Hand Written English Character Recognition using Column-wise Segmentation of Image Matrix (CSIM)

RAKESH KUMAR MANDAL
N R MANNA
Department of Computer Science & Application
University of North Bengal
PO: NBU, Distt: Darjeeling
West Bengal-734013
INDIA
rakesh_it2002@yahoo.com, nrmanna@sify.com http://www.nbu.ac.in

Abstract: - Research is going on to develop both hardware and software to recognize handwritten characters easily and accurately. Artificial Neural Network (ANN) is a very efficient method for recognizing handwritten characters. Attempts have already been made to recognize English alphabets using similar type of methods. A new method has been tried, in this paper, to improve the performance of the previously applied methods. The input image matrix is compressed into a lower dimension matrix in order to reduce non significant elements of the image matrix. The compressed matrix is segmented column-wise. Each column of a particular image matrix is mapped to identical patterns for recognizing a particular character. Majority of a known pattern decides the existence of a particular character.

Key-Words: - ANN, CSIM, Compression, Perceptron, Segmentation, Learning Rule

1 Introduction

Expert systems can be developed using Artificial Neural Network (ANN) which can recognize hand written English alphabets easily and accurately. The handwriting styles of different individuals vary infinitely which makes the development of expert systems to recognize handwritten characters very difficult. Common features extracted out from different handwriting styles have proved to be a very successful method. Generalized problems can be solved using Multiscale Training Technique (MST), [1, 2, 3].

One such method has been applied in Devnagri Script where some feature extraction techniques like intersection, shadow features, chain code histogram and straight line fitting were used, [4].

One feature extraction method is finding out twelve directional feature inputs which depend upon the gradients. The features of the characters are directions of the pixels w.r.t. neighbouring pixels, [5].

Hand written English character recognition using Row-wise Segmentation Technique (RST) is an approach to find out common features of same characters written in different hand writing styles by segmenting the input pattern matrix into separate rows and trying to find out common rows among different hand writing styles, [6].

In order to find out common features among the characters written by different individuals some methods are added to the already developed ANNs like perceptron learning method, [7, 8].

In this paper an attempt has been made through the compressed Column-wise Segmentation of Image Matrix (CSIM) to recognize hand written characters using Neural Network.

The overall program is divided into three parts, compression of the image matrix, segmenting the compressed matrix column-wise and training the net and finally testing the net by providing characters taken from different individuals.

2 Methodology

A scanner is used to scan the handwritten English alphabets written on a piece of A4 size paper. Each character is enclosed in a box of size 80 x 80 in jpeg format. The image is converted into a binary matrix of size 80 x 80. Each matrix contains a large number of elements which conveys no information about the pattern. In order to eliminate such elements 80 x 80 matrix is compressed into a matrix of size 10 x 10. The 10 x 10 matrix is segmented column-wise into 10 segments of size 10 each. All the columns of a particular character are mapped into identical patterns used to recognize that particular character. In this way standard weight matrix is obtained for each character. Test characters are presented to the trained net. A counter is used to find out the number of identical patterns produced by the test character. Majority of the identical pattern present in a test character decides the existence of a particular character.

2.1 Compression of 80 x 80 matrix into 10 x 10 matrix

A matrix can be compressed into a matrix of lower dimension in order to reduce the non significant elements of the matrix. Matrix A can be compressed into matrix A_COM by using the function Φ.

Matrix A can be split into n uniform blocks. The dimension of each block A_BLOCK$_i$ is m x m.

2.1.1 Algorithm Compression

Step1. Input Matrix A.

Step2. Split A into n uniform blocks, where A_BLOCK$_i$ is the ith block of A.

Step3. for i=1 to n,

A_ COM$_i$= Φ(A_BLOCK$_i$)

End,

where A_ COM$_i$ is the ith element of the compressed matrix A_COM and

Φ(A_BLOCK$_i$) = 1, if there exists at least one element in A_BLOCK$_i$, 1

Φ(A_BLOCK$_i$) = 0, if all the elements of A_BLOCK$_i$ are 0

Step 4. Stop

Example 1. Given matrix A of size 4 x 4

$$A = \begin{matrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{matrix}$$

A can be split into 4 uniform blocks of dimension 2 x 2 each.

$$A\_BLOCK_1 = \begin{matrix} 1 & 1 \\ 1 & 0 \end{matrix}$$

$$A\_BLOCK_2 = \begin{matrix} 0 & 0 \\ 0 & 0 \end{matrix}$$

$$A\_BLOCK_3 = \begin{matrix} 0 & 1 \\ 0 & 0 \end{matrix}$$

$$A\_BLOCK_4 = \begin{matrix} 1 & 0 \\ 0 & 0 \end{matrix}$$

Using algorithm Compression

$$A\_COM = \begin{matrix} 1 & 0 \\ 1 & 1 \end{matrix}$$

In the same way 80 x 80 binary matrix is compressed into a 10 x 10 binary matrix considering 100 uniform blocks of dimension 8 x 8 each.

2.2 Column-wise Segmentation of 10 x 10 matrix

The 10 x 10 matrix is segmented column-wise into 10 columns of size 10 each. Each column is mapped into 10 identical patterns used to recognize a character, (Figure 1). Weight matrix is initialized to zero. Perceptron learning rule is used to train the net.

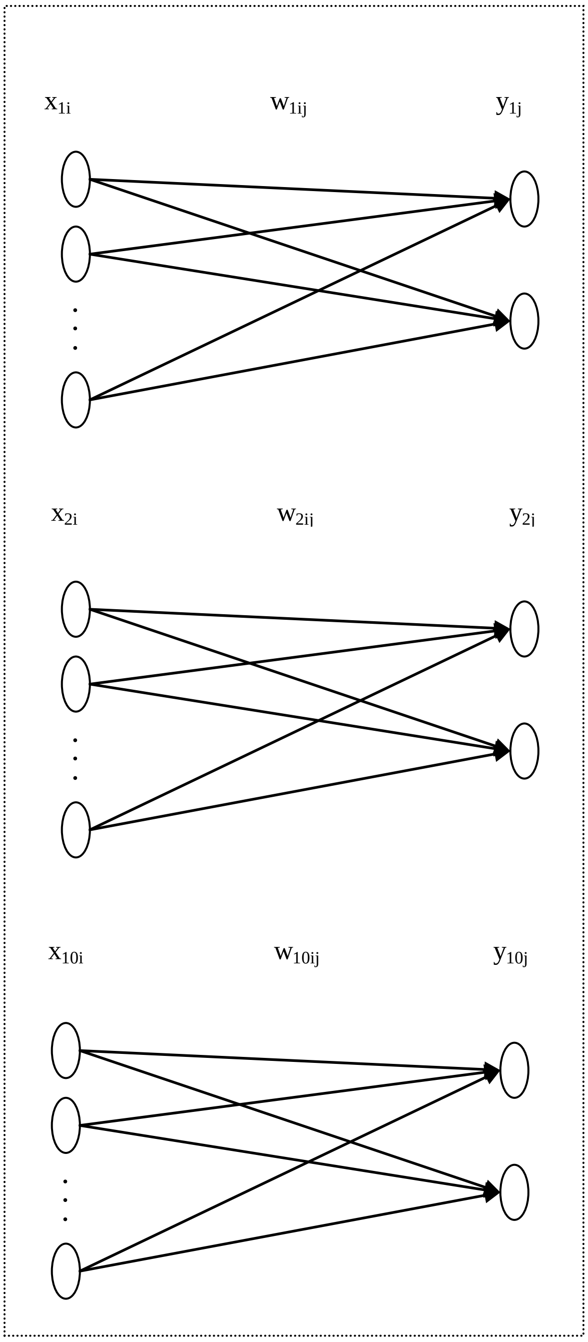

Figure 1. CSIM Neural Net, each block contains 10 inputs and 2 outputs

2.2.1 Perceptron Learning Rule

In a simple peceptron one or more than one neurons are connected to a single neuron with the help of weights, (Figure 2). Weights may be initialized to 0 or very small values. An iterative procedure is used to find out the standard weights [7, 8]. At standard weights the net generates correct outputs. The output of the jth perceptron is y_j. Output is calculated by applying the activation function f, which is a function of the net output, y_out. Thus, y_j=f(y_out$_j$)

and

$$y\_out_j = \sum_i x_i w_{ij} \qquad (1)$$

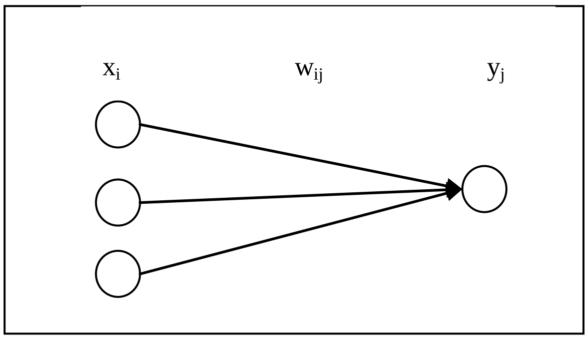

Figure 2. A simple perrceptron having three inputs and one output

Where, x_i is the ith element of input vector and w_{ij} is the weight between the ith element of the input vector and jth element of the output vector. The function f(y_out$_j$) takes the following values depending upon the values of y_out$_j$.

$$f(y\_out_j) = \begin{cases} 1 \text{ if } y\_out_j > \theta \\ 0 \text{ if } -\theta <= y\_out_j <= \theta \\ -1 \text{ if } y\_out_j < -\theta \end{cases}$$

Here, θ is the threshold value, taken at random. For each training input, the net would calculate the response of the output unit.

The net would determine whether an error occurred for this pattern by comparing the calculated output with the target value. If an error occurs for a particular training input pattern, the weights would be changed according to the formula:

$$w_{ij}(new) = w_{ij}(old) + \alpha t_j x_i \qquad (2)$$

α is the learning rate, the value of which is taken at random, t_j is the target value, i.e. the output expected from the net. The output y_j produced by the net is compared with t_j and the difference leads to the modification in weight given by equation (2). The process continues until y_j becomes equal to t_j. Weights obtained at that point are the final and standard weight.

2.3 Training of the net to obtain the weight matrix

The training started with an input vector of size [80 x 80 = 6400], compressed into a vector of size [10 x 10 = 100], (Figure 3).

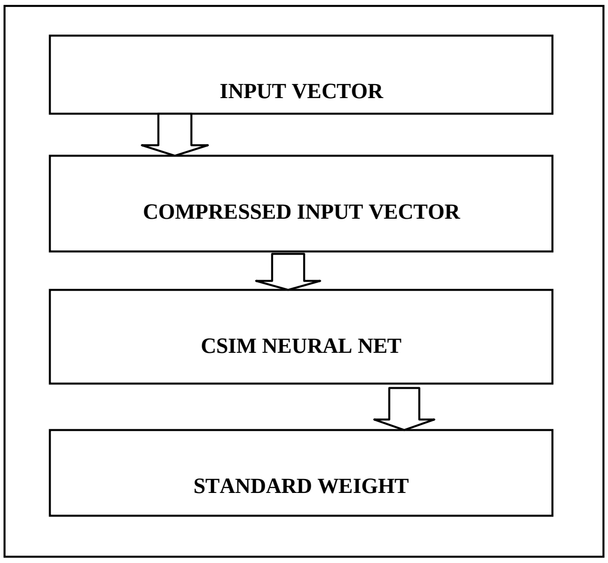

Figure 3. Conceptual Diagram of CSIM training

First four characters of the English alphabets are considered for training, (Figure 4).The weights are initially set to zero. The net, (Figure 1) is trained using Perceptron Learning Rule. After training the net for few epochs, the final weight that generates the correct output is obtained. The final weight obtained remains the same for the next few epochs and is considered as the standard weight for testing.

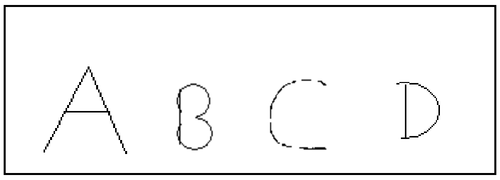

Figure 4. Alphabets considered for CSIM training

2.4 Testing of the net

The net is tested with first four characters of English alphabets, deviated to the extent, at which the deviated character can be identified, (Figure 5). The testing starts with a [80 x 80 = 6400] vector compressed into a [10 x 10 =100] vector on the net as shown in Figure 1.

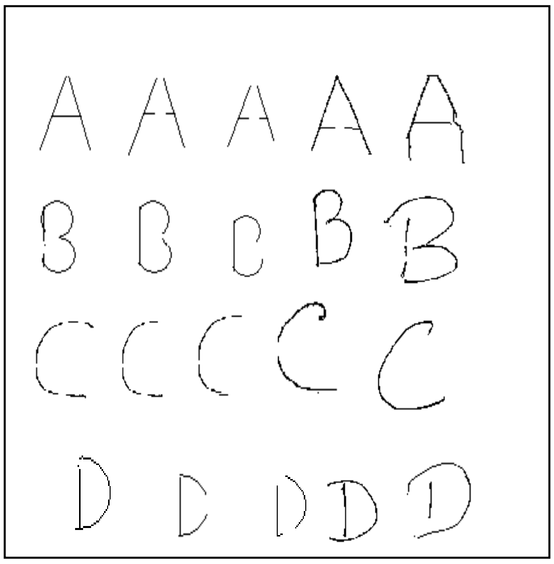

Figure 5. Alphabets considered for CSIM testing

The compressed vector is presented to the net having the standard weight matrix. Since, there are two outputs, input patterns are mapped as either [-1, -1], [-1, 1], [1, -1] or [1,1] to identify the characters A, B, C or D respectively. A counter is used to count the number of identical pattern outputs from the ten different segments for confirmation. A value greater than or equal to six is considered for the identification of a particular character, (Figure 6).

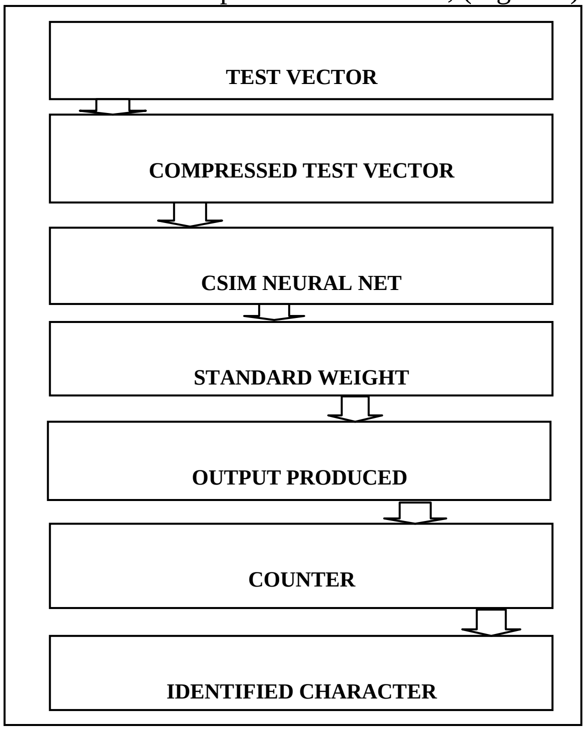

Figure 6. Conceptual Diagram of CSIM testing

3 Algorithm CSIM

Step 1. Scan, N number of characters to be trained and convert each into a binary matrix of dimension m x m.

Step 2. Apply algorithm compression on each m x m matrix and compress into n x n matrix, where m>n.

Step 3. Store each compressed matrix in column major form and assign one by one to the input vector x.

Step 4. Divide x into M number of segments, where each segment contains n number of elements.

Step 5. Consider M number of groups in the output layer, y. Each group contains p number of elements, where

p=2, if 1<N<=4

p=3, if 4<N<=8

p=4, if 8<N<=16 and so on.

Step 6. Completely interconnect all the neurons of each M segment in the input layer to all the neurons of each corresponding M group in the output layer.

Step 7. For 1 to N number of alphabets to be trained, consider the target vector in the following way:

For example,

A=00

B=01

C=10

D=11 and so on.

Step 8. Initialize weight matrix

```
    For k=1 to M
        For i=1 to n
            For j=1 to p
                $W_{kij}=0$
            End p
        End n
    End M
```

Step 9. Apply Perceptron Learning Algorithm to each segment and find out the standard weights.

Step 10. Scan the sample character to be tested and convert into binary matrix of dimension m x m.

Step 11. Apply algorithm Compression on the sample m x m matrix and compress into n x n matrix, where m>n.

Step 12. Store the compressed matrix in column major form and assign one by one to the input vector x.

Step 13. Present the test vector x into the CSIM net.

Step 14. Apply counter to the output produced, to identify the character.

The following example demonstrates the applicability of the CSIM algorithm. It can be observed that the characters when compressed using the compression algorithm holds the pattern identified by the naked eye.

Example 2. Let us consider four characters of English alphabet A, B, C and D.

A can be represented by a 10 x 10 matrix shown below.

					*				
				*		*			
			*				*		
		*						*	
	*								*
*									*
*	*	*	*	*	*	*	*	*	*
*									*
*									*
*									*

In the above matrix the blank space can be considered as -1 and * can be considered as 1. So, the vector representation of the matrix A in column major form is represented as:

A={-1,-1,-1,-1,-1,1,1,1,1,1,-1,-1,-1,-1,1,-1,1,-1,-1,-1,-1,-1,1,-1,-1,1,1,1,1,-1,-1,1,-1,-1,-1,1,-1,-1,-1,-1,1,1,-1,-1,-1,-1,1,-1,-1,-1,1,-1,-1,-1,-1,-1,1,-1,-1,-1,-1,1,1,-1,-1,-1,1,-1,-1,-1,-1,-1,1,-1,-1,-1,1,-1,-1,-1,-1,-1,1,-1,-1,1,-1,-1,-1,-1,-1,-1,-1,1,1,1,1,1,1,1}

A can be compressed to form a matrix of dimension 5 x 5 by applying compression algorithm.

		*	*	
	*		*	*
*	*	*	*	*
*				*
*				*

A_COM={-1,-1,1,1,1,-1,1,1,-1,-1,1,-1,1,-1,-1,1,1,1,-1,-1,-1,1,1,1,1}

Similarly, compression algorithm is applied on B, C and D to form B_COM, C_COM and D_COM.

Matrix representation of character B is given below.

*	*	*	*	*	*	*	*	*	
*									*
*									*
*									*
*									*
*	*	*	*	*	*	*	*	*	
*								*	
*									*
*									*
*	*	*	*	*	*	*	*	*	

B={1,1,1,1,1,1,1,1,1,1, 1,-1,-1,-1,-1,1,-1,-1,-1,1, 1,-1,-1,-1,-1,1,-1,-1,-1,1,1,-1,-1,-1,-1,1,-1,-1,-1,1,1,-1,-1,-1,-1,1,-1,-1,-1,1,1,-1,-1,-1,-1,1,-1,-1,-1,1,1,-1,-1,-1,-1,1,-1,-1,-1,1,1,-1,-1,-1,-1,1,-1,-1,-1,1,1,-1,-1,-1,-1,1,-1,-1,-1,1, 1,-1,-1,-1,-1,1,1,-1,-1,1,1, -1,1,1,1,1,-1,-1,1,1,-1}

Matrix representation of character B_COM is given below.

*	*	*	*	*
*				*
*	*	*	*	*
*				*
*	*	*	*	*

B_COM={1,1,1,1,1, 1,-1,1,-1,1, 1,-1,1,-1,1, 1,-1,1,-1,1, 1,1,1,1,1}

Matrix representation of character C is given below.

	*	*	*	*	*	*	*	*	*
*									
*									
*									
*									
*									
*									
*									
*									
	*	*	*	*	*	*	*	*	*

C={-1,1,1,1,1,1,1,1,1,-1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,1,1}

Matrix representation of character C_COM is given below.

*	*	*	*	*
*				
*				
*				
*	*	*	*	*

C_COM={1,1,1,1,1,1,-1,-1,-1,1, 1,-1,-1,-1,1, 1,-1,-1,-1,1, 1,-1,-1,-1,1}

Matrix representation of character D is given below.

*	*	*	*	*	*	*			
*							*		
*								*	
*									*
*									*
*									*
*									*
*								*	
*							*		
*	*	*	*	*	*	*			

D={1,1,1,1,1,1,1,1,1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, 1,-1,-1,-1,-1,-1,-1,-1,-1,1, -1,1,-1,-1,-1,-1,-1,-1,1,-1, -1,-1,1,-1,-1,-1,-1,1,-1,-1, -1,-1,-1,1,1,1,1,-1,-1,-1}

Matrix representation of character C_COM is given below.

D_COM={1,1,1,1,1, 1,-1,-1,-1,1, 1,-1,-1,-1,1, 1,-1,-1,-1,1, -1,1,1,1,-1}

*	*	*	*	
*				*
*				*
*				*
*	*	*	*	

D_COM={1,1,1,1,1, 1,-1,-1,-1,1, 1,-1,-1,-1,1, 1,-1,-1,-1,1, -1,1,1,1,-1}

The training of the net will be compleneted in the following epochs.

Epoch 1

Table 2. Segmented inputs of four characters

	x1	x2	x3	x4	x5
A_C	-1, -1, 1, 1, 1	-1, 1, 1, -1, -1	1, -1, 1,- 1, -1	1, 1, 1, -1, -1	-1, 1, 1, 1, 1
B_C	1, 1, 1, 1, 1	1, -1, 1, -1, 1	1, -1, 1, -1, 1	1, -1, 1, - 1, 1	1, 1, 1, 1, 1
C_C	1, 1, 1, 1, 1	1 ,-1, -1, -1, 1	1, -1, -1, -1, 1	1, -1, -1, -1, 1	1, -1 ,-1, -1, 1
D_C	1, 1, 1, 1, 1	1, -1 ,-1, -1, 1	1, -1, -1, -1, 1	1, -1, -1, -1, 1	-1, 1, 1, 1, -1

The segmented inputs of four characters are presented to the net one by one for training. The initial weight matrix is set to zero as the weights are not known in advance. The target matrix is known in advance. The learning used here is supervised in nature. The target matrix depicts the expected result. Taking w_{ij}=0, initially the net output as well as the activations are calculated. The net output and the activations calculated after first epoch are given in the following tables. It can be observed that the expected results are not obtained after the first epoch.

Table 3. Initial Weight Matrix

	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg1	0	0	0	0	0	0	0	0	0	0
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg2	0	0	0	0	0	0	0	0	0	0
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg3	0	0	0	0	0	0	0	0	0	0
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg4	0	0	0	0	0	0	0	0	0	0
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg5	0	0	0	0	0	0	0	0	0	0

Table 4. Net Output after epoch1

	Y_IN11	Y_IN 21	Y_IN 31	Y_IN 41	Y_IN 51
A_C	0	0	0	0	0
B_C	-3	1	-3	-1	-3
C_C	-3	2	-1	1	3
D_C	4	2	6	3	-1
	Y_IN12	Y_IN 22	Y_IN 32	Y_IN 42	Y_IN 52
A_C	0	0	0	0	0
B_C	-3	1	-3	-1	-3
C_C	0	3	0	2	0
D_C	-1	0	-1	1	5

The epochs are repeated until the expected results are obtained. In this example it is found that the correct results are obtained in three epochs. The following condition is satisfied at epoch 3.

$y_j = t_j$

Table 5. Activation after epoch1

	Y11	Y 21	Y 31	Y 41	Y 51
A_C	0	0	0	0	0
B_C	-1	1	-1	-1	-1
C_C	-1	1	-1	1	1
D_C	1	1	1	3	-1
	Y12	Y 22	Y 32	Y 42	Y 52
A_C	0	0	0	0	0
B_C	-1	1	-1	-1	-1
C_C	0	1	0	1	0
D_C	-1	0	-1	1	1

Epoch 2

The activation obtained after epoch1 are not equal to the target. So, the net is trained for the next epoch.

Table 6. Weight matrix obtained after epoch2

	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg1	0	0	2	2	0	0	0	0	0	0
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg2	0	1	0	-1	-2	-1	2	1	0	1
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg3	0	0	0	0	-2	0	0	0	2	2
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg4	-1	-1	-1	-1	-1	1	1	1	1	1
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg5	0	1	0	1	0	1	0	1	-2	-1

Table 7. Net Output after epoch2

	Y_IN11	Y_IN 21	Y_IN 31	Y_IN 41	Y_IN 51
A_C	-4	-6	-6	-7	-6
B_C	0	-6	-2	-1	-2
C_C	1	2	6	1	-2
D_C	-1	2	6	3	1
	Y_IN12	Y_IN 22	Y_IN 32	Y_IN 42	Y_IN 52
A_C	-6	-9	-6	-5	-1
B_C	6	3	6	5	7
C_C	-2	1	-2	-5	-7
D_C	6	2	6	1	5

Table 8. Activation after epoch2

	Y11	Y 21	Y 31	Y 41	Y 51
A_C	-1	-1	-1	-1	-1
B_C	0	-1	-1	-1	-1
C_C	1	1	1	1	-1
D_C	-1	1	1	1	1
	Y12	Y 22	Y 32	Y 42	Y 52
A_C	-1	-1	-1	-1	-1
B_C	1	1	1	1	1
C_C	-1	1	-1	-1	-1
D_C	1	1	1	1	1

Epoch3 produces the final standard weight matrix for training.

<u>Epoch3</u>

Table 9. Weight matrix obtained after epoch3

	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg1	0	0	2	2	0	0	0	0	0	0
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg2	0	0	0	0	-2	0	2	2	0	0
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg3	0	0	0	0	-2	0	0	0	2	2
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg4	-1	-1	-1	-1	-1	1	1	1	1	1
	w11	w12	w21	w22	w31	w32	w41	w42	w51	w52
Seg5	1	1	-1	1	-1	1	-1	1	-1	-1

Table 10. Net Output after epoch3

	Y_IN11	Y_IN 21	Y_IN 31	Y_IN 41	Y_IN 51
A_C	-6	-6	-6	-7	-9
B_C	-2	-6	-2	-1	-5
C_C	6	2	6	1	5
D_C	6	2	6	3	1
	Y_IN12	Y_IN 22	Y_IN 32	Y_IN 42	Y_IN 52
A_C	-6	-6	-6	-5	-1
B_C	6	2	6	5	7
C_C	-2	-6	-2	-5	-7
D_C	6	2	6	1	5

Table 11. Activation after epoch3

	Y11	Y 21	Y 31	Y 41	Y 51
A_C	-1	-1	-1	-1	-1
B_C	-1	-1	-1	-1	-1
C_C	1	1	1	1	1
D_C	1	1	1	1	1
	Y12	Y 22	Y 32	Y 42	Y 52
A_C	-1	-1	-1	-1	-1
B_C	1	1	1	1	1
C_C	-1	-1	-1	-1	-1
D_C	1	1	1	1	1

4 Result Analysis

Scilab was used to test the accuracy of the net. Four characters A, B, C and D are taken initially for testing. The response of the experiment shows very good results for the first four alphabets. It was found that the neural network identifies each sample up to 40% distortion of the test sample from the standard character.

Neural Network parameters used in the experiment

Number of input neurons =6400

Number compressed input neurons=100

Number of neurons in each input pattern segment=10

Number of input pattern segments=10

Number of Neurons in each output group = 2

Dimension of weight matrix for each segment = 10 x 2

Training Algorithm used = Perceptron

Number of Epoch = 3, $\theta = 0.2$, $\alpha = 1$

t_j = {A(-1,-1), B(-1,1),C(1,-1), D(1,1)}

Table 1, shows the results produced by the neural net when characters were presented to the net with different levels of distortions from the standard ones. NI stands for not identified.

Table 12. Identification of the deviated characters

	Characters presented with different levels of distortion				
	10%	**20%**	**30%**	**40%**	**50%**
A	Identified	Identified	Identified	Identified	Identified
B	Identified	Identified	Identified	Identified	Identified
C	Identified	Identified	Identified	Identified	Identified
D	Identified	Identified	Identified	Identified	NI

5 Discussion

The results show that CSIM training, allows very fast convergence. The number of epochs required for the training is very less. Accuracy of identifying the characters is also very good. Number of samples identified with maximum distortion from the standard ones is satisfactory. The compression of the input matrix helps to improve the performance of CSIM net to a great extent. It was also found that the method of compression applied is able to hold the information in the compressed matrix with great accuracy. However, the method is tested for a few sample characters.

6 Conclusion

CSIM is a better approach as compared to the previously tried methods of segmentation. Compression of the input matrix helped a lot to get rid of the unused elements of the input matrix. Removal of unused elements makes the net small, simple and fast. Column-wise segmentation is a better approach than the row-wise segmentation as segmentation of the matrix column-wise produces more variation in the patterns and helps to obtain a sharp eye in the identification of the pattern. Experiments will be carried out to refine the already developed methods in order to enhance the performance and accuracy of the net.

7 Acknowledgements

Thanks to all my colleagues who contributed their moral support in the development of this paper.

References:

[1] Robinson, G. (1995), The Multiscale Technique, Available: http://www.netlib.org/utk/Isi/pcwLSI/text/node123.html.

[2] Handwritten Character Recognition, Available: http://tcts.fpms.ac.be/rdf/hcrinuk.htm.

[3] Velappa Ganapathy, and kok Leong Liew, Handwritten Character Recognition Using Multiscale Neural Network Training Technique, World Academy of Science, Engineering and Technology 39 2008.

[4] Sandhya Arora, Debotosh Bhattacharjee, Mita Nasipuri, Dipak kumar Basu and Mahantapas Kundu, Combining Multiple Feature Extraction Techniques for Handwritten Devnagri Character Recognition, Available: http://arxiv.org/ftp/arxiv/papers/1005/1005.4032.pdf, (Accessed: 26, July, 2010).

[5] Dayashankar Singh, Sanjay Kr. Singh and Dr. (Mrs.) Maitreyee Dutta, Hand Written Character Recognition Using Twelve Directional Feature Input and Neural Network, Available: http://www.ijcaonline.org/journal/number3/pxc387173.pdf, (Accessed: 26, July, 2010).

[6] Rakesh Kumar Mandal and N R Manna, Hand Written English Character Recognition using Row-wise Segmentation Technique (RST), International Symposium on Devices MEMS, Intelligent Systems & Communication (ISDMISC) 2011, Proceedings published by International Journal of Computer Applications (IJCA).

[7] Neural Networks, G.N. Swamy, G. Vijay Kumar, SCITECH.

[8] Fundamentals of Neural Networks, Architectures, Algorithms and Applications, Laurene Fausett, Pearson Education.