Engineering Journal of Qatar University, Vol. 7, 1994, p. 37 - 51

RECOGNITION OF CURSIVE TEXTS USING HAMMING NEURAL NETS

S. D. Katebi
Department of Computer Science and Engineering
School of Engineering, Shiraz University
Shiraz, Iran

ABSTRACT

Hamming Neural networks are employed for cursive text character separation and recognition. The digitized image of the scanned text is first enhanced by applying simple contrast stretching algorithm. Image registration is then performed to eliminate the white margins. Lines are separated by rows of white pixels and the morphological components identified. The projection method in conjunction with histogram analysis is used to estimate the width of each character in a word. The word is approximately decomposed into its constituent characters. The Hamming net is used to identify thus separated characters, while supposing the included portions of the adjacent characters as noise. The property of the trained Hamming net, i.e., the associative feature for detecting the best matched patterns from the prototype in a mean square sense, is the basis for constructing the recognition algorithm.
Illustrative examples are presented and it is shown that the proposed approach is fast and efficient to provide an on-line separation and classification system for cursive texts such as Persian (Farsi), Arabic, Urdu and English script.

NOMENCLATURE

M	Number of input nodes
N	Number of ouput nodes
x	Input to neural network
y	Output of neural network
w	Connection weight
θ	Threshold in the lower subnet
t	Threshold in the upper subnet
f	Threshold logic nonlinearity
ε	A small positive number
i, j, l, k	Positive integers
Iter	Number of iterations

INTRODUCTION

English, as the major language of science and technology has enjoyed the privilege of being the first language considered for machine data entry. English character readers date back to the 1970's. The problem of automatic reading of machine printed English text seems to be completely solved. Many commercial packages are now available, which enable even general purpose computers to read and recognize machine printed text.

However, little research has been reported on the processing of characters other than Latin fonts. Chinese and Kanji have been given more consideration in recent years as presented in Blackwell (1992), and quite a few reports are available on character recognition of Chinese and Kanji. The only similarity between Chinese and Latin text is the separation of characters, a feature not present in Arabic, Persian, or Urdu.

This paper is concerned with the problem of Persian character recognition, the result of which may be extended to other cursive texts such as Arabic, Urdu, and Latin scripts. There are only a few papers dealing with the problem, among which the more important ones are mentioned below.

Parhami (1981) discusses character separation using Potential and Actual Connection Columns. He further uses geometrical features of each character for classification. Almuallim (1987) classifies handwritten Arabic character strokes using their geometrical and topological properties. Mahmoud (1991) uses Fuzzy ISODATA to skeletonize Arabic characters in his Clustering Based Skeletonization Algorithm. Goraine (1992) proposes a syntactical method for recognizing thinned strokes of cursive Arabic script. Alyousefi (1992) uses projections or histograms to separate characters and statistical moments to identify them. In the field of English script and handwritten characters much more research have been reported.

In the field of Neural networks several *de facto* standard texts exist, including Lippmann (1987), and Widrow (1990). As far as the use of neural networks in character recognition is concerned, Fukushima's paper (1991) on alphanumeric character recognition incorporating Neocognitron, Blackwell's method (1992) using Associative memory, and Pavlidis' Polygonal Approximation method (1975) are typical.

The present work introduces the application of Hamming nets to the problem of optical character recognition of Persian (Farsi) text. Algorithmic

procedures based on Hamming neural nets for both character separation and classification is described. It is shown that the proposed approach provide a means for implementing an efficient and fast on-line Optical Character Recognition (OCR) system for cursive-based written forms .

In the following sections, the salient features of cursive script are described and the difficulties in recognition are elaborated. A theoretical overview of the Hamming net and its application to cursive character recognition is presented. The implementation procedures are then discussed and finally results and conclusions are presented.

CHARACTERISTICS OF CURSIVE TEXT

In automated recognition of cursive machine printed text a number of difficulties are encountered. These problems are elaborated, and salient features of Persian text are discussed.

The main feature of Persian text which is in contrast with Latin, is its being cursive. The characters are continuously written, obeying specific rules. A number of characters join to form segments or strokes. The space between adjacent strokes in a word are less than those between words. In this aspect the identification of words may use the same methods as English OCR methods. Another feature of the cursive script of Persian, Arabic, and Urdu is that they are written from right to left, however this feature is not critically disturbing.

There are a total of 32 basic characters in Persian, compared to the 28 in Arabic and the 26 in English. These are shown in Table (I). There are also three special characters, namely, a variation of *Alef* "آ", four *Hamzeh* letters "ء ؤ ئ أ", and combination of *Laam* and *Alef* "لا". Additionally, Arabic script has other characters ("ک", "ی", and "ۀ"). The letters in general have four forms: The beginning, middle, end, and isolated. Because of the great similarity between the middle form and the beginning form, they can be classified as one. The same is usually true for the end form and the isolated form.

Another feature of the Persian script also present in Arabic is the minor shapes used as modifiers. Diacritics play the role of short vowels which are added to the basic letters in specific positions above or below the letter. Therefore, the actual letter may consist of three parts: The primary shape of the character; The secondaries, which are a dot or dots above or below the letter; and the diacritics. Considering this division, the total number of primary letter

Table 1: Persian Character Set

Char. Name	Begin. Form	Middle Form	End Form	Isolated Form	Char. Name	Begin. Form	Middle Form	End Form	Isolated Form
Alef	ا	ا	ا	ا	*Saad*	صـ	ـصـ	ـص	ص
Be	بـ	ـبـ	ـب	ب	*Zaad*	ضـ	ـضـ	ـض	ض
Pe	پـ	ـپـ	ـپ	پ	*Ta*	طـ	ـطـ	ـط	ط
Te	تـ	ـتـ	ـت	ت	*Za*	ظـ	ـظـ	ـظ	ظ
Se	ثـ	ـثـ	ـث	ث	*Ein*	عـ	ـعـ	ـع	ع
Jim	جـ	ـجـ	ـج	ج	*Ghein*	غـ	ـغـ	ـغ	غ
Che	چـ	ـچـ	ـچ	چ	*Fe*	فـ	ـفـ	ـف	ف
He	حـ	ـحـ	ـح	ح	*Ghaaf*	قـ	ـقـ	ـق	ق
Khe	خـ	ـخـ	ـخ	خ	*Kaaf*	کـ	ـکـ	ـک	ک
Daal	د	ـد	ـد	د	*Gaaf*	گـ	ـگـ	ـگ	گ
Zaal	ذ	ـذ	ـذ	ذ	*Laam*	لـ	ـلـ	ـل	ل
Re	ر	ـر	ـر	ر	*Mim*	مـ	ـمـ	ـم	م
Ze	ز	ـز	ـز	ز	*Noon*	نـ	ـنـ	ـن	ن
Zhe	ژ	ـژ	ـژ	ژ	*Vaav*	و	ـو	ـو	و
Sin	سـ	ـسـ	ـس	س	*He*	هـ	ـهـ	ـه	ه
Shin	شـ	ـشـ	ـش	ش	*Ye*	یـ	ـیـ	ـی	ی

forms reduces to 35, which may considerably reduce the feature space dimensions.

The other problem encountered in Persian character recognition is the diversity of Persian fonts used. The typewriter fonts used in administrative and educational documents are numerous with an average correlation of %75. This makes it difficult to build a global font-independent character recognition system. Besides, different fonts use different font widths for different letters.

Therefore, the nature of Persian text calls for an *ad hoc* form of solution to the problem of character recognition. In this work neural nets are used to solve

the problem of character separation as well as classification. The characteristics of Persian text are used to design an efficient character recognition system optimized to give the maximum performance for cursive text recognition. The processing speed and the robustness of the method seem to be promising.

THEORETICAL OVERVIEW

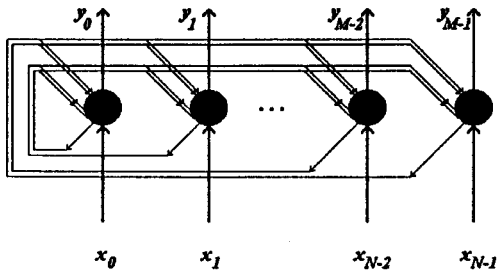
A neural network is a combination of inter-linked cells, called neurons. Each neuron has a specific response characteristic which mathematically models the input/output relationship. In general the output of a node is a linear or nonlinear function of the weighted sum of its different inputs. The basic memory component of a neural net is the weights assigned to links joining its neurons to one another. A network is identified by three major factors: the net topology, node characteristics, and learning algorithm.

The net topology states how different neurons are joined together and which links are necessary. The nature of the net is mostly dictated by its topology. The input to a net may be either binary, or continuous. For our purpose a classifier is needed to process binary data obtained from images of a character. A node may have a specific characteristic function. The three main functions used in the literature are hard limiters, threshold logic, and the sigmoidal function. The sigmoidal function, because of its graded non linearity best suits problems with human factors present. However computer implementation of a hard limiter is the simplest.

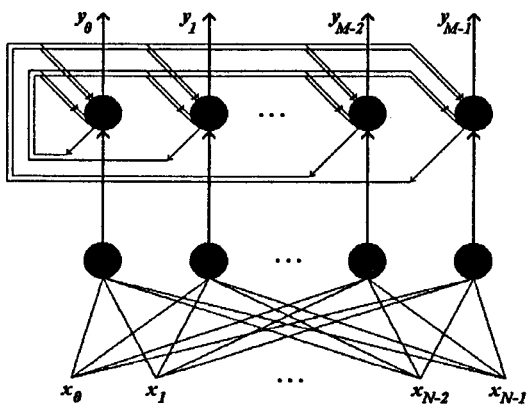
The learning algorithm or training rules may be divided into two main categories: supervised, and unsupervised. In supervised learning a net is given both the input and the correct output - in our case the class the input belongs to - to decide upon. In unsupervised learning the net is required to build classes itself and thus it may assign an input to either of the classes already built or build a new class. In this paper we will focus on the binary supervised nets which has been introduced recently and in particular on the Hopfield and the Hamming nets.

The Hopfield net (1984) is normally used with binary inputs and is most appropriately used for associative memory and optimization problems. An N node Hopfield net is shown in Figure (1-a). The nodes are considered to contain hard limiter non-linearities shown in Figure (1-c) and the inputs and the outputs take binary values, "-1" or "+1". The input, is a binary sequence applied to the x_i's at time zero, the network iterates until convergence when output nodes y_i's remain unchanged. The weights w_{ij}'s, are preset using the exemplar for each

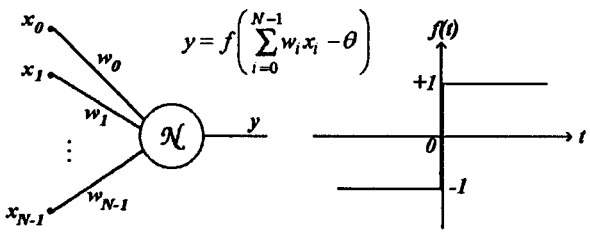
known class. At each time step the output of the network will get nearer to the prototype of the class closest to the unknown input. For classification purposes another step is taken to distinguish the class to which an input belongs to, by matching the output of the Hopfield net with the prototype.



a)- Hopfield neural network.



b)- Hamming neural network.



c)- A Neuron with Hard Limiter Characteristics.

Fig. 1: Neural networks and neurons

The Hopfield net is usually exemplified by problems where inputs are generated by selecting a prototype and reversing bit values randomly and independently with a given probability. This is a classic problem in communications theory that occurs when binary fixed-length signals are sent through a memoryless binary symmetric channel. In this case the optimum error classifier calculates the Hamming distance to the exemplar for each class and selects that class with the minimum Hamming distance. The Hamming distance is the number of bits in the input which do not match the corresponding exemplar bits. Lippmann's (1987) proposed Hamming neural net is used to implement the pattern recognition algorithms. This is shown in Figure (1-b)..

The net is composed of two subnets. The first subnet, contains M input nodes and N output nodes, therefore, it has MxN connections and calculates the matching scores. It finds the similarity measure of the input and the prototypes of each class. The output of this layer is then applied iteratively to the second subnet, which finds the maximum of its M inputs. This *MAXNET* usually picks up the maximum input and magnifies it while attenuating the other inputs. After convergence the total output represents the class whose prototype optimally matches that of the input. The method used for setting weights and thresholds and updating them is presented here closely following that of Lippmann (1987).

***Step 1) Assign Connection Weights and Offsets*:**

In the lower subnet:

$$w_{ij} = \frac{x_i^j}{2}, \qquad \theta_j = \frac{N}{2}, \qquad \text{for } 0 \le i \le N-1, \;\; 0 \le j \le M-1$$

In the upper subnet:

$$t_{kl} = \begin{cases} 1, & k = l \\ -\varepsilon, & k \ne l, \;\; \varepsilon \prec 1/M \end{cases} \qquad \text{for } \; 0 \le k,l \le M-1$$

where w_{ij} is the connection weight from input node i to output node j in the lower subnet and q_j is the threshold in that node. The connection weight from node k to node l in the upper subnet is t_{ij} and all thresholds in this subnet are zero. x_i^j is element i of prototype j.

Step 2) Initialize with unknown input pattern:

$$y_j(0) = f_t\left(\sum_{i=0}^{N-1} w_{ij} \,.\, x_i - \theta_i\right) \quad \text{for } 0 \le j \le M-1$$

where $y_j(t)$ is the output of node j in the upper subnet at time t, x_i is element i of the input, and f_t is the threshold logic nonlinearity of Figure (1-c)..

Step 3) Iterate until convergence:

$$y_j(t+1) = f_t\left(y_j(t) - \varepsilon \cdot \sum_{k \neq j} y_k(t)\right) \quad \text{for } 0 \le j,k \le M-1$$

Convergence is reached when the output of only one neuron is active.

IMPLEMENTATION PROCEDURE

The implementation procedure of the complete character recognition system consists of the following steps.

Digitization: The text is scanned using a standard desktop page scanner. The scanner used was an Epson GQ-600 with a maximum resolution of 600 dot per inch. However the maximum scanner resolution was not necessary in this case. For the typewritten texts used as test script, a resolution of 150 dpi proved to be sufficient. The image was saved in standard image formats on an 80486 IBM-compatible PC. The Tagged Image File Format (TIFF) was used for simplicity. Each pixel of the image has a maximum gray level of 256.

Preprocessing: As the scanned image may be contaminated with noise, a number of preprocessing measures are taken. Usually the image is enhanced using specially implemented algorithms. The contrast stretching algorithm, proved to be enough in this case. In this algorithm the maximum range of gray shades are used to present the image. This also normalizes the gray level histogram of the image. The image is also binarized to provide the binary data for neural nets for further processing.

Registration: The image is then registered, such that the data presenting white margins of the paper are canceled out. This step also provides a start point for

the next step of the processing sequence. Since the first line is aligned at the beginning of the image, its identification is simplified.

Component Break Down: In this stage each part of the image is extracted from the whole image. The morphological components of the image are identified using the blank (white) pixels between the entities. Lines are separated by rows of white pixels, and easily extracted. After dividing the image into subimages each including a line, the line is divided into words. Each word is distinguished by the columns of spaces at its sides. A word in turn is broken into several characters. To perform character separation, Hamming nets are used to identify characters sequentially from right to left. This simultaneously separates and detects the characters as they are seen. The algorithm used works as follows. First an approximate evaluation of character width is performed using projection. The rise and fall of the histogram is utilized to estimate the average character width. The actual character limits differ from the estimated separation borders. Thus each character may be accompanied by a number of columns of pixels from its adjacent left or right character. These pixels are ignored by the Hamming net as being random noise. When the character is identified, its exact margins are known. This procedure is repeated for all the characters in a line.

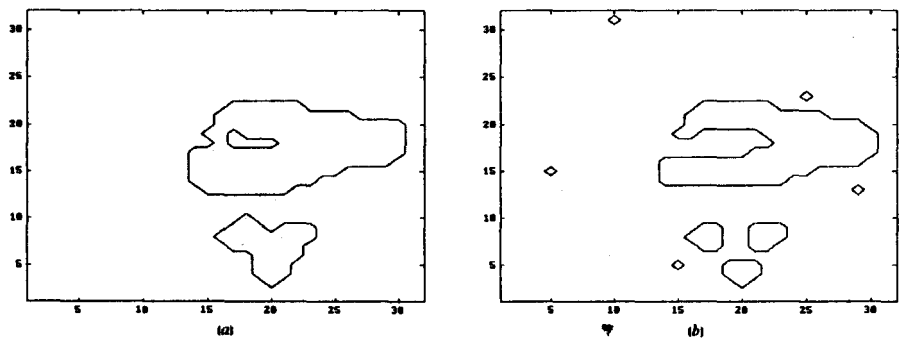
Recognition: The Hamming net, after being trained once, has the associative feature of detecting a pattern matching any of the prototype in a mean square sense. This property is used to construct the recognition algorithm. First the weights of the network connections are initialized using single typical prototype for each class. The prototype are scanned from a sample of characters typed by Olympia, Facit, and IBM typewriters. Several fonts with different pitch sizes were averaged to give meaningful prototype. A drawback of the Hamming net for this application is that their simplicity is gained on the expense of the inability to utilize several prototypes. This deficiency is overcome by averaging the prototype. The network simulation is both implemented in mathematical language using MathWorks' PC MATLAB, and the mid-level language C. A recognition rate of 20 characters per second was obtained on an 80486 DX-2 66.

EXPERIMENTAL RESULTS

The proposed techniques are applied to recognition of samples of Persian text and characters. The results are presented in two main categories. First those regarding the Hamming net, the learning process, and the character recognition features. The second category deals with the character separation. In the first case, the parameters of the neural net are obtained. The net is then used

to classify images of a single character containing binary noise. For the second category, an example of character separation of a Persian word is given where the image of the word is broken into connected character images including part of the adjacent characters. The connected characters are classified by the Hamming net.

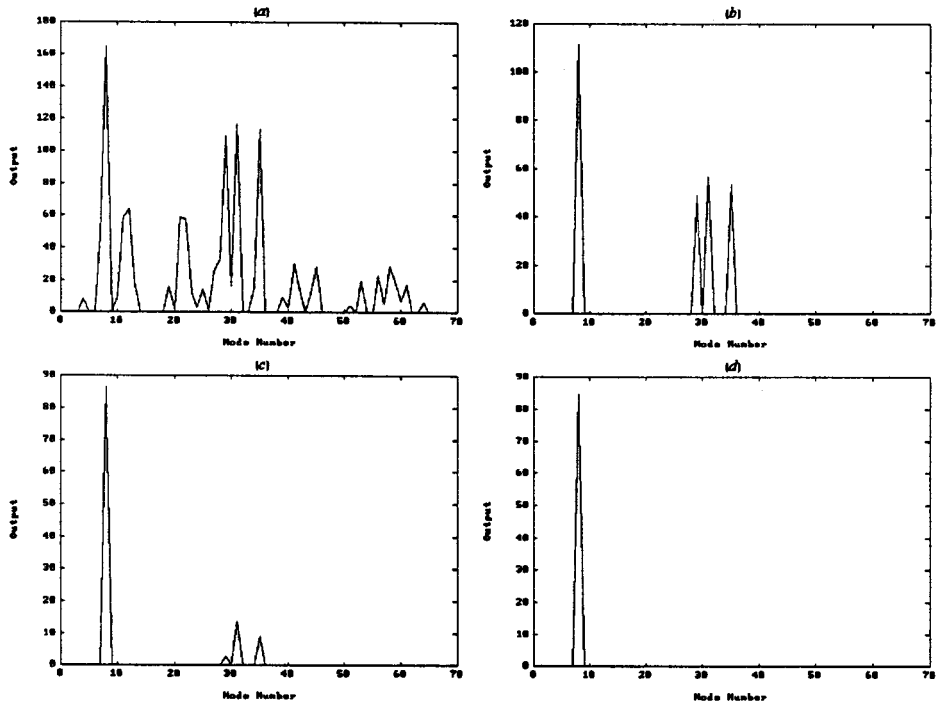
Figure (2) shows the contour presentation of the letter *Che*. In Figure (2-a), the original form of the character is depicted. The character is intentionally contaminated with binary noise, resulting in a distorted form, as shown in Figure (2-b). The noisy image is presented to the network, it iterates to find the



a) Original form, b) The character contaminated with binary noise

Fig. 2: Different images representing the beginning *Che* character

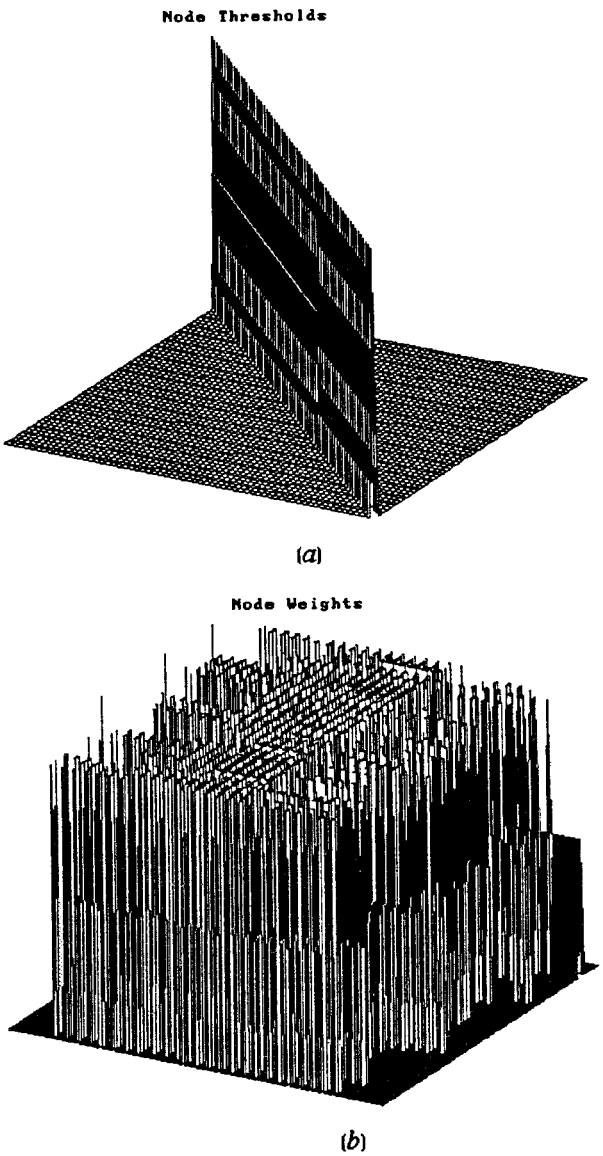
maximum similarity. Results for four iterations are shown in Figure (3). It is observed that the nearest character is assigned the maximum output, and as the iterations proceed other outputs are attenuated. The algorithm converged in less than 50 iterations in most experimental cases. The thresholds of the hamming net neurons, and the connection weights are shown in Figure (4). These values have been obtained after the net was trained with the prototypes.



a) Iter=1, b) Iter=10, c) Iter=29, d) Iter=40

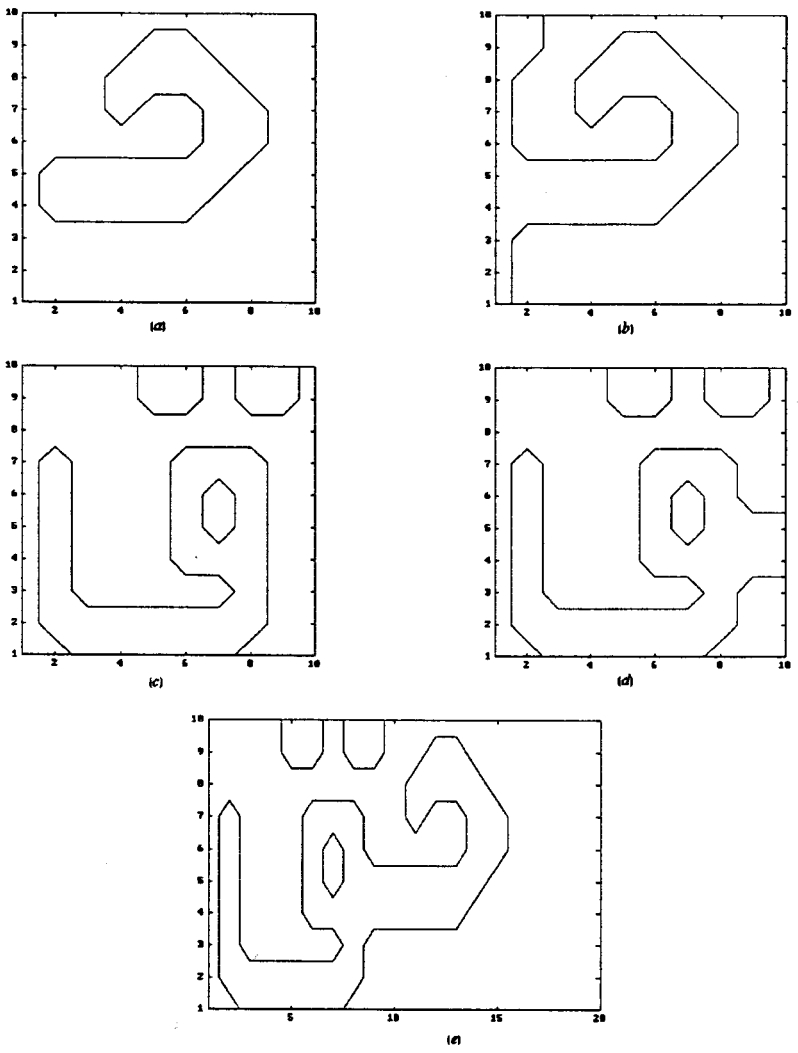
Fig. 3: Output values of hamming net for different iterations, when the input is the noisy version of the character *Che*

In the second part of the experiments, the word "*Haghe*" is used to show the use of the Hamming net for character separation. Figure (5) depicts the contour plot of the word and its constituents characters. The word consists of two characters which are approximately separated as described earlier. It is seen that each character may contain part of its adjacent character, Figures (5-b and 5-d). The neural net effectively classifies the connected forms as their original characters , Figures (5-a and 5-c).



a) Node threshold, b) Connection weights

Fig. 4: Parameters of hamming neural net after training



a) Isolated beginning *He*, b) Connected beginning *He*, c) Isolated end *Ghaaf*, d) Connected end *Ghaaf*, e) The word "*Haghe*"

Fig. 5: The word "Haghe" and its component He and Ghaaj

CONCLUSION

A new method based on Hamming neutral nets have been presented for recognition of Persian cursive text. Special features of the Persian script have been discussed in detail. The major problem of cursive character separation have been addressed. Novel properties of the Hamming Neural network have been utilized to implement an efficient OCR system for Persian (Farsi) script. Experimental results are presented to illustrate the usefulness of the proposed approach.

REFERENCES

1. **Blckwell, K. T., T. P. Vogl, S .D. Hyman, G .S. Barbour, and D.L.Alkon,** 1992. A New Approach to Handwritten Character Recognition. *Pattern Recog.*, (25) 6, 655-666.

2. **Goraine, H., M. Usher, and S. Al-Emari,** 1992. Off-Line Arabic Character Recognition. *IEEE Comput. Mag.*, 7, 71-74.

3. **Mahmoud, S. A., I. Abu-Haiba, and R.J. Green,** 1991. Skeletonization of Arabic Characters using Clustering-Based Skeletonization Algorithm. *Pattern Recog.*, (24) 5, 453-466.

4. **Widrow, B., R. Winter, and R. Baxter,** 1988. Layered Neural Nets for Pattern Recognition. *IEEE Trans. Acous. Speech Sig. Proc.*, (36) 7, 1109-1118.

5. **Almuallim, H., and S. Yamaguchi,** 1987. A Method of Recognition of Arabic Cursive Handwriting. *IEEE Trans. Pattern. Anal. Machine Intel.*, (9) 5, 715-722.

6. **Al-Yousefi, H., and S.S. Upda,** 1992. Recognition of Arabic Characters. *IEEE Trans. Pattern Anal. Machine Intel.*, (14) 8, 853-857.

7. **Fukushima, K., and N. Wake,** 1991. Handwritten Alphanumeric Character Recognition by the Neocognitron. *Neural Network*, (2) 3, 355-365.

8. **Parhami, B., and M. Taraghi,** 1981. Automatic Recognition of Printed Farsi Texts. *Pattern Recog.*, (14) 6, 395-403.

9. **Pavlidis, T., and F. Ali,** 1975. Computer Recognition of Handwritten Numerals by Polygonal Approximations. *IEEE Trans. Syst. Man Cybern.*, (5) 6, 610-614.

10. **Widrow, B., and M.A. Lehr,** 1990. 30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Back-propagation. *Proc. IEEE*, (78) 9, 1415-1441.

11. **Hopfield, J. J.,** 1984. Neural Networks and Physical Systems with Emergent Collective Computational Properties like those of Two-state Neurons. *Proc. National Acad. Sci.*, (81) 5, 3088-3092.

12. **Lippman, R. P.,** 1987. An Introduction to Computing with Neural Nets. *IEEE Acous. Speech Sig. Proc. Magazine*, 4, 4-22.