AMSE JOURNALS –2014-Series: Advances B; Vol. 57; N° 1; pp 88-107
Submitted Dec. 2013; Revised July 7, 2014; Accepted July 20, 2014

Handwritten English Character Recognition using Hoof Segmentation of Image Matrix (HSIM)

* R.K. Mandal, **N R Manna

*Department of Computer Science & Application, University of North Bengal,
Siliguri, Distt : Darjeeling, West Bengal-734013, India (rakesh_it2002@yahoo.com)
** Department of Computer Science & Application, University of North Bengal,
Siliguri, Distt : Darjeeling, West Bengal-734013, India (nrmanna@sify.com)

Abstract

This paper presents an approach to simplify the feature extraction method to identify varying character patterns. Segments can be formed in different shapes and presented to the Artificial Neural Network (ANN) for training and testing purpose. Segments are identified from different characters in the form of horse hoof like structures and presented to a multiple layer ANN for training and testing purpose.

Keywords

Neural Networks, Hoof Segmentation, Training, Testing, Handwritten Character Recognition

1. Introduction.

Recognizing handwritten characters is an interesting and challenging work for many researchers now a day. ANNs can be used to accomplish the task. Variation in the handwriting styles makes it very difficult to recognize the handwriting characters. In order to overcome the variation problem common features of the variant characters can be extracted out by applying different feature extraction methods. Some feature extraction methods are already developed. Feature extraction methods like Multiscale Training Technique (MST) [1, 2, 3] are applied to identify the variant characters. These techniques are also applied in non English scripts like Devnagri script [4]. Twelve-Directional-Method is an example of feature extraction method [5]. Some feature extraction methods are appended to the already developed ANNs like perceptron learning method [6, 7, 8, 9].

Common features are extracted out by identifying different shapes formed in a character matrix. The input image matrix is compressed into a lower dimensional matrix. The compressed matrix is segmented in such a way that takes one row and two columns of the image matrix forming the shape of a horse hoof like structure in the image matrix. Many hoofs of different sizes are formed from the input image matrix and presented to the net for training.

The overall program is divided into four parts, Methodology is discussed in Section-2, Result analysis is done in Section-3, Section-4 is the Discussion and Conclusion is described in Section-5.

2. Methodology.

A prototype model is designed using the first five characters of English alphabet set. These characters are written on a piece of paper and scanned using a high definition scanner. Each character image is resized into a matrix of size 15 x 15. This matrix is segmented, taking two columns, one taken from the extreme left and the topmost row (two pixels less than the column taken) forming a horse hoof like structure so that the pixels contained in the first segment is 15 + 13 + 15= 43, (Figure 1). Remaining portion of the matrix vector is segmented in the same way. All the segments are presented to the net one by one to train the net. Once the standard weights are found, test characters are presented to the trained net. Each segment of the middle layer will produce any one of the five outputs {(1,-1,-1,-1,-1),(-1,1,-1,-1,-1),(-1,-1,1,-1,-1), (-1,-1,-1,1,-1) or (-1,-1,-1,-1,1)} corresponding to alphabets A, B, C, D and E (Figure 3). There will be five neurons for identifying five characters in each output segment. The neurons in the output segments are further connected to 5 neurons in the third layer by different weights, each neuron will identify the final character. The segment consisting of maximum number of elements of the image matrix vector will contribute with maximum weight to identify the character. The final neuron with the maximum value, which is called maximum output factor, will be the winner to identify the character.

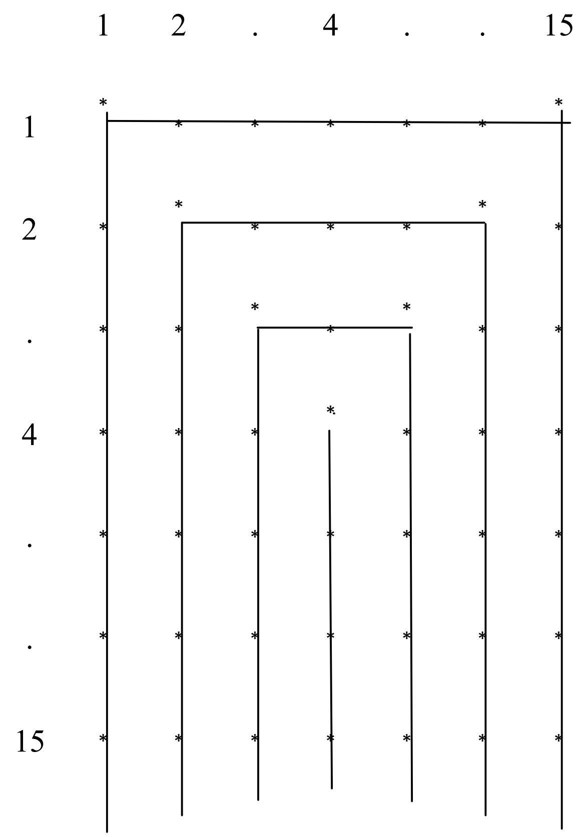


Figure 1. Conceptual diagram of ASIM

2.1 Hoof Segmentation of 15 x 15 Matrix

Eight input segments, forming the shape of a horse hoof, can be carved out of the 15 x 15 matrix. The first segment consists of 15 + 13 + 15 = 43 elements; the second segment consists of 14 + 11 +14 = 39 elements, and so on. Each neuron in each of the input segments is completely interconnected to five neurons in the corresponding output segments.

2.1.1 Perceptron Learning Rule

Figure 2 displays a basic perceptron. The output of the j^{th} perceptron is y_j. Output is calculated by applying the activation function f, which is a function of the net output, y_out.

So, $y_j=f(y\_out_j)$ and

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

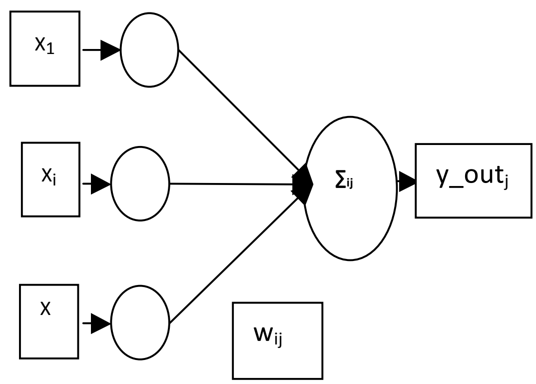


Figure 2. A Basic Perceptron

2.2 Architecture of HSIM net

HSIM net consists of three layers, input layer, middle layer and output layer (Figure 3). Input layer x_i is divided into seven segments of different lengths. Middle layer, y_j also consists of seven segments of length five each. Each segment in the middle layer corresponds to a particular segment in the input layer. Length of the segments in the input layer depends upon the size of the hoof and length of the segment in the middle layer is taken as five in order to classify five alphabets. Weights $w_{ij,}$ taken between input layer and the middle layer are initialized to zero. Each segment in the middle layer is trained by the input layer to produce the correct pattern and in this way the weight w_{ij} is standardized.

There is another layer called output layer that finally recognizes the alphabet. The output layer consists of five neurons. Each neuron identifies an alphabet. The weight v_{jk} taken between middle layer and the output layer is fixed. First neuron of each segment in the middle layer is connected to first neuron in the output layer, second neuron connected to second neuron and so on. The neurons present in the middle layer of the first segment contributes with maximum weights to the output layer as it is trained by maximum number of neurons in the input layer, second segment contributes with comparatively less weight and so on.

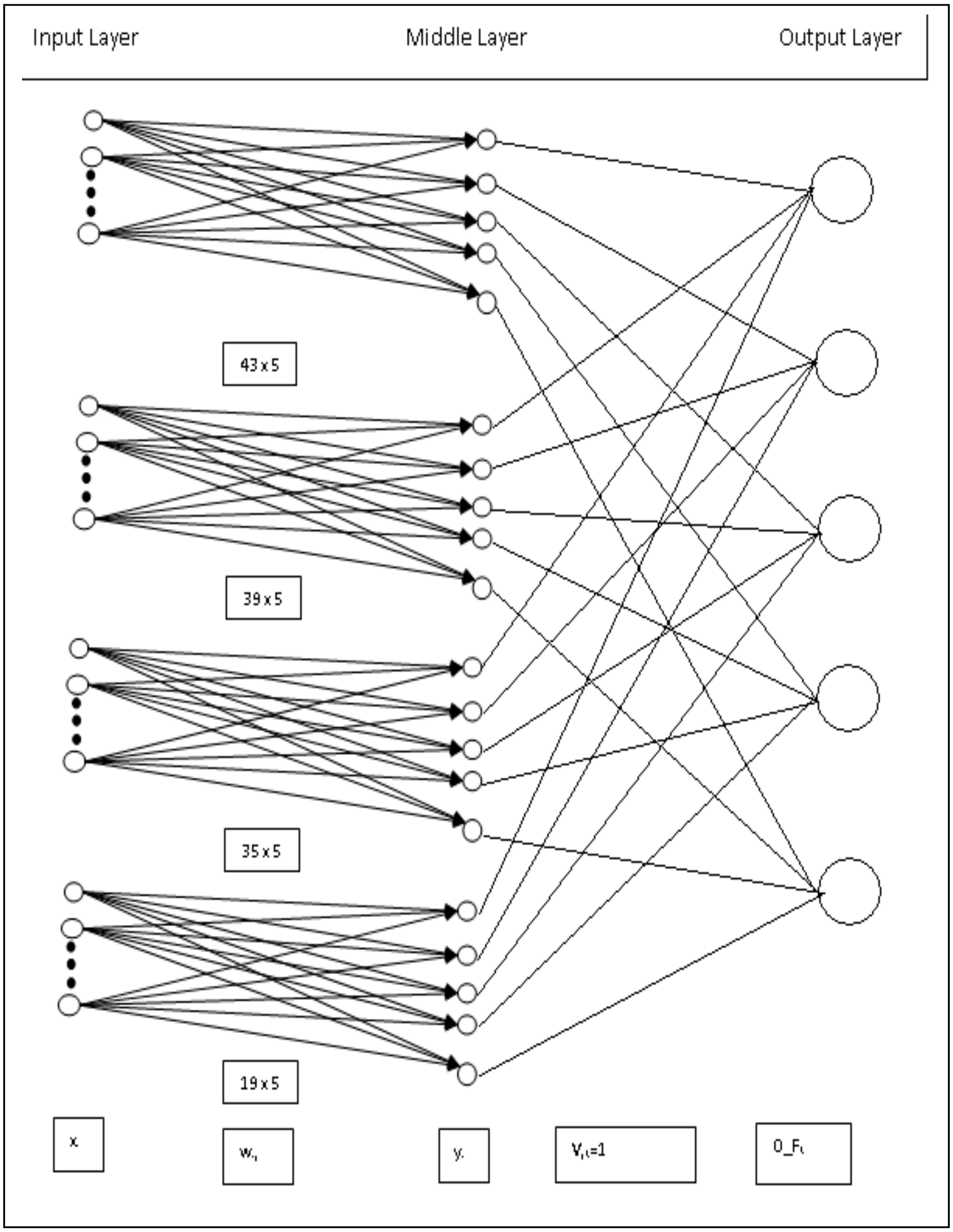


Figure 3. Neural Network representing arrow segmentation of image matrix

2.3 Training of the net to obtain the weight matrix

The training started with an input vector of size [15 x 15 = 225]. First five characters of the English alphabets are considered for training, (Figure 4). The weights are initially set to zero.

The net, (Figure 3) is trained using Perceptron Learning Rule. After training the net for few epochs, the final weights that generates the correct output is obtained. The final weights obtained remain the same for the next few epochs and is considered as the standard weight for testing.

2.4 Testing of the net

Another layer with fixed weights is implemented to test the character sets taken from different individuals, (Figure 3). The neuron having the maximum value at the output layer wins to choose a character. Each neuron recognizes a particular character, like neuron-1 identifies A, neuron-2 identifies B and so on. The neurons responsible for recognizing a character is connected to the neuron in the output layer with different weights. Maximum weight is given to the link between the segments which is having maximum number of neurons. An output factor is calculated for each neuron in the output layer. The neuron with the maximum output factor is the winner to identify the character.

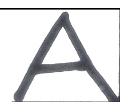
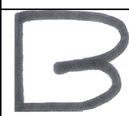
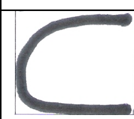
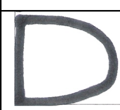
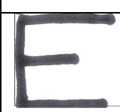
S.No.	**Test Alphabet**
1	
2	
3	
4	
5	

Figure 4. Training Samples

$$O\_F_k = 0.14*Y\_1(k) + 0.13*Y\_2(k) + 0.12*Y\_3(k) + 0.10*Y\_4(k) + 0.9*Y\_5(k) + 0.8*Y\_6(k) + 0.6*Y\_7(k) \quad (3)$$

O_F_k gives the output factor for any k where k = 1 to 5, which represents five neurons in the output layer. Y_1(k), Y_2(k), Y_3(k), Y_4(k) …. Y_7(k) are outputs produced by the different segments of the middle layer by five characters. 0.14, 0.13, 0.12 …. 0.6 are the weights of different segments connecting to the decisive layer. The weights are chosen depending upon the number of elements in a particular segment. For example, segment-1 having 43 elements, the weight chosen is 0.14, which is calculated as below:

$$v = \lceil \text{number of elements}/3 \rceil /100 \qquad (4)$$

43/3 gives 14.3, in order to simplify the calculation, ceiling of the number is taken. The result is further divided by 100 in order to reduce the weight. Low value of weight is taken in order to simplify the calculations.

2.5 Algorithm HSIM

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

Step 2. Compress each m x m matrix and compress into n x n matrix, where m>n.

Step 3. Segment the compressed matrix into eight segments, forming the shape of horse hoofs, starting from the extreme left and right columns and topmost row in the following way:

Hoof Segment-1: 3n-2 elements;

Hoof Segment-2: 3n-6 elements;

Hoof Segment-3: 3n-10 elements;

Hoof Segment-4: 3n-14 elements;

Hoof Segment-5: 3n-18 elements;

Hoof Segment-6: 3n-22 elements;

Hoof Segment-7: 3n-26 elements;

Hoof Segment-8: 3n-37 elements;

Last segment contains only eight elements, as it contains a single line.

Step 4. 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 5. 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 6. 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 7. Initialize weight matrix

```
For k=1 to M
    For i=1 to n
        For j=1 to p
            Wkij=0
        End p
    End n
End M
```

Step 7. Apply Perceptron Learning Algorithm to each segment and find out the standard weights.

Step 8. Scan the sample character to be tested and convert into binary matrix of dimension m x m.

Step 9. Apply Step 3 to the test sample.

Step 10. Present the test vector x into the HSIM net.

Step 11. Apply counter to the output produced, to identify the character.

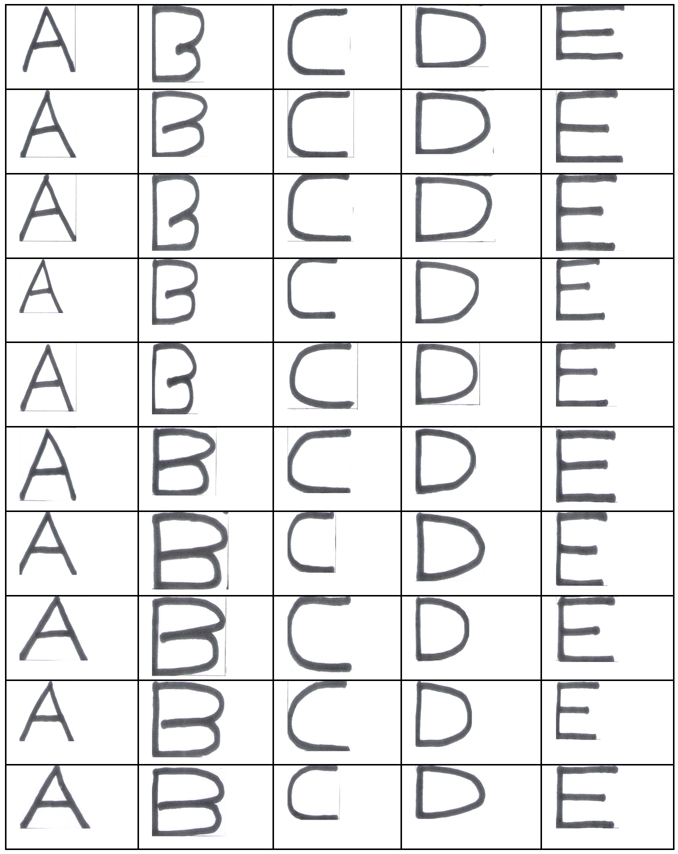


Figure 5. Test Sample Set 1

3. Result Analysis

Matlab software has been used to test the accuracy of the net. Five characters A, B, C, D and E are taken initially for testing. The response of the experiment shows good results for the first five alphabets.

Following are the Neural Network parameters used in the experiment:

Number of input pattern segments=8

Number of Neurons in each output group in the middle layer = 5

Number of neurons in the output layer=5

Training Algorithm used = Perceptron

Number of Epochs = 3

Threshold, $\theta = 0.2$, Learning Rate, $\alpha = 1$

In Figure 3, x_i is the input to the net, w_{ij} is the weight matrix which is dynamic, y_i is the activation produced by the middle layer of the net, v_{jk} is the fixed weight. Figure 4 shows the alphabets taken to train the net. A sample set of 50 alphabets were, initially, presented to the net, out of that 37 samples were identified by the net, Figure 5. Another sample set of 50 characters were presented to the net which identifies 39 characters, Figure 6. Therefore, the total accuracy of the net is measured 76%, by considering two sample sets. Table 1 and Table 3 shows the output factors produced by different samples of characters which are used to identify each character. Table 2 and Table 4 show the accuracy of identifying each character individually with an average performance of the net. It is found that character 'A' shows 100% accuracy, which indicates that shape of character plays a vital role in the identification of the character. The performance degrades with the inclusion of curves in the image matrix.

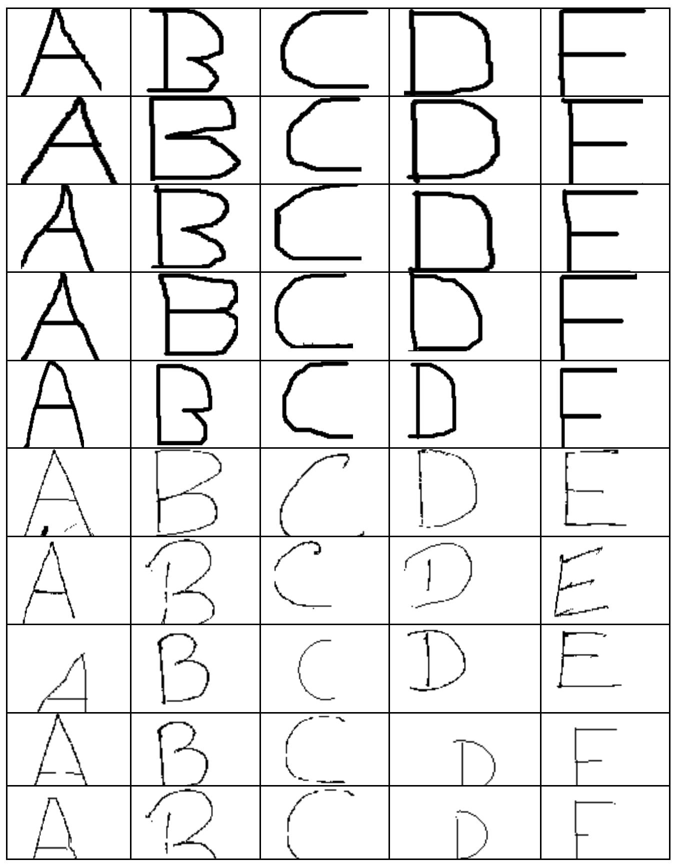


Figure 6. Test Sample Set 2

Table 1. Output factors produced by first set of alphabets

Set	Character	O_F_1	O_F_2	O_F_3	O_F_4	O_F_5
1	A	**0.19**	-2.29	-3.29	-3.29	-3.29
	B	-2.06	**-0.67**	-3.29	-1.21	-3.29

Set	Character	O_F_1	O_F_2	O_F_3	O_F_4	O_F_5
	C	-2.69	-2.29	**-1.29**	-1.59	-3.29
	D	-2.31	-0.39	-3.29	**-0.21**	-3.29
	E	-1.69	-1.49	-3.29	-3.19	**-0.69**
2	A	**-0.01**	-3.29	-3.29	-3.29	-3.29
	B	-3.29	**0.13**	-0.49	-3.29	-1.69
	C	-2.69	<u>**-0.29**</u>	-2.09	-3.19	-3.29
	D	-3.29	-2.29	<u>**-2.09**</u>	-2.91	-3.29
	E	-2.29	-2.39	-3.29	-3.19	**-2.09**
3	A	**0.65**	-2.29	-3.29	-3.29	-3.29
	B	-3.29	**-1.29**	-1.59	-2.81	-2.79
	C	-2.43	<u>**-1.19**</u>	-3.29	-1.59	-3.29
	D	<u>**-1.69**</u>	-2.29	-2.09	-3.19	-3.29
	E	-2.03	<u>**-1.29**</u>	-2.09	-1.49	-3.29
4	A	**1.51**	-2.29	-3.29	-3.29	-3.29
	B	-2.53	**-0.79**	-3.29	-3.19	-2.09
	C	-1.56	-3.29	-3.29	<u>**-1.49**</u>	-3.29
	D	-2.69	<u>**-2.29**</u>	-3.29	-2.91	-3.29
	E	<u>**-1.43**</u>	-2.29	-3.29	-1.59	-2.69
5	A	**1.31**	-2.29	-3.29	-3.29	-3.29
	B	-3.29	**-1.29**	-1.59	-3.09	-2.79
	C	-3.03	-2.09	**-1.59**	-2.49	-3.29
	D	-2.69	-2.29	-2.09	**-1.59**	-3.29

Set	Character	O_F$_1$	O_F$_2$	O_F$_3$	O_F$_4$	O_F$_5$
	E	-3.03	**-0.29**	-2.09	-1.49	-3.29
6	A	**1.59**	-2.29	-3.29	-3.15	-3.29
	B	-2.79	-2.79	-3.29	-3.29	**-2.09**
	C	-2.69	-2.29	**-1.97**	-2.91	-3.16
	D	-2.69	**-2.19**	-3.29	-3.09	-3.29
	E	-3.29	**-0.89**	-3.29	-3.01	-2.09
7	A	**0.97**	-3.29	-3.29	-3.29	-3.29
	B	-1.89	**-1.45**	-2.49	-3.15	-2.49
	C	-2.69	**-0.29**	-3.29	-1.49	-3.29
	D	-1.32	-2.19	-2.09	**-1.21**	-3.29
	E	-3.29	-1.29	-2.09	-2.49	**-0.29**
8	A	**0.21**	-2.29	-3.29	-3.29	-3.29
	B	-3.29	**-1.29**	-2.19	-1.49	-2.19
	C	-2.09	-2.39	**-1.69**	-3.19	-3.29
	D	-1.43	-3.19	-2.09	**-1.09**	-3.29
	E	-2.79	-1.39	-3.29	-1.69	**-1.29**
9	A	**1.45**	-2.29	-3.29	-3.29	-3.29
	B	-2.79	**-1.76**	-3.29	-3.09	-2.09
	C	-2.19	-2.09	**-2.09**	-1.59	-3.29
	D	-3.03	-2.29	-2.69	**-1.49**	-3.29
	E	-2.79	-1.79	-3.29	-3.29	**-1.29**
10	A	**1.39**	-3.29	-3.29	-3.29	-3.29

Set	Character	O_F$_1$	O_F$_2$	O_F$_3$	O_F$_4$	O_F$_5$
	B	-3.29	**-1.29**	-2.79	-2.25	-2.79
	C	-1.69	-2.09	**-1.49**	-1.59	-3.29
	D	-2.45	-2.29	-3.29	**-1.95**	-3.29
	E	-3.29	1.11	-3.29	-3.19	**-0.29**

Table 2. Accuracy of the net by presenting sample set 1

S. No	Alphabets	No. of Characters presented to the net	No. of characters identified	Percentage of identification
1	A	10	10	100%
2	B	10	9	90%
3	C	10	6	60%
4	D	10	6	60%
5	E	10	6	60%
			Average	74%

Table 3. Output factors produced by second set of alphabets

Set	Character	O_F$_1$	O_F$_2$	O_F$_3$	O_F$_4$	O_F$_5$
1	A	**0.10**	-2.23	-1.29	-3.09	-3.19
	B	-1.26	**-0.61**	-1.29	-1.21	-3.29
	C	-1.61	-2.19	**-1.22**	-2.52	-2.29
	D	-1.32	-1.19	-2.29	**-0.14**	-2.29
	E	-1.29	-2.49	-2.29	-3.29	**-0.55**
2	A	**-1.21**	-2.19	-2.27	-3.21	-3.21
	B	-1.29	**1.23**	-1.19	-1.29	-1.69

Set	Character	O_F_1	O_F_2	O_F_3	O_F_4	O_F_5
	C	-2.62	-2.29	**-2.19**	-3.29	-2.49
	D	-2.22	-1.22	**-2.09**	-2.92	-2.29
	E	-2.19	-2.19	-3.19	-3.19	**-2.14**
3	A	**1.35**	-1.29	-2.29	-2.29	-2.29
	B	-2.25	**-1.21**	-2.59	-2.82	-2.72
	C	-2.13	-2.19	**-2.09**	-2.59	-3.29
	D	**-1.61**	-2.19	-2.19	-2.19	-2.29
	E	-2.33	**-1.27**	-2.49	-1.69	-2.29
4	A	**1.01**	-3.19	-3.19	-3.19	-3.29
	B	-2.51	**-1.79**	-3.21	-3.12	-2.39
	C	-2.36	-2.29	-2.29	**-1.19**	-3.29
	D	-2.29	**-2.21**	-3.22	-2.92	-2.29
	E	**-1.44**	-2.24	-3.24	-1.57	-2.49
5	A	**1.11**	-2.24	-3.21	-3.21	-3.29
	B	-3.22	**-1.19**	-1.29	-2.09	-2.19
	C	-3.13	-2.19	**-1.29**	-1.49	-2.29
	D	-2.62	-2.22	-2.29	**-1.39**	-3.19
	E	-3.23	**-0.19**	-2.19	-1.29	-2.29
6	A	**1.29**	-2.19	-2.29	-1.15	-1.29
	B	-2.19	**-1.79**	-3.19	-3.29	-2.19
	C	-2.49	-2.19	**-1.27**	-2.92	-3.26
	D	-2.63	**-2.39**	-3.34	-3.19	-3.39
	E	-3.25	-0.59	-4.29	-3.21	**-0.29**

Set	Character	O_F_1	O_F_2	O_F_3	O_F_4	O_F_5
7	A	**1.87**	-3.21	-3.21	-2.24	-3.29
	B	-2.29	**-1.25**	-2.44	-2.12	-2.45
	C	-2.61	**-1.29**	-3.29	-2.49	-3.29
	D	-1.12	-2.13	-2.19	**-1.11**	-3.19
	E	-2.29	-2.29	-2.29	-2.39	**-2.03**
8	A	**2.21**	-2.21	-3.21	-3.21	-3.21
	B	-3.22	**-1.29**	-2.11	-1.69	-2.11
	C	-2.19	-2.19	**-1.49**	-2.19	-2.29
	D	-1.42	-3.49	-2.29	**-1.29**	-3.22
	E	-2.76	-2.39	-2.29	-2.69	**-1.59**
9	A	**2.45**	-2.21	-1.29	-1.29	-3.29
	B	-2.76	**-1.79**	-3.21	-3.19	-2.19
	C	-2.19	**-2.09**	-2.19	-2.59	-3.29
	D	-3.23	-4.29	-2.49	**-0.29**	-3.19
	E	-2.74	-1.49	-3.49	-3.49	**-1.19**
10	A	**2.39**	-2.29	-3.29	-3.29	-2.29
	B	-3.22	-2.19	-2.39	-2.22	**-2.73**
	C	-1.63	-2.19	**-1.29**	-1.69	-2.29
	D	-2.42	-2.29	-2.29	**-1.40**	-2.29
	E	-3.19	2.11	-3.19	-3.19	**-0.39**

Table 4. Accuracy of the net by presenting sample set 2

S. No	**Alphabets**	**No. of Characters presented to the net**	**No. of characters identified**	**Percentage of identification**
1	A	10	10	100%
2	B	10	9	90%
3	C	10	7	70%
4	D	10	6	60%
5	E	10	7	70%
			Average	78%

Table 5. Comparison of different handwritten character methods

S. No.	**Name of the method**	**Nature of method**	**Accuracy**
1.	Handwritten Devanagri Character Recognition using Gradient Features, [10]	More processing time required	94%
2.	Optical Character Recognition using 40-point Feature Extraction and Artificial Neural Network, [11]	More processing time required	83.84%
3.	Robust Handwritten Character Recognition with Features Inspired by Visual Ventral Stream, [12]	More processing time required	81% to 99%
4.	Handwritten Character Recognition using Twelve Directional Feature Input and Neural Network, [5]	Average processing	75% to 97%
5.	Combining Multiple Feature Extraction Techniques for Handwritten Devnagri Character Recognition, [4]	More processing time required	92.80%
6.	Handwritten Character Recognition using Multiscale Neural Network Training Technique, [3]	Long learning time	85%
7.	An Artificial Neural Network Model to Recognize English Alphabets, [13].	Use of binary pixels to train the ANN	82.5%
8.	Sliding window based on the Hough transform as feature extraction technique, [14].	More processing time required	92.5%
9.	Modified Hough transformation technique by SVM classifier, [15].	More processing time required	93.1%
10.	Modified Hough transformation technique by MLP classifier, [15].	More processing time required	72.5%

11.	Four view projection profiles technique by SVM classifier, [15].	More processing time required	96.04%
12.	Four view projection profiles technique by MLP classifier, [15].	More processing time required	98.7%
13.	Handwritten English Character Recognition using Hoof Segmentation of Image Matrix(HSIM)	Simple and very less processing required	76%

4. Discussion

HSIM is compared with a number of already developed handwritten recognition method. It is found that in terms of simplicity and complexity in processing, HSIM is much better method than the other methods. The reason behind developing HSIM is to develop a more generalized method that is capable of accepting text in any form without any preprocessing. The method can be developed in future to enhance the accuracy keeping the nature of the method as simple as possible. HSIM is a type of feature extraction method. The aim of HSIM is to extract out common features of different samples of a character by slicing out samples from different portions of the character.

It is found that HSIM is comparatively simple and requires very less processing which will reduce the programming cost. Performance can be further enhanced with some pre-processing and further development in the method. HSIM is purely ANN based.

5. Conclusion

HSIM is a better approach as compared to the previously tried methods of segmentation. Resizing of the input matrix helped a lot to reduce the number of neurons in the input matrix. Resizing of the input makes the net small, simple and fast. Hoof Segmentation is a better approach than the previously tried segmentation because segmentation of the matrix in this way produces more variation in the patterns and helps to obtain accuracy to large extent in the identification of the character patterns.

The weights are chosen in this way because more the number of neurons in a segment more is the information the segment is contributing to the net to identify a character. Maximum value of output factor among output factors produced by different layers identifies a character.

References

1. G.Robinson(1995), "The Multiscale Technique", www.netlib.org 2010 (June).
2. "Handwritten Character Recognition", http://tcts.fpms.ac.be 2010 (June).
3. Velappa Ganapathy, kok Leong Liew, "Handwritten Character Recognition Using Multiscale Neural Network Training Technique", Proceedings of World Academy of Science, Engineering and Technology, Vol. 29 (May 2008), pp. 32-37.
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), 2008 IEEE Region 10 Colloquium and the third ICIIS Kharagpur, INDIA, December 8-10, pp. 1-6.
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), International Journal of Computer Applications (0975-8887), Vol. 1 No. 3, pp. 82-85.
6. G.N. Swamy, G. Vijay Kumar, "Neural Networks", Scitech, India, 2007.
7. L. Fausett, "Fundamentals of Neural Networks, Architectures, Algorithms and Applications", Pearson Education, India, 2009.
8. Apash Roy and N R Manna, "Character Recognition using Competitive Neural Network with Multi-scale training", UGC Sponsored National Symposium on Emerging Trends In Computer Science (ETCS 2012) on 20-21 January 2012, pp 17-20.

9. Apash Roy and N R Manna, “Competitive Neural Network as applied for Character Recognition ” - International Journal of advanced research in Computer science and Software Engineering, Volume 2, Issue 3, 2012, pp 06-10.
10. Ashutosh Aggarwal, Rajneesh Rani, RenuDhir, “Handwritten Devanagri Character Recognition using Gradient Features”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 2, Issue 5, May 2012, ISSN 2277 128X, pp. 85-90.
11. Sandeep, Saha, Nabarag Paul, Sayam Kumar Das, Sandip Kundu, “Optical Character Recognition using 40-point Feature Extraction and Artificial Neural Network”, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 4, April 2013, ISSN 2277 128X, pp. 495-502.
12. Ali Borji, Mandana Hamidi, Fariborz Mahmoudi, “Robust Handwritten Character Recognition with Features Inspired by Visual Ventral Stream”, © Springer Science+Business Media, LLC. 2008, published online (31 August 2008), pp. 97-111.
13. Y Perwej and A Chaturvedi, “Neural Networks for Handwritten English Alphabet Recognition”, International Journal of Computer Applications, Volume 20, No. 7, pp. 1-5, 2011.
14. M Amrouch, Y Es-Saady, M El Yassa, D Mammass, “Handwritten Amazigh character recognition system based on continuous HMMs and directional features”, International Journal of Modern Engineering Research, Volume 2, 2, pp 436-441, 2012.
15. M Garg and D Ahuja, “A Novel Approach to Recognize the off-line Handwritten Numerals using MLP and SVM Classifiers”, International Journal of Computer Science & Engineering Technology (IJCSET), ISSN: 2229-3345 Volume 4 No. 07, pp 953-958, July, 2013.
16. Al-Shridah, Nuhammad, Sharieh, Ahmad, “Recognition process of handwritten and typed Arabic letters”, AMSE Journal, Advances in Modelling, Signal Processing and Pattern Recognition, Volume 45, Issue 1, pp 1, 2002.

17. N Liolios, K Anastasiou, B Kostos, “A new shape transformation approach to handwritten character recognition”, AMSE Journal, Advances B, Volume 46, Issue 6, pp 55, 2003.
18. K G Srinivasagan, L Ganesan, S Arumugam, AMSE Journal, Advances in Modelling, Signal Processing and Pattern Recognition, Volume 50, Issue 2, pp 64, 2007.