 121-gene

w_{11}	w_{21}	Bias of hidden unit 1 of layer 1	w_{12}	w_{22}	Bias of hidden unit 2 of layer 1	------	w_{11}	w_{21}	-------	w_{61}	Bias of unit 1 of ou layer

between -1 and 1 to that particular gene produces the next generation population of 121-gene chromosomes. The size of the next generated population will be n+1, if the mutation operator applied n times over the old chromosome.

Elistim was used for creating each generation so that the genetic operators do not lose good solutions. This involved copying the best-encoded network unchanged into the new population.

This process selects among the mutated population of chromosomes the one for which the sum of squared errors is minimum for the feed-forward neural network. After assigning the values, the network architecture will be able to fabricate output using these assigned values. For each chromosome of new population, the error can be calculated. Now the selection operator will pick a chromosome from new population, which generates minimized error for the network.

A fitness evaluation function defines a function for evaluating the chromosome performance. This function must estimate the performance of weight population of a given feed-foward neural network. A simple function based on the proportion of the sum of squared errors is defined and applied. To evaluate the fitness of a given chromosome, each weight and biased value contained in the chromosome is assigned to the respective link and neuron in the network. The training set is then presented to the network, and the sum of squared errors is calculated. The smaller the sum, the higher is the fitness of the chromosome. In other words, the genetic algorithm attempts to find a set of weights and biased values that minimizes the sum of squared errors.

4. Results

The results presented in this Section demonstrate that, within the simulation framework presented above, large significant difference exists between the performance of Back-propagation feed-forward neural network and genetic feed-forward neural network for handwritten English alphabets recognition problem.

The results for handwritten English alphabets recognition problem performed (5 times) with the two algorithms up to having the maximum limit of 50000 iterations. All the results take five different types of handwritten samples for each English alphabet.

The results show (Table 2 and Table 3) how the number of convergence matrices for each alphabet recognition process. This value shows how many number of convergence weight matrices have been obtained for the particular character by applying the genetic algorithm. For Back-propagation algorithm the same entry is not required, because if the characters are correctly recognized by the network, then only one convergence matrix will be obtained. Therefore, this entry is required only for genetic algorithm through which multiple number of convergence matrices can be obtained. This shows the higher accuracy of the algorithm for character recognition.

Entries with real numbers (less than 1) in these tables represent the error existed in the network after executing the simulation program for 50000 iterations, i.e. up to 50000 iterations the algorithm could not converge a sample of a handwritten English alphabets into the feed-forward neural network. The simulation programe, which was developed in MATLAB 6.5, to test the two algorithms for the classification problem of handwritten English alphabets, generates the initial weights randomly through its random generator. As a result, the epochs for the algorithms are different every

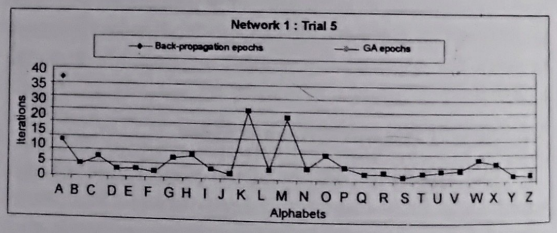

Figure 1. Comparison chart of the iterations preformed by Back-propagation algorithm and Genetic algorithm for 1st network

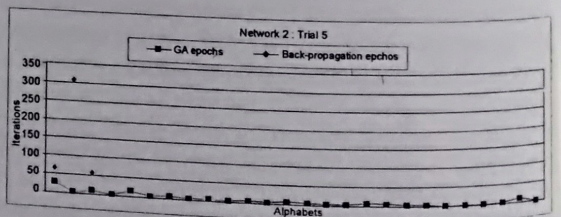

Figure 2. Comparison chart of the iterations preformed by Back-propagation algorithm and Genetic algorithm for 2nd network

Table 2. Average Iterations/Network e written English alphabets, calcualted (4, 5, 5, 5).

Sl. No.	Characters	Back-propagation Epoch Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
1.	A	.34	0.3	0.4	764	3
2.	B	0.4	0.4	0.4	0.36	0
3.	C	0.3	0.3	0.3	0.3	0
4.	D	0.4	0.4	0.4	0.4	0
5.	E	0.3	0.3	0.3	0.3	0
6.	F	0.3	0.3	0.3	0.3	0
7.	G	0.2	0.2	0.2	0.2	0
8.	H	0.4	0.4	0.4	0.4	0
9.	I	0.3	0.3	0.3	0.3	0
10.	J	0.3	0.3	0.3	0.3	0
11.	K	0.2	0.2	0.2	0.2	0
12.	L	0.3	0.3	0.3	0.3	0
13.	M	0.2	0.2	0.2	0.2	0
14.	N	0.2	0.2	0.2	0.2	0
15.	O	0.1	0.1	0.1	0.1	0
16.	P	0.4	0.4	0.4	0.4	0
17.	Q	0.3	0.3	0.3	0.3	0
18.	R	0.3	0.3	0.3	0.3	0
19.	S	0.2	0.2	0.2	0.2	0
20.	T	0.3	0.3	0.3	0.3	0
21.	U	0.2	0.2	0.2	0.2	0
22.	V	0.2	0.2	0.2	0.2	0
23.	W	0.1	0.1	0.1	0.1	0
24.	X	0.3	0.3	0.3	0.3	0
25.	Y	0.2	0.2	0.2	0.2	0
26.	Z	0.2	0.2	0.2	0.2	0

* C.M.: Convergence Matrices

time with the same network st training data set.

5. Conclusion

The results described in th for the handwritten English a problem, feed-forward neural Back-propagation algorithm doe comparison to feed-forward neur

Table 2. Average Iterations/Network error and number of covergence weight matrices of five different samples of hand written English alphabets, calcualted by performing Back Propagation Algorithm and Genetric Algorithm for 1st network (4, 5, 5, 5).

Sl. No.	Characters	Back-propagation Epochs					GA Epochs									
		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Tiral 1	No. of C.M.*	Trial 2	No. of C.M.*	Trial 3	No. of C.M.*	Trial 4	No. of C.M.*	Trial	No. of C.M.*
1.	A	.34	0.3	0.4	764	37.4	21.4	1804.4	26	1448.8	32.6	1446.8	22.8	1819.6	13.4	1444.8
2.	B	0.4	0.4	0.4	0.36	0.36	8	1455.8	2.4	1808.2	2	1448	3.8	1453.6	4.8	1616
3.	C	0.3	0.3	0.3	0.3	0.3	9	1725.6	5.2	1814	1.6	1247.2	8.8	1443.2	7.4	1479.2
4.	D	0.4	0.4	0.4	0.4	0.4	1.4	1116.2	9.4	1440.2	3.4	1454.8	5.4	1449.8	3.2	1447.4
5.	E	0.3	0.3	0.3	0.3	0.3	18.8	1448.6	16.8	1444.2	1.8	1454.8	17	1437.8	3.2	1447.4
6.	F	0.3	0.3	0.3	0.3	0.3	3.6	1449.8	1.6	1444.2	2.2	1443.2	3	1454.4	2.2	1441.8
7.	G	0.2	0.2	0.2	0.2	0.2	6.4	1444.8	12.4	1819.4	1.2	1434.8	11.2	1452.6	7.6	1806.2
8.	H	0.4	0.4	0.4	0.4	0.4	1.8	1001.4	3.6	1445.6	2.8	791	2.2	1443.8	8.6	1436
9.	I	0.3	0.3	0.3	0.3	0.3	5.2	1448.2	1.6	1446.8	2	1249.8	6.2	1438.8	3.4	1448.8
10.	J	0.3	0.3	0.3	0.3	0.3	1.4	1385	3	1291	1.6	1448	5.4	1443	1.6	1444
11.	K	0.2	0.2	0.2	0.2	0.2	1.2	1444.8	6.2	1356.4	1	1090.6	17.2	1452.4	25.4	1452.6
12.	L	0.3	0.3	0.3	0.3	0.3	4.6	1454.4	3.8	1451	3.6	1249.4	2.6	1448	3	1444
13.	M	0.2	0.2	0.2	0.2	0.2	1.4	1449.4	12	1439	1	1459.4	1.2	1443.4	23.2	1211.6
14.	N	0.2	0.2	0.2	0.2	0.2	2.8	1099.6	2.6	1317.8	1.2	1347.2	1.6	1407	3.6	1811.4
15.	O	0.1	0.1	0.1	0.1	0.1	6.8	1210.8	1.4	1452.6	1	1453.6	4.4	1438.4	9.2	1436.6
16.	P	0.4	0.4	0.4	0.4	0.4	7.8	1441	3.8	1449	3.8	1445	1.8	1133	4.8	1447
17.	Q	0.3	0.3	0.3	0.3	0.3	2.6	1242.8	3.6	1443	1	1453.4	4	1444.2	2.4	1447.4
18.	R	0.3	0.3	0.3	0.3	0.3	1.2	862.2	2.4	1443	1.8	1445	12	1251	2.8	1148
19.	S	0.2	0.2	0.2	0.2	0.2	2.4	1447.2	5.4	1314.2	1.2	1103.8	4.6	887.4	1.4	1454
20.	T	0.3	0.3	0.3	0.3	0.3	4	1195.2	2	1154.8	2.6	969	1.4	1096.2	2.6	1457.8
21.	U	0.2	0.2	0.2	0.2	0.2	1.2	1446.4	11	1450.4	2.4	914	1	1134.6	3.4	1451.8
22.	V	0.2	0.2	0.2	0.2	0.2	4.4	1453.2	2.6	1383	1.2	1440.8	5.6	1438.4	3.6	1306
23.	W	0.1	0.1	0.1	0.1	0.1	3	1471.2	1	1453.6	1.4	1446.6	2.8	1193.8	7.8	1249.6
24.	X	0.3	0.3	0.3	0.3	0.3	4.2	1457.2	1.8	1317.6	2.4	1446.6	8.6	1445.4	6.6	920.2
25.	Y	0.2	0.2	0.2	0.2	0.2	7.2	1440.2	3.6	1133.8	1	1447	3.2	1101.6	1.6	1446.4
26.	Z	0.2	0.2	0.2	0.2	0.2	2	1276.4	2.2	1445.4	1.4	1023	4	1455	2	908.8

* C.M.: Convergence Matrices

time with the same network structure and the same training data set.

5. Conclusion

The results described in this paper indicate that, for the handwritten English alphabets recognition problem, feed-forward neural network trained with Back-propagation algorithm does not perform better in comparison to feed-forward neural network trained with genetic algorithm. The performance of genetic algorithm is efficient and accurate in all the simulations.

It is found that, in each and every case, the genetic algorithm gives more than one convergent weight matrices for every input pattern in comparison to the Back-propagation algorithm. This shows the higher accuracy rate in the pattern recognition with genetic algorithm. The higher number of convergence weight

Table 3. Average Iterations/Network error and number of covergence weight matrices of five different samples of hand written English alphabets, calcualted by performing Back Propagation Algorithm and Genetric Algorithm for 2nd network (6, 6, 6, 5).

Sl. No.	Alphabets	Back-propagation Epochs					GA Epochs									
		Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Tiral 1	No. of C.M.*	Trial 2	No. of C.M.*	Trial 3	No. of C.M.*	Trial 4	No. of C.M.*	Trial	No. of C.M.*
1.	A	965.2	5073	8.8	8.8	68.4	26.6	1843.4	24.4	1468	35.8	1474	17.2	1663.6	28.2	1481.6
2.	B	0.38	0.4	156	17.4	308	8.6	1372	3.8	1475.8	2.6	1280.2	2.2	1479.6	2.2	1482.6
3.	C	0.3	0.3	479	539	58	5.2	1471	25	1486.6	19.2	1473	10	1847.6	9.2	1848
4.	D	0.4	0.3	0.38	1193	0.36	1.8	1210	4.4	1479.2	5	1472.6	1.6	1231.6	2.6	1344.6
5.	E	0.3	0.3	0.3	0.3	0.3	13	1479.4	7.8	1478.2	4	1487.4	5	1480.4	15.2	1483.2
6.	F	0.3	0.3	0.3	103	0.3	1.6	1475	1.8	1128.4	4.6	1371.8	3	1479.8	3.4	1481.4
7.	G	0.2	0.2	0.2	0.2	0.2	7.2	1854.4	10	1487	2.4	1484	3	13480.2	7.6	1120.6
8.	H	103	0.4	0.4	0.4	0.4	2	778.2	7.6	1633.4	1.8	805.6	2.2	1093	2.4	1194
9.	I	315	0.3	0.3	0.3	0.3	1.4	1294.8	7	1473.6	4.6	1114.6	1	1354	5.6	1481.2
10.	J	0.3	0.3	0.3	0.3	0.3	2.6	1212	1.4	912	1.2	1230	3.6	1481	2.8	1481
11.	K	0.2	0.2	0.2	0.2	0.2	7	1836	3.4	1850.6	4.4	1485.4	2.4	1470.6	5.2	1483.2
12.	L	0.3	0.3	0.3	0.3	0.3	2	1472	2.8	1279.8	1.4	1258.6	4.8	1229.2	2.4	857.6
13.	M	0.2	0.2	0.2	0.2	0.2	15	1483.6	2.4	1402.4	2.6	1486	10.4	945.4	8.8	1483
14.	N	0.2	0.2	0.2	0.2	0.2	3	1481.4	2	1083.8	1.6	892.8	1.2	523.6	1.4	1597
15.	O	0.1	0.1	0.1	315	0.1	3.4	1477.4	9	1485.8	11.8	1484	6	1482.6	2.2	1390.8
16.	P	0.4	315	0.4	0.4	0.2	2.4	869	3	842	2	280	1.8	714	2.4	923
17.	Q	0.3	0.3	0.3	0.3	0.3	4.2	1490.4	2	1341	1.6	1195.6	1.2	1254.6	7	1349
18.	R	0.3	0.3	0.3	0.3	0.3	2.4	1186	1.6	1484	1.2	1157	1.2	612.4	5.2	1479
19.	S	0.2	0.2	315	0.2	0.2	1	1480.6	3.8	1323.4	4.8	1471	1.2	1449	2.4	1479.4
20.	T	0.3	0.3	0.3	0.3	0.3	7	1009.8	1.6	1348.8	3	1473.8	1.4	668.2	1.8	516.4
21.	U	0.2	0.2	0.2	0.2	0.2	5.2	1479.8	10.8	1488.6	4.6	862.8	2.6	1484	1	1114.2
22.	V	0.2	0.2	0.2	0.2	0.2	1.8	1210.8	2.6	1479	2.2	1469	2.4	1255	1.2	1134.2
23.	W	0.1	0.1	3222	0.1	0.1	2.6	1478.4	2.8	1246.4	9.6	1476.6	1.4	1850.4	1	1477
24.	X	0.3	3222	0.3	0.3	0.3	2.2	1209	2.6	1375.4	4.6	1474.2	2.4	819.8	2.4	882
25.	Y	0.2	0.2	0.2	0.2	0.2	6.8	1859.6	2	1485.8	3	1475.6	3.6	1485.6	11.2	1476.6
26.	Z	0.2	2684	0.2	0.2	0.2	2	1128.6	1.6	1384.4	1.4	922.2	2.2	945.2	1.2	1304

* C.M.: Convergence Matrices

matrices in the GA training process, suggests that this algorithm may not be trapped in the false minima of the error surface. It may also minimize the possibilities of misclassification for any unknown testing input pattern.

By comparing the results in Table 4.1 & 4.2, it is also found that by increasing the number of neurons in the hidden layers of a feed forward neural network, the performance of the network increases significantly.

REFERENCE

1. Pat Christenson, As... Handwriting recognition... the URL: htt://csci.mr... CSCi4555s04/InsertTe... 2005.
2. Mangal M., Manu Pr... classification for the mu... problem with hybrid ... network. Neurocomput...
3. Mangal M., Manu... multidimensional X... evolutionary feed-forw... Journal on Artificial I... 2007, pp. 111-120.
4. McCulloch W.S. and ... ideas immanent in ne... Vol. 5, 1943, pp. 115-...

written
, 6, 5).

No. of C.M.*
481.6
482.6
848
344.6
483.2
481.4
120.6
194
481.2
481
883.2
57.6
883
997
990.8
3
49
79
79.4
6.4
14.2
34.2
77
2
76.6
04

it is
s in
the
.

REFERENCE

1. Pat Christenson, Andrew Maurer, Grant Miner, Handwriting recognition by neural network, extracted by the URL: htt://csci.mrs.umn.edu/UMMCSciWiki/pub/CSCi4555s04/InsertTeamNameHere/handwriting.pdf, 2005.
2. Mangal M., Manu Pratap Singh, Analysis of patern classification for the multidimensional parity-bit-checking problem with hybrid evolutionary feed-foward neural network: Neurocomputing, Vol. 70, 2007, pp. 1511-1524.
3. Mangal M., Manu Pratap Singh, Analysis of multidimensional XOR classification problem with evolutionary feed-forward neural networks: International Journal on Artificial Intelligence Tools, Vol. 16, issue 1, 2007, pp. 111-120.
4. McCulloch W.S. and Pitts W., A logical calculus of the ideas immanent in nervous activity: Bull. Math. Biopy. Vol. 5, 1943, pp. 115-133.
5. Rosenblatt, The perceptron: A probabilistic model for information storage and organization in the brain: Psychological Review, Vol. 65, 1958, pp. 386-408.
6. Seiffert U., multiple layer perceptron training using Genetic Algorithms: European symposium on Artificial Neural Networks (ESANN 2001), 2001, pp. 159-164.
7. Sontag E.D., Feed-forward nets for inerpolation and classification: J. Computing System Sciences, Vol. 45, 1992, pp. 20-48.
8. Sprinkhuizen G., Kuyper, Egbert, Boers J.W., The local minima of the error surface of the 2-2-1 XOR network: Annals of Mathematics and Artificial Intelligence, Vol. 25, Issue 1-2, 1999, pp. 107-136.
9. Yahya H. Zweiri, Lakmal D. Seneviratne, Kaspar Althoefer. Stability analysis of a three-term backpropagation algorithm: Neural Network, Vol. 18, Issue 10, 2005, pp. 1341-1347. ❐