

Performance Analysis of feed forward neural networks for the Recognition of Scaled and Rotated Hand Written Alphabets Using Soft-Computing Technique

S. R. Pande

Associate Professor Deptt. of Computer Science, SSES Amti's Science College, Congressnagar, Nagpur, India

M:+919422804245

srpande65@rediffmail.com

Saurabh Srivastava

Assistant Professor Deptt. of Mathematical Sciences and Computer Applications Bundelkhand University, Jhansi

hanu.saurabh@gmail.com

Rajesh Lavania

Assistant Professor Deptt. of Computer science & Engineering, Institute of Engg. & Tech., Dr. B. R. Ambedkar University, Khandari, Agra

M:+919410423311

r11304@rediffmail.com

Manu Pratap Singh

Associate Professor Deptt. of Computer, Institute of Computer & Information Science,

Dr. B. R. Ambedkar University, Khandari, Agra.

M:+919319102434

manu_p_singh@hotmail.com

ABSTRACT

In this paper we are analyzing the performance of feedforward neural networks with hybrid evolutionary algorithms for the recognition of hand written English alphabets after their scaling and rotation in the test pattern set whereas in the training set straight handwritten alphabets have used. In this performance analysis the neural network is trained with hybrid evolutionary algorithm for the given training set of handwritten English alphabets and the performance of trained neural network is analyze for the test pattern set of scaled and rotated form of the patterns used in the training set. To accomplish the task we have taken two samples of each hand written English alphabets, one is considered as the training sample and another is considered as the unknown sample of the test pattern set, after applying the general mathematical algorithm for rotation and scaling for this sample along both X and Y axis. The random genetic algorithm and the hybrid evolutionary algorithm are applied to train the network for the samples of training set. Network trained with straight alphabet samples have been used to recognize the rotated and scaled samples from the set of already trained five straight alphabet's sample. The hybrid evolutionary algorithm is taking the definite lead on the conventional approaches of neural network and soft computing techniques. The results of the experiments clearly show that the hybrid approach is efficient for the recognition of hand written samples of any shape and size most reliably.

Categories and Subject Descriptors

I.5.2 [Pattern Recognition-Design Methodology]

General Terms

Algorithms, Experimentation, Performance

Keywords

Soft Computing, Pattern Recognition, Backpropagation learning rule, hybrid evolutionary algorithm, feed-forward neural networks.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

© Copyright 2011 Research Publications, Chikhli, India

1. INTRODUCTION

This new stage in the evolution of handwriting processing results from a combination of several elements: improvements in recognition rates, the use of complex systems integrating several kinds of information, the choice of relevant application domains, and new technologies such as high quality & high speed scanners and inexpensive powerful processors[1]. Methods and recognition rates depend on the level of constraints on handwriting. The constraints are mainly characterized by the types of handwriting, the number of *scripter*, the size of the vocabulary and the spatial layout. Recognition strategies heavily depend on the nature of the character set to be recognized. Intense activity was devoted to the character recognition problem during the seventies and the eighties and pretty good results have been achieved [2]. Character recognition techniques can be classified according to two criteria: the way preprocessing is performed on the data and the type of the decision algorithm. There have been reports of successful use of neural networks for the recognition of handwritten characters [3-5, 17], but it is very difficult to find any general investigation which might shed light on the systematic approach of a complete neural network system for the automatic recognition of cursive character.

Traditionally, the recognition of hand-written characters has been considered of importance for various applications in which the automatic reading of hand written text is needed. More recently, the recognition of hand-written characters has gained importance to develop the intelligent system. The hand written characters may have the different shapes and different size. It may be of the type straight, scaled and / or rotated. The human can easily recognize the character of any shape & size and recall the correct memorized one. But for the machine every alphabet after scaling or/ and rotation becomes new pattern. So the recognition of the hand written character is very difficult-indeed, the symbols can be of any size and orientation in the image frame. The recognition rate for rotated and / or scaled hand written characters cannot be as good as the straight characters.

The recognition rate depends on the number of writers and their training. If the number of writers is more, recognition can be more difficult. There are several techniques to recognize the hand written character which are in the straight form but a very few work have been done for scaled and rotated hand written alphabets [6]. A feature based approach is also suggested [7, 8] that are not sensitive to rotations, translations, and scaling, and are

resistant to noise and distortions as well. Nawwaf & Rabab [9] also proposed a simple and computationally efficient mapping, which can be used for character recognition by taking the application of hand-written Arabic characters, and explained that it (taken with other features of the character) can be used to produce an effective recognition system to identify each character uniquely. The recognition process for a large set of complex problems such as hand written characters after scaling and rotation mostly depends on the way of training. Efficient learning of a pattern largely depends upon the training methods. The process of learning of the pattern may be divided into two basic principals ‘on-line learning’ and ‘off-line’ learning. In an off-line learning the given patterns are used together to determine the weights. On the other hand in an on-line learning the information in each new pattern is incorporated into the network by incrementally adjusting the weights [10]. Thus, an on-line learning allows the network to update the information continuously. The powerful combination of analytical methods provides more insight and deeper understanding of on line learning algorithms, and leads to novel and principled proposals for their improvement. The results of the handwritten English straight alphabets recognition problem clearly shows that [17], 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 higher number of convergence weight matrices in the hybrid EA training process suggests that this algorithm may not be trapped in the false minima of the error surface [11]. The performance of hybrid evolutionary algorithm has been found better in comparison to the genetic algorithm by finding the less number of iterations and higher rate of convergence for character recognition [17].

In this paper feedforward neural networks of different architectures are trained with hybrid evolutionary learning algorithm i.e. the combination of generalized delta learning rule and genetic algorithm for the given training set of handwritten English alphabets. To evaluate the performance of trained neural networks the test pattern set is considered. The test pattern set is consisting with scaled and / or rotated form of the alphabets those have been used in the training set. The performance of the neural networks is analyzing on the basis of number of epoch, convergence and number of converged weight matrices during the training and the success of networks for the recognition of the scaled and / or rotated samples of the test pattern set. The recognition for any test pattern is considered in terms of the trained networks response for the presented pattern. It means that on the presentation of scaled and / or rotated sample for the trained pattern, networks should produce the same desired or expected output that has been used during the training of networks for the normal shape of the same pattern. It has been observed that the performance of online training by hybrid evolutionary algorithm is found better in terms of number of epoch, convergence and more number of convergence weight matrices in comparison to random genetic algorithm. The rate of recognition for scaled and / or rotated hand written English alphabets is also found very high for the hybrid evolutionary algorithm.

Rest of the paper is containing four more sections follow with the references in the last. In Second section the feature extraction for the training and test pattern set is discussed. The third section of the paper deals with simulation and implementation details. Section four shows the results of the experiments. In section fifth

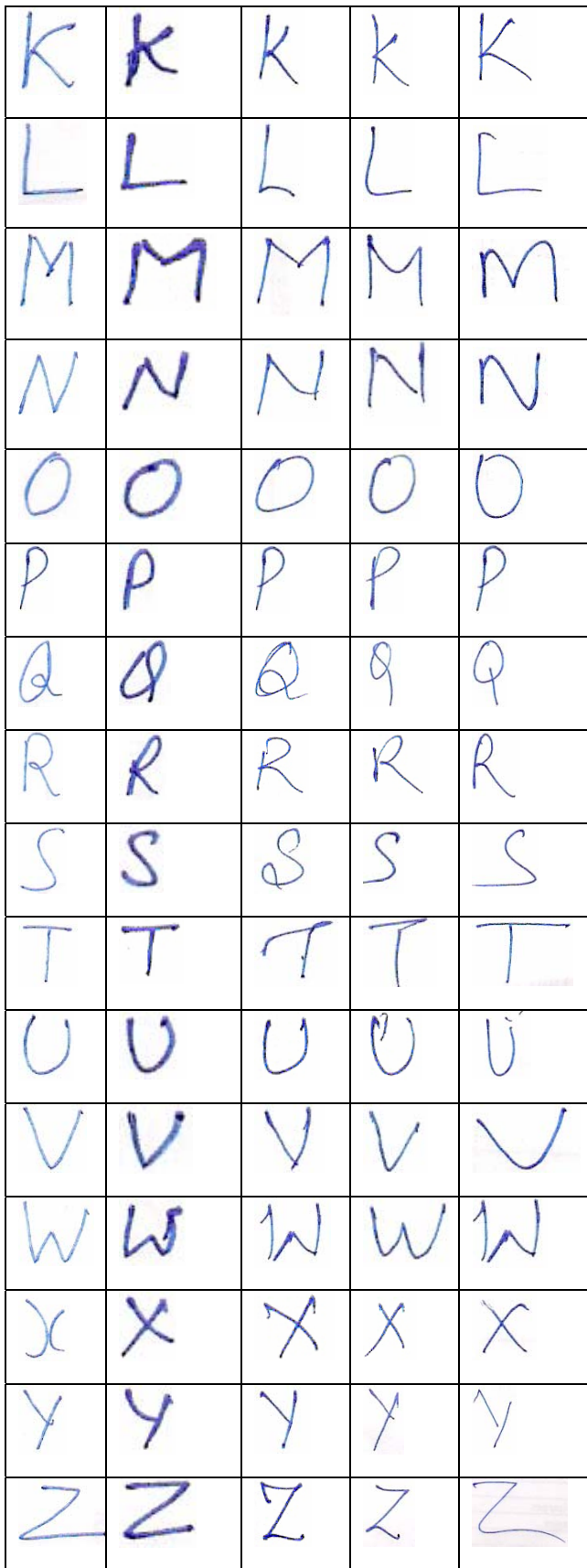
discussions and conclusion have been made with the future works followed by references.

2. FEATURE EXTRACTION

All The training set which is used by the networks for learning is consisted with the patterns of handwritten English alphabets of normal shape. Total five samples of handwritten English alphabets have been collected from five different people. In this way total 130 input samples are used for the training. Each input sample of alphabet used as pattern, first scanned and then converted to bitmap image. Thus, the samples of handwritten English alphabets are considered as images. The scanned images of five different samples of handwritten English alphabets as shown in Table 1 were obtained [17].

Table 1. Scanned images of five different samples of handwritten English alphabets

A	A	A	A	A
B	B	B	B	B
C	C	C	C	C
D	D	D	D	D
E	E	E	E	E
F	F	F	F	F
G	G	G	G	G
H	H	H	H	H
I	I	I	I	I
J	J	J	J	J



All collected hand written alphabets images were 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 were calculated. Consequently four values were obtained from an image of handwritten English alphabets, which were then used as the input pattern vector of dimension four for the training with feed-forward neural network architectures. This procedure was used to present the input pattern to the feed-forward neural networks for each of the sample of English Alphabets. The same procedure has been applied for the formation of test pattern set but instead of considering the normal shape of the handwritten English alphabets, the scaled and / or rotated form of the samples those have been used in the training set are considered here. Thus, there are 260 samples in scaled and rotated form of training samples have been used as the test patterns to analyze the performance of feed forward neural networks with hybrid evolutionary learning algorithm and random genetic algorithm. Hence, to construct the scaled and rotated form for the samples of training set the image transformation technique is used.

In geometry and linear algebra, the scaling is defined [12, 13] as mathematics to resize the image onto the X and Y axis (for two dimensional objects). The more obvious and common way to change the size of an image is scaling the image. The content of the image is enlarged by increasing the size of the pixel values of the image. But while the actual image pixels and colors are modified, the content represented by the image is essentially left unchanged. Scaling of an image makes a drastic change to the content of the image. An image can be represented as pixel matrix and a scaling can be represented by a scaling matrix. On the other hand, the rotation [14] is a transformation in a plane that describes the change in the orientation of an image. A rotation is different from a translation, which has no fixed points, and from a reflection, which "flips" the image it is transforming. This overall process of change the shape, position and orientation of an image is known as the transformation of image. When an image is moved to stationary co-ordinate system or background, referred as geometric transformation and applied to each point of an image. And while the co-ordinate system is moved relative to the image and image is held stationary then this process is termed as a co-ordinate transformation. In the process of transformation every image is assumed as a set of points. Every point P of the image has co-ordinate (x, y) and the complete image is defined as sum of total of all co-ordinates points [15]. Image rotation is a transformation where the image is rotated θ° about origin [16] and the co-ordinate of new point is given as:

$$P' = R_\theta(P) \tag{2.1}$$

Let us consider that the co-ordinate of a point P is (x, y) and after rotation the coordinate of point P' is (x', y') as shown in figure 1. If initially P is at the α angle from x-axis then we have,

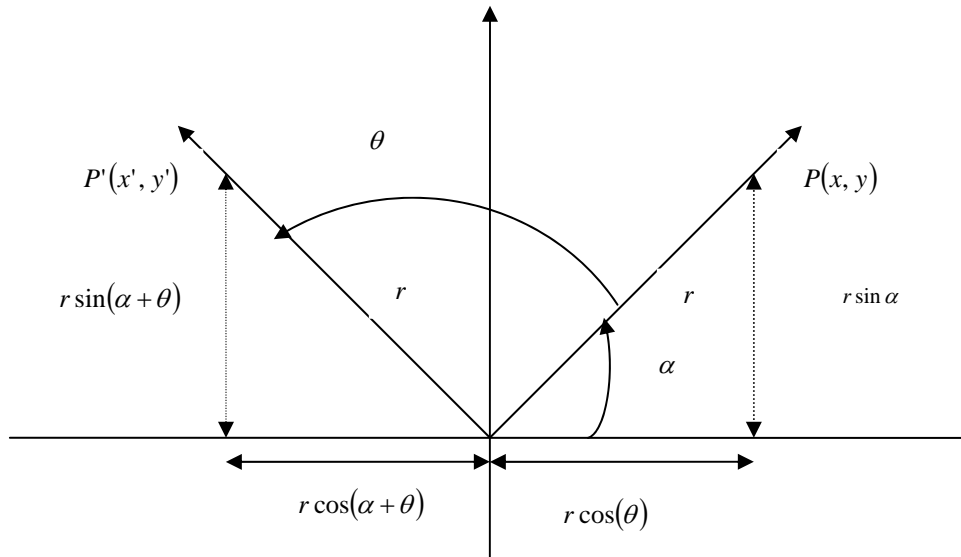


Figure 1. The diagrammatical representation of rotation.

$$x' = r \cos(\alpha + \theta)$$

$$= r \cos \alpha \cos \theta - r \sin \alpha \sin \theta$$

since $x = r \cos \alpha$ and $y = r \sin \alpha$

therefore $x' = x \cos \theta - y \sin \theta$ (2.2)

and $y' = r \sin \alpha \cos \theta + r \sin \theta \cos \alpha$

$$y' = x \sin \theta + y \cos \theta$$

So, the rotation matrix can be defined

as $R_\theta = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$, by multiplying this to the original

image we can get a rotated image whereas the scaling is the process of changing the size and proportion of the image. There are two factors used in scaling transformation i.e. S_x and S_y , where S_x is a scale factor for the x co-ordinate and S_y is scale factor of y co-ordinate. Let us assume that the co-ordinate of a point P is (x, y) and after scaling the coordinate of point P' is (x', y') then we have,

$$P' = S_{S_x, S_y}(P) \tag{2.3}$$

Where

$$x' = S_x \cdot x \quad \text{and} \tag{2.4}$$


$$y' = S_y \cdot y$$



Or we may define the scaling matrix as $S_{S_x, S_y} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$.

Thus, by multiplying the original image with the scaling matrix we can get scaled image.

Thus samples of test pattern set are considered on applying these scaled and rotation transformation on the samples of training set. So that the nature and the density values of alphabets are changed after rotation and scaling transformations. Thus, it is obvious that by applying these transformations the samples of training set are considered as the noisy or distorted form of the original sample. An example for the handwritten alphabet 'A' and its density values in its normal form and after the transformations can be seen in the table 2.

Table 2. The example for the alphabet 'A' in normal and after the transformation with Density values

	Alphabets	Density values Matrix
Straight Sample		2.466523
		2.190908
		2.350251
		2.361397

	Alphabets	Density values Matrix
Scaled 2X3 Sample		2.409672 2.550000 2.455222 2.550000
Rotate By 30°		2.302031 2.443418 2.262767 2.399731

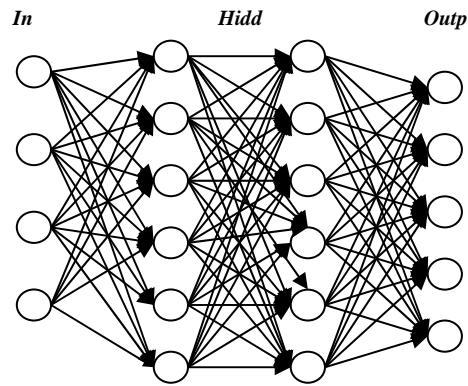


Figure 2. The Architecture of used Neural Network.

Now, we consider here the extended training set from the training set of handwritten straight English alphabets. This extended pattern set consists with the scaling and rotated form of the pattern samples of handwritten straight English alphabets. It has been seen [17] that during the recognition of straight hand written English alphabets (A to Z), on five trials for each training set the total number 1475(approx.) of converged weights obtained for hybrid evolutionary algorithm and 965 (approx.) for random genetic algorithms. Thus there is 1475 and 965 number of optimal weight vectors explored, on which (for any one of them) all the patterns of training set have properly trained and also the performance has verified from the test pattern set. Hence from the above mentioned analysis, it is quite clear that any one of the weight vector (Converged or Optimal) can be selected for further training for the extended training set. The chromosome of 737-genes of the selected weight vector (Converged or Optimal) shown in figure 3 is used as the initial population of the solution.

3. SIMULATION DESIGN AND IMPLEMENTATION DETAILS

The problem of straight hand written English character recognition has already been discussed [17] for feedforward neural network architecture with three different algorithms (Generalized delta learning rule, random GA and Hybrid Evolutionary learning). Here in this paper we consider two neural network architectures those are trained with training set of hand written straight English alphabets i.e. without scaling and rotation. The performance of these networks is verified with test pattern set of rotated and scaled form of the pattern samples used in the training set. As each network consist with two hidden layers with input and output layers. The difference between these two neural network architectures is in terms of number of neurons in each hidden layer, the first architecture is using five neurons in each hidden layer whereas the second architecture is using six neurons in each hidden layer. The analysis of the training and testing for these two architecture with the three algorithms(BP, Random GA, and Hybrid EA), has suggested that the network with six neurons in each hidden layer performs better in terms of number of iterations and number of converged weight vectors [17]. Thus, on the basis of this analysis, we consider the neural network with six neurons in two hidden layers (as shown in figure 2), for the training with given training set of handwritten straight English alphabets.

In this simulation design, the feedforward neural network architecture consist with four neurons in the input layer, two hidden layers with six neurons in each and five neurons in the output layer. This neural network architecture is considered for the training purpose with the two algorithms i.e. random genetic algorithm and hybrid evolutionary algorithm. The input / output pattern vectors are constructed from the already existing training pattern set of straight alphabets. The network has already been trained for the existing training set for the five trials of each pattern [17]. Now we select any one of the pattern vector for each alphabet and apply the scaling & rotation randomly on the selected pattern. This modified pattern now becomes the new training sample for the extended training set. The network is trained for this extended training set with already converged weight vectors, in this manner the network is continuously adapting the change in its behavior corresponding to changing training set. Thus the training pattern set is continuously increasing with the newly modified pattern. The network is also expected to adopt the continuous changes of the training pattern set. Again, as the training process is progressed the more and more new converged weight vectors are explored. The performance of the network is analyzed with respect to random GA and hybrid EA in terms of the number of iterations and converged weight vectors. And also, after completion of the training up to a satisfactory level, a new randomly generated pattern (Not used in the extended training set) has used as test pattern to exhibit the generalization capability of trained neural network.

W_{11}^{IH}	W_{41}^{IH}	Bias of Hidden unit 1 of layer 1	W_{21}^{IH}	W_{42}^{IH}	Bias of Hidden unit 2 of layer 1	W_{16}^{IH}	W_{46}^{IH}	Bias of Hidden unit 6 of layer 1
W_{11}^{HO}	W_{61}^{HO}	Bias of unit 1 of Output layer	W_{12}^{HO}	W_{62}^{HO}	Bias of unit 1 of Output layer	W_{15}^{HO}	W_{65}^{HO}	Bias of unit 5 of output layer
W_{11}^{HH}	W_{61}^{HH}	Bias of Hidden unit 1 of layer 2	W_{21}^{HH}		W_{62}^{HH}	Bias of Hidden unit 2 of layer 1	W_{16}^{HH}	W_{66}^{HH}	Bias of Hidden unit 6 of layer 2

Figure 3. The Chromosome of selected weight vector.

3.1 The Neural Network Architecture

The architecture of the neural networks was based on a fully connected feed-forward multilayer generalized perceptron [17]. The hidden layers were used to investigate the effects of the algorithms on the hyper plane and the output units are used to represent classes for possible outputs. Each network had five output units with the following activation and output functions,

$$y_k^o = \sum_{i=0}^H w_{io_k} O_i^h \tag{3.1}$$

$$f(y_k^o) = f\left(\sum_{i=0}^H w_{io_k} O_i^h\right) = O_k^o \tag{3.2}$$

where function $f(y_k^o)$ is given as,

$$O_k^o = \frac{1}{1 + e^{-Ky_k^o}} \tag{3.3}$$

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

$$y_k^h = \sum_{i=0}^N w_{ik} O_i \tag{3.4}$$

and
$$O_k^h = \frac{1}{1 + e^{-Ky_k^h}} \tag{3.5}$$

and
$$O_k^i = f(y_k^i) = x_k^i \tag{3.6}$$

To accomplish the training, change in weight populations was done according to the calculated error in the network after each of the iteration of training. The change in weight and error in the network can be calculated as follows,

$$\Delta w_{ho}(s+1) = -\eta \sum_{i=1}^H \frac{\partial E}{\partial w_{ho}} + \alpha \Delta w_{ho}(s) + \frac{1}{1 - (\alpha \Delta w_{ho}(s))} \tag{3.7}$$

$$\Delta w_{ih}(s+1) = -\eta \sum_{i=1}^N \frac{\partial E}{\partial w_{ih}} + \alpha \Delta w_{ih}(s) + \frac{1}{1 - (\alpha \Delta w_{ho}(s))} \quad (3.8)$$

$$E = \frac{1}{2} \sum_{p=1}^P (O^o - T)^2 \quad (3.9)$$

where $(O^o - T)^2$ is the squared difference between the actual output value of the output layer for pattern p and the target output value. Doug's momentum term was used with momentum descent term for calculating the change in weights in equations (3.7) and (3.8). 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 1.0. After the momentum is added, the length of the resulting weight change vector can grow as high as $1 / (1 - \text{momentum})$. This change allows stable behavior with much higher initial learning rates, resulting in less need to adjust the learning rate as the training progresses. The evolutionary 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.

3.2 The Genetic Algorithm Implementation

A mutation operator which randomly selects a gene in a chromosome and adds a small random value between -1 and 1 to that particular gene produces the next generation population of 737-gene chromosomes. The total number of next generated population is $n + 1$, if the mutation operator applied n times over the old chromosome:

$$C^{new} = C^{old} \bigcup_{i=1}^n [C_{121-\lambda} \bigcup (C_{\lambda}^{old} + \epsilon)] \quad (3.10)$$

where C^{old} symbolizes the old chromosome of 737-gene, ϵ symbolizes the small randomly generated value between -1 to 1,

$$C^{next} = C^{sel} \bigcup_{i=1}^n \left[(C_{\alpha}^{sel} - C_{\alpha}^{sel} - C_{\beta}^{sel}) \bigcup \left(C_{\alpha}^{sel} \xleftrightarrow{v_{\alpha} \leftrightarrow v_{\beta}} C_{\beta}^{sel} \right) \right] \quad (3.11)$$

where α and β symbolize the randomly generated genes positions in CP_1 and CP_2 in C^{sel} chromosome, and C^{next} is the population of next generation.

λ symbolizes the randomly selected gene of C^{old} chromosome for adding the ϵ , and C^{new} symbolizes the next generation population of chromosome, i.e. $C^{new} = [C_1, C_2, C_3, \dots, C_n, C_{n+1}]$.

The inner \bigcup operator prepares a new chromosome at the each iteration of mutation², and the outer \bigcup operator builds the new population of chromosomes called C^{new} .

Elitism was used when creating each generation so that the genetic operators did not lose good solutions. This involved copying the best-encoded network unchanged into the new population as given in equation 3.10, which includes C^{old} for creating C^{new} .

Selection of a chromosome C^{sel} is made from the mutated population of chromosomes for which the sum of squared errors is minimized for the feed-forward neural network, i.e. iteratively all the chromosome values is assigned to the network architecture in terms of weights and biased values defined in chromosome. After assigning the values, the network architecture is able to fabricate output using these assigned values. For each new

chromosome C^{new} , the error can be calculated using these fabricated outputs as in equation 3.9. Next, the selection operator will pick a chromosome C^{sel} from C^{new} , which generates minimized error for the network.

Crossover operator takes selected chromosomes and creates $n + 1$ new generation of population each of the size 737-genes. This next generation of population is produced by applying the crossover operator n times. Experiments performed the crossover operation by swapping the two randomly selected gene values of the parent chromosome as given in Figure 4 and equation 3.11:

1.1.1 Chromosome

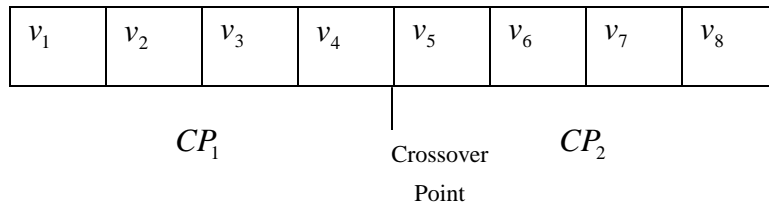


Figure 4 (A)

1.1.2 Chromosome

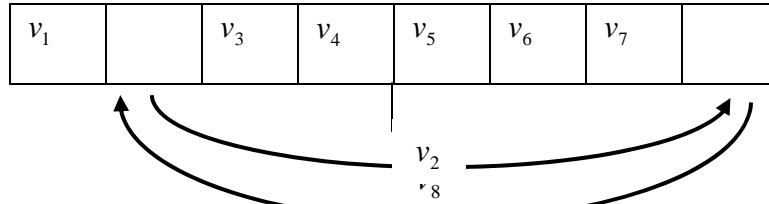


Figure 4 (B)

1.1.3 Chromosome

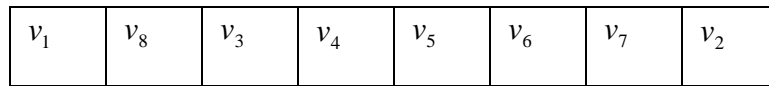


Figure 4 (C)

Figure 4. (A) Chromosome before applying crossover operator, (B) Applying crossover operator on chromosome, and (C) Chromosome after applying crossover operator

A *fitness evaluation function* defines a function for evaluating the performance of the selected population. This function must estimate the performance of weight population of a given feed-forward neural network. A simple function defined based on the proportion of the sum of squared errors is 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.

$$\begin{aligned}
 &\min error = 1.0 \\
 &\text{For all the } n+1 \text{ chromosomes} \\
 &\text{if } (\min error > E_{C_i^{next}}) \text{ then} \\
 &\quad (\min error = E_{C_i^{next}}) \\
 &\quad C^{\min} = C_i^{next} \\
 &\text{else } (\min error = \min error)
 \end{aligned}
 \tag{3.12}$$

where $E_{C_i^{next}}$ symbolizes the error calculated for i^{th} chromosome among the $n + 1$ chromosomes of C^{next} population and C^{\min} symbolizes the chromosome which has minimized error. Hence in order to accomplish the experiment and simulation for the training and testing of the performance for neural network, the following operators and parameters as shown in table 3 and 4 have been considered.

Table 3. Genetic operators used in the experiments

Training Algorithms	Genetic Operators Used
GA	Mutation with probability ≤ 0.1 and Crossover
Hybrid EA	Mutation with probability ≤ 0.1 and Crossover

Table 4. Parameters used for genetic algorithm and hybrid evolutionary algorithm

Parameter	Value
Adaptation Rate (K)	3.0
Learning Rate (η)	0.1
Momentum Descent Term (α)	0.9
Doug's Momentum Term $\left(\frac{1}{1-\alpha}\right)$	$\left(\frac{1}{1-\alpha}\right)$
Mutation Population Size	3

Parameter	Value
Crossover Population Size	2000
Minimum Error Exist in the Network (MAXE)	0.00001
Initial weights and biased term values	Randomly Generated Values Between 0 and 1

4. RESULTS AND DISCUSSIONS

Experimental results of the simulation as shown below are represented in four tables [table 5 to 8] and ten figures [5 to 10]. Tables [5 & 6] contain the simulation results for average number of iterations and number of convergence weight matrices performed by the hybrid evolutionary algorithm and random genetic algorithm respectively for the training pattern set of

straight alphabets and extended training pattern set of scaled & rotated alphabets. Tables [7 & 8] contain the entries for average number of iterations and number of convergence weight matrices performed by the hybrid evolutionary algorithm and random genetic algorithm respectively for the test pattern set of straight alphabets and scaled & rotated alphabets. In these tables [7 and 8] the number of correct respond is representing the number of times for the successful recognition of presented test pattern with respect to converged weight matrices during the training. Figures [5 to 10] show the comparison charts between the performance of hybrid evolutionary algorithm and random genetic algorithm during the training and recognition for the test pattern set. The performance evaluation based on these simulation results shows the better performance in terms of number of iterations and number of convergence weight matrices during the training and accuracy in terms of recognition for the presented test patterns of scaled & rotated form of patterns used during the training. The samples of straight alphabets from training set for scaling and rotation are considered on random bases for any arbitrary angle for the rotation and scaling parameters.

Table 5. Number of iterations and number of convergence weight matrices performed for training set of straight and scaled and rotated alphabets for hybrid evolutionary algorithm.

Sample (Straight Alphabets)	Average Iterations for Five Trails of Five Samples of Each Alphabets	Average number of convergence weight Vectors for Five Trails of Five Samples of Each Straight Alphabets	Sample	Scaling	Rotation	Average Iterations for five trials	Average convergence weight Vectors for five trials of Scaled & Rotated Alphabets
A	27	1262	A1	2 by 3	0	15	368
B	16	1479	B2	2 by 2	50	78	749
C	88	1548	C3	3 by 3	35	5	373
D	32	1734	D4	1 by 2	22	356	1840
E	20	1474	E2	2 by 2	65	1003	1778
F	14	1496	F2	2 by 1	45	252	1112
G	12	1434	G1	3 by 3	20	81	746
H	14	1737	H4	1 by 1	35	16	1113
I	31	1752	I1	3 by 2	25	26	1126
J	77	1538	J2	2 by 3	30	119	1091
K	56	1555	K3	2 by 3	20	4	381
L	20	1403	L4	2 by 2	35	80	1375

M	30	1260	M2	3 by 3	25	18	363
N	63	1337	N2	2 by 3	20	69	1091
O	4	1394	O1	2 by 2	25	14	385
P	12	1756	P4	2 by 1	65	30	1448
Q	10	1411	Q1	3 by 3	15	181	1107
R	4	1292	R2	2 by 2	25	57	1082
S	22	1601	S3	1 by 2	25	220	1479
T	17	1216	T4	2 by 3	30	52	730
U	38	1284	U2	2 by 3	20	79	1445
V	23	1648	V5	2 by 2	35	196	53.2
W	18	1410	W1	3 by 3	25	4	409
X	11	1358	X4	2 by 2	45	27	1391
Y	19	1297	Y5	3 by 3	15	6	1055
Z	9	1692	Z3	2 by 3	30	158	665

Table 6. Number of iterations and number of convergence weight matrices performed for training set of straight and scaled and rotated alphabets for random genetic algorithm.

Sample (Straight Alphabets)	Average Iterations for Five Trails of Five Samples of Each Alphabets	Average number of convergence weight vectors for Five Trails of Five Samples of Each Straight Alphabets	Sample	Scaling	Rotation	Average Iterations for five trials	Average convergence weight Vectors for five trials of Scaled & Rotated Alphabets
A	1552	848	A1	2 by 3	0	814	1027
B	610	1037	B2	2 by 2	50	159	521
C	678	916	C3	3 by 3	35	1132	940
D	400	981	D4	1 by 2	22	1077	854
E	1161	1193	E2	2 by 2	65	364	776
F	569	849	F2	2 by 1	45	126	794
G	67	910	G1	3 by 3	20	663	936
H	2482	811	H4	1 by 1	35	271	546
I	384	1046	I1	3 by 2	25	108	580

J	1365	981	J2	2 by 3	30	684	732
K	4053	1209	K3	2 by 3	20	426	1143
L	408	1047	L4	2 by 2	35	765	544
M	2171	850	M2	3 by 3	25	574	960
N	53	646	N2	2 by 3	20	76	575
O	19	929	O1	2 by 2	25	81	866
P	81	512	P4	2 by 1	65	391	762
Q	523	953	Q1	3 by 3	15	1185	864
R	397	1116	R2	2 by 2	25	1506	589
S	108	1186	S3	1 by 2	25	524	1055
T	6636	1067	T4	2 by 3	30	122	280
U	2612	890	U2	2 by 3	20	411	559
V	7048	1137	V5	2 by 2	35	2399	910
W	70	898	W1	3 by 3	25	81	655
X	9317	1018	X4	2 by 2	45	975	1013
Y	8385	997	Y5	3 by 3	15	461	278
Z	5298	1068	Z3	2 by 3	30	699	593

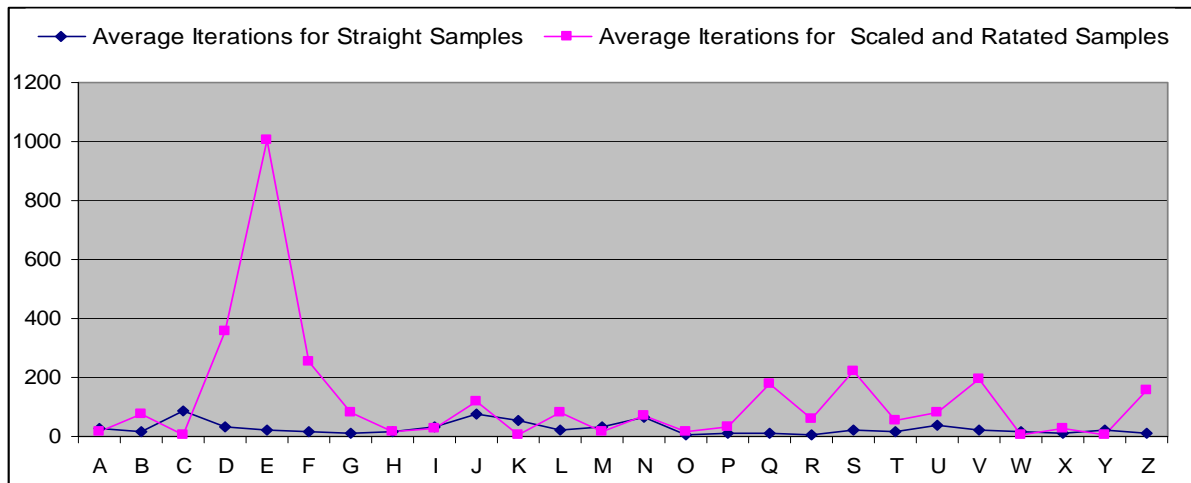


Figure 5. The comparison chart between the average numbers of iterations performed to train the network for straight and scaled & rotated alphabets by hybrid evolutionary algorithm

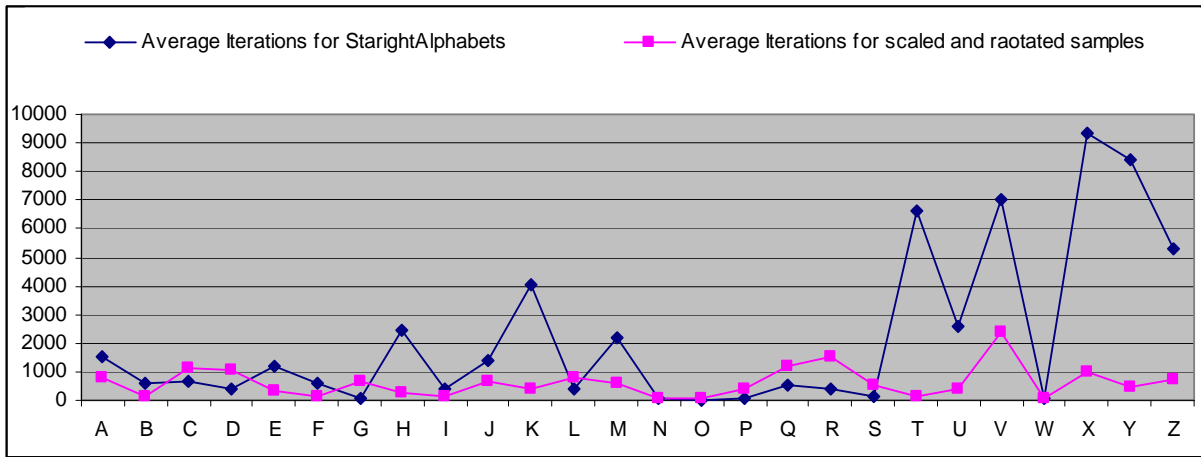


Figure 6. The comparison chart between the average numbers of iterations performed to train the network for the straight and scaled & rotated alphabets by hybrid evolutionary algorithm.

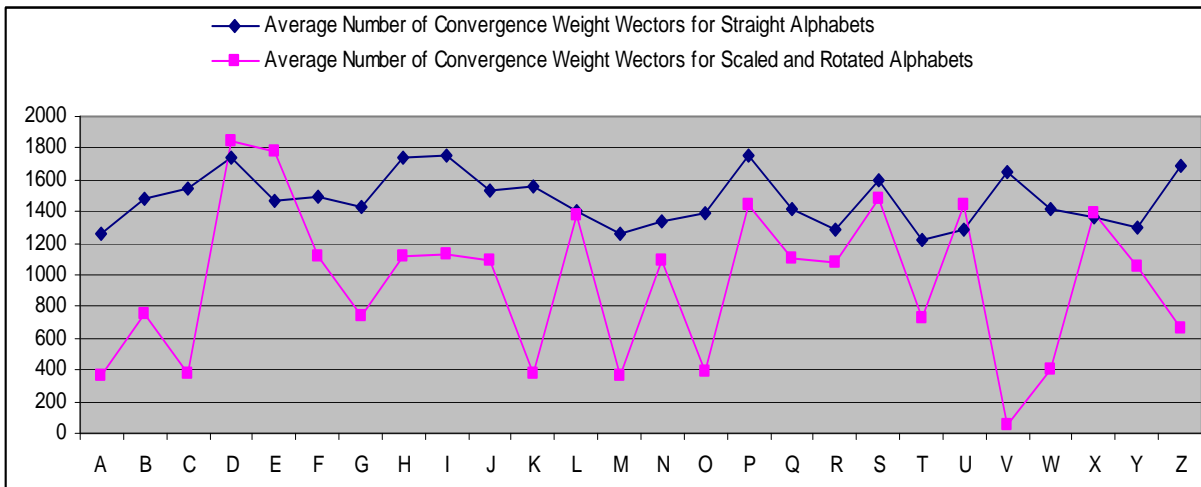


Figure 7. The comparison chart between the average numbers of convergence weight matrices obtained to train the network for straight and scaled & rotated alphabets by hybrid evolutionary algorithm.

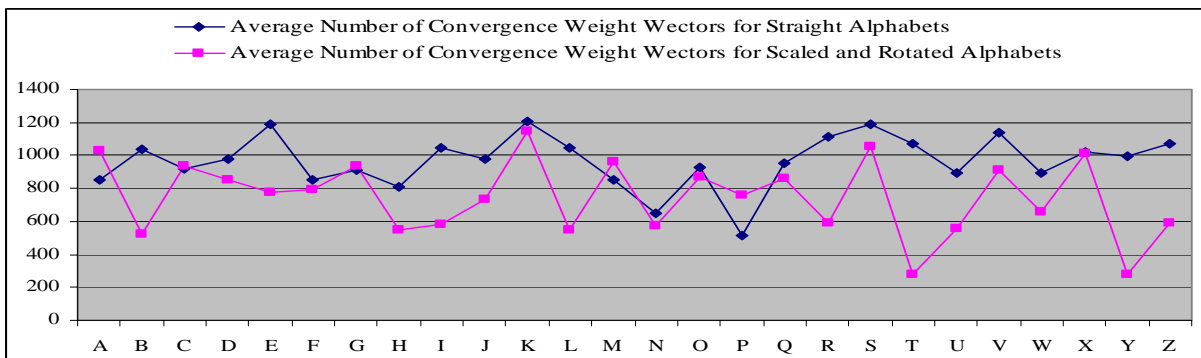


Figure 8. The comparison chart between the average numbers of convergence weight matrices obtained to train the network for straight and scaled & rotated alphabets by genetic algorithm.

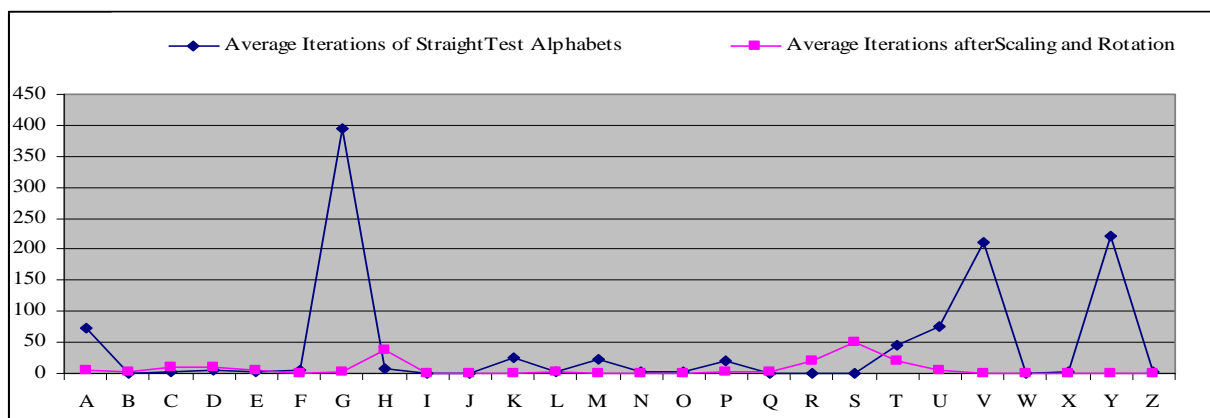


Figure 9. The comparison chart between numbers of time of correct respond for the presented test patterns with respect to number of convergence weight matrices to recognize the test pattern, of straight and scaled & rotated alphabets by hybrid evolutionary algorithm.

Table 7. Average number of correct responds for the number of convergence weight matrices for test pattern set of straight and scaled & rotated alphabets for hybrid evolutionary algorithm.

Alphabets	Average number of correct respond for Straight Test Alphabets	Average Number of Convergence Weight Matrices for the correct respond with Straight Test Alphabets	Scaling	Rotation	Average number of correct respond after Scaling and Rotation	Average Number of Convergence Weight Matrices for the correct respond with Scaling and Rotation
A	73	1121	2 by 1	45	4	4
B	1	1	2 by 3	25	2	2
C	2	27	2 by 2	36	9	9
D	4	2	3 by 3	45	9	9
E	2	57	3 by 3	55	6	6
F	4	10	2 by 2	35	1	1
G	395	1705	2 by 2	30	2	2
H	7	1	2 by 3	35	37	37
I	1	1	2 by 2	45	1	1
J	1	14	2 by 2	35	1	1
K	24	1904	2 by 2	30	1	1
L	2	1831	2 by 3	15	3	3
M	23	1849	2 by 2	20	1	1
N	2	8	3 by 3	25	1	1

Alphabets	Average number of correct respond for Straight Test Alphabets	Average Number of Convergence Weight Matrices for the correct respond with Straight Test Alphabets	Scaling	Rotation	Average number of correct respond after Scaling and Rotation	Average Number of Convergence Weight Matrices for the correct respond with Scaling and Rotation
O	2	1794	3 by 3	45	1	1
P	21	1828	2 by 2	35	2	2
Q	1	11	2 by 2	20	2	2
R	1	4	3 by 2	65	21	21
S	1	1790	2 by 2	45	51	51
T	45	1783	2 by 2	35	19	19
U	76	1748	2 by 1	25	6	6
V	212	1790	2 by 3	15	1	1
W	1	26	2 by 2	20	1	1
X	3	9	2 by 3	55	1	1
Y	222	1874	3 by 2	30	1	1
Z	3	1	2 by 2	45	1	1

Table 8. Average number of correct responds for the number of convergence weight matrices for test pattern set of straight and scaled & rotated alphabets for genetic algorithm.

Alphabets	Average number of correct respond for Straight Test Alphabets	Average Number of Convergence Weight Matrices for the correct respond with Straight Test Alphabets	Scaling	Rotation	Average number of correct respond after Scaling and Rotation	Average Number of Convergence Weight Matrices for the correct respond with Scaling and Rotation
A	68	2	2 by 1	45	320	1003
B	47	3	2 by 3	25	43	858
C	3207	1	2 by 2	36	831	848
D	189	4	3 by 3	45	4	804
E	240	5	3 by 3	55	4	1029
F	95	1	2 by 2	35	382	1027

Alphabets	Average number of correct respond for Straight Test Alphabets	Average Number of Convergence Weight Matrices for the correct respond with Straight Test Alphabets	Scaling	Rotation	Average number of correct respond after Scaling and Rotation	Average Number of Convergence Weight Matrices for the correct respond with Scaling and Rotation
G	113	1	2 by 2	30	1	604
H	72	10	2 by 3	35	15	596
I	2309	905	2 by 2	45	68	769
J	45	1	2 by 2	35	1	692
K	9	1	2 by 2	30	45	874
L	50	1	2 by 3	15	58	1031
M	120	1	2 by 2	20	12	683
N	32	6	3 by 3	25	24	745
O	24	2	3 by 3	45	143	644
P	34	2	2 by 2	35	1	713
Q	22	9	2 by 2	20	2	1163
R	588	1	3 by 2	65	232	873
S	29	5	2 by 2	45	89	1373
T	114	5	2 by 2	35	1	270
U	42	3	2 by 1	25	215	664
V	23	13	2 by 3	15	1	924
W	3	1	2 by 2	20	33	1168
X	159	1	2 by 3	55	64	1400
Y	180	11	3 by 2	30	13	1178
Z	387	1	2 by 2	45	283	1182

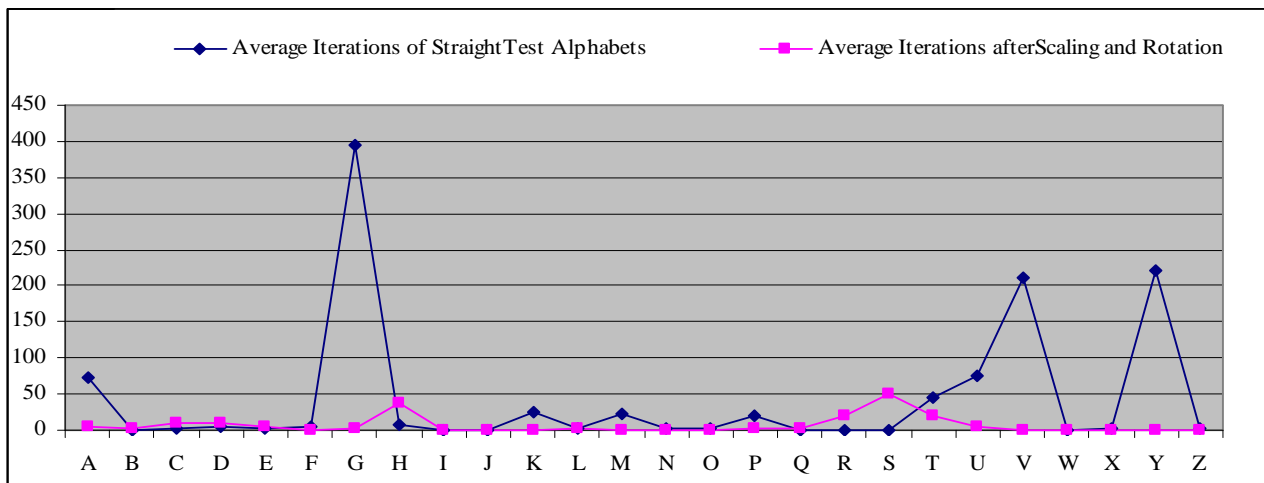


Figure 10. The comparison chart between numbers of time of correct respond for the presented test patterns with respect to number of convergence weight matrices to recognize the test pattern, of straight and scaled & rotated alphabets by random genetic algorithm.

5. CONCLUSION

The simulated results as presented in the form of tables and comparison charts mentioned above clearly show that the on-line training to recognize the scaled and rotated hand written English alphabets by genetic algorithm and hybrid evolutionary algorithm, provide a better scope in terms of convergence, epochs and more number of convergence weight matrices as the number of optimal solutions to solve these kind of problems. The recognition is done in a very few iterations through the trained network for straight hand written English alphabets, suggests the performance of online training is good enough. The following observations are drawn from the simulated results:

1. Training of the scaled and rotated patterns have taken less number of iterations and obtained higher number of weight matrices in comparison to training of initial network for the given training set of straight patterns as shown in Figure [5 to 10] for both hybrid evolutionary algorithm and genetic algorithm. It shows that the already trained network for the given training set is trained with more accuracy and speed. Thus, it shows that the network can easily adapt the new similar patterns or the extended training set.
2. The recognition of test pattern set of scaled and rotated alphabets has also considered with respect to the number of converged weight matrices during the training for extended training set. The numbers of times the test pattern is recognized as shown in table 7 and 8 reflects that there are as many optimal solutions for the presented test pattern. Thus, there are more than one optimal solutions are obtained during the training and testing. Hence the given neural network architecture is performing in good generalization and approximation manner.
3. The comparison of performance evaluation in between hybrid evolutionary algorithm and the genetic algorithm as presented in the simulation results exhibited that the performance of hybrid evolutionary algorithm is far better in comparison to the genetic algorithm in terms of

accuracy in recognition, number of iterations for training and convergence weight matrices for number of optimal solutions.

4. The successful implementation of feed forward neural networks with evolutionary algorithms for hand written English (straight and scaled and /or rotated) alphabets, motivated to apply the same for other complex problems of pattern recognition. It is expected to extend the work in future for the following problems:
 - (a) Recognition of overlapped alphabets.
 - (b) Refinement of training of the incomplete training set due to wage ness, fuzzy ness, or some other complexity in the samples.

6. REFERENCES

- [1] Impedovo, S. (editor), "Fundamentals in Handwriting Recognition." ,NATO-Advanced Study Institute Series F. Springer-Verlag, (1994).
- [2] Mori, S., Suen, C. Y., and Yamamoto, K., "Historical review of OCR research and development". *Proceedings of the IEEE* (Special Issue on Optical Character Recognition.), vol. 80(7), pp. 1029--1058, (1992).
- [3] Fukushima, K and Wake, N., "Handwritten alphanumeric character recognition by the neocognitron", *IEEE Trans. on Neural Networks*, vol. 2(3), pp. 355-365, (1991).
- [4] Blackwell, K. T., Vogl, T. P. and Hyman, S. D., Barbour, G. S., and Allcon, D. L., "A new approach to handwritten character recognition", *Pattern Recognition*, vol. 25(6), pp. 655-666, (1992).
- [5] Le Cun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbad, W. and Jackel, L. D., "Handwritten digit recognition with a backpropagation network", *Neural Information Processing Systems*, Touretzky editor, vol. 2, pp. 396-404, Morgan Kaufmann Publishers, (1990).
- [6] Ahmed, M. and Ward, R., "An Expert System for General Symbol Recognition.", *Pattern Recognition*, vol. 33(12), pp. 1975-1988, (2000).

- [7] Amin, A., "Recognition of Hand-Printed Latin Characters Based on Generalized Hough Transform and Decision Tree Learning Techniques.", International Journal of Pattern Recognition and Artificial Intelligence, vol. 14(3), pp. 369-387, (2000).
- [8] Hailong, Liu and Xiaoqing, D., "Handwritten character recognition using gradient feature and quadratic classifier with multiple discrimination schemes", Proceedings, Eighth International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp.: 19 - 23 (2005).
- [9] Kharma, N., and Ward, R. , "A Novel Invariant Mapping Applied to Handwritten Arabic Character Recognition", Pattern Recognition, vol. 34(11), August, (2001).
- [10] Yegnarayana, B., "Artificial Neural Networks", Prentice Hall of India, (2004).
- [11] Shrivastava, S. and Singh, M. P., "Analysis of Soft Computing Techniques for Minmizing the Problem of Local Minima in Back Propagation for Hand written English Alphabets.", Proceedings, International Conference on Soft Computing and Intelligent Systems, vol. 2, pp. 307-313,(2007)
- [12] "Transformation matrix" - Wikipedia, the free encyclopedia.htm
- [13] "Scaling (geometry)" - Wikipedia, the free encyclopedia.htm
- [14] "Rotation (mathematics)" - Wikipedia, the free encyclopedia.htm
- [15] Annadurai, S. and Shanmugalakshmi, R., "Fundamental of Digital Image Processing.", Pearson Education, ISBN 81-7758-479-0,(2007)
- [16] Vince, J.A., "Mathematics for Computer Graphics.", Springer, ISBN 10:1-84-628-034-6,(2006)
- [17] Shrivastava, S., Singh, M. P., "Performance evaluation of feed-forward neural network with soft computing techniques for hand written English alphabets", Applied Soft Computing, Elsevier, 11(2011), 1156-1182.

Author Biographies



S. R. Pande is currently Associate Professor of Computer Science and Head, Department of Computer Science, S.S.E.S. Amt's, Science College, Nagpur, Maharashtra state, India. He has about 19 years of experience in teaching and research. He has published several papers in national and international journal and conferences. He is currently a Ph.D. candidate in the Department of Electronics and Computer Science, R.T.M. Nagpur University, Nagpur, M.S., India. His areas of interest are Data structure, Discrete mathematics, Theory of Computations, Neural Network, Fuzzy logic, Computer Graphics, Image Processing, Relational DBMS.



Manu Pratap Singh, received his Ph.D. in computational physics from Kumaun Univesity, Nanital, UA, India, in 2001. He is currently Associate Professor in Department . of Computer, Institute of Computer& Information Science, Dr. B. R. Ambedkar University, Khandari, Agra. He has about 13 years of experience in teaching and research. His research interests are focused on neuro-computing, neuroscience, neuro-informatics, soft-computing, etc. Dr. Singh is a member of technical committee of IASTED, Canada since 2004. He is being referee for various international journals. In 2005, he received young scientist award from International Academy of Physical Sciences, Allahabad, UP, India. He has also gotten more than 60 publications in various international and national journals.

Saurabh Srivastava is currently Assistant Professor of Computer Science in Department of Mathematical Sciences and Computer Applications Bundelkhand University, Jhansi, Uttar Pradesh India. He has about 11 years of experience in teaching and research. He has published several papers in national and international journal and conferences. He was awarded with Ph.D. degree in the areas of Pattern Recognition using soft computing techniques in the year 2009. His research interest are Fuzzy Systems, Neural Network, Pattern Recognition and their applications.areas.

Rajesh Lavania is Assistant Professor of Computer Science in Department of Computer Science&Engineering, Institute of Engg. & Tech ., Dr. B. R. Ambedkar University, Khandari, Agra Uttar Pradesh India. He has about 8 years of experience in teaching and research. He has published several papers in national and international journal and conferences. He is currently a Ph.D. candidate in the Department of Computer, Institute of Computer& Information Science, Dr. B. R. Ambedkar University, Khandari, Agra, India