OCR For Printed Urdu Script Using Feed Forward Neural Network

Inam Shamsher, Zaheer Ahmad, Jehanzeb Khan Orakzai, and Awais Adnan

***Abstract*—**This paper deals with an Optical Character Recognition system for printed Urdu, a popular Pakistani/Indian script and is the third largest understandable language in the world, especially in the subcontinent but fewer efforts are made to make it understandable to computers. Lot of work has been done in the field of literature and Islamic studies in Urdu, which has to be computerized. In the proposed system individual characters are recognized using our own proposed method/ algorithms. The feature detection methods are simple and robust. Supervised learning is used to train the feed forward neural network. A prototype of the system has been tested on printed Urdu characters and currently achieves 98.3% character level accuracy on average .Although the system is script/ language independent but we have designed it for Urdu characters only.

***Keywords*—**Algorithm, Feed Forward Neural Networks, Supervised learning, Pattern Matching.

I. Introduction

A. Definition of Character

Character is the basic building block of any language that is used to build different structures of a language. Characters are the alphabets and the structures are the words, strings, and sentences etc. [1]

B. Optical Character Recognition

Optical character recognition (OCR) is the process of converting an image of text, such as a scanned paper document or electronic fax file, into computer-editable text. The text in an image is not editable: the letters/characters are made of tiny dots (pixels) that together form a picture of text. During OCR, the software analyzes an image and converts the pictures of the characters to editable text based on the patterns of the pixels in the image. After OCR, you can export the converted text and use it with a variety of word-processing, page layout and spreadsheet applications. OCR also enables screen readers and refreshable Braille displays to read the text contained in images.

C. Scope of Study

The scope of this project is to build a system, that automatically recognize the characters of Urdu language input to the system, and later on they may be used for different purposes.

Authors are with Center for Computing, Institute of Management Sciences, Peshawar, Pakistan (e-mail: inamshamsher1@yahoo.co.in, ahmad.zaheer@yahoo.com , janzeb@yahoo.com, awaisadnan@gmail.com).

D. Objective

Since in practice there are very few projects of this type used: for Urdu characters recognition, the primary objective is to develop a recognition system that efficiently recognizes Urdu characters utilizing minimum processor time.

II. Corpora

For our experimentation, we collected a corpora consisting of two sets of images (and associated transcriptions): computer generated, i.e. synthetic, images and real-world images consisting of scans of commonly available hardcopy documents (See Fig. 1).

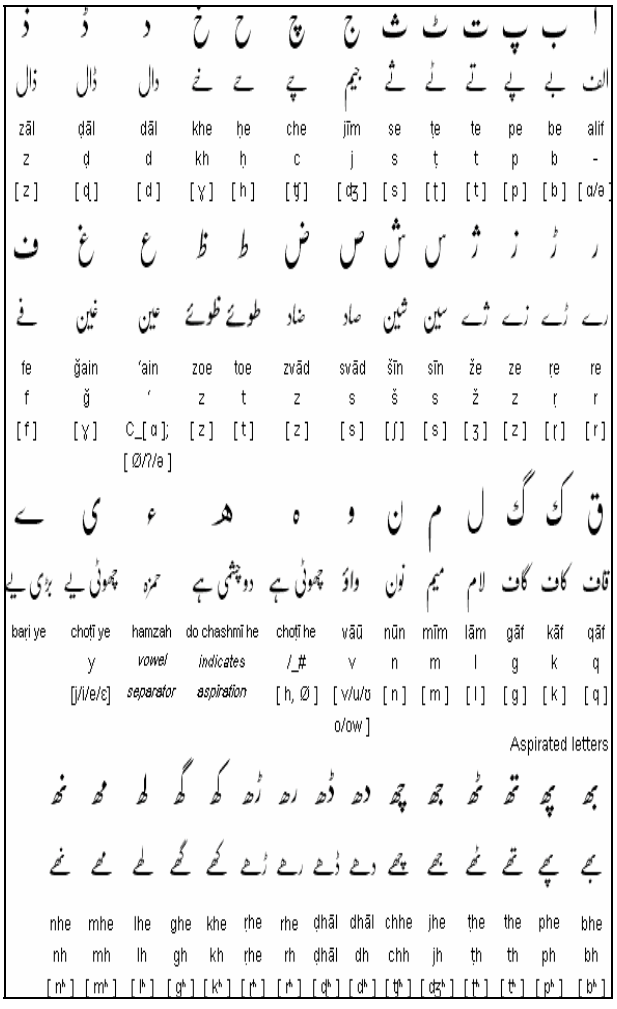

Fig. 1(a) Urdu characters

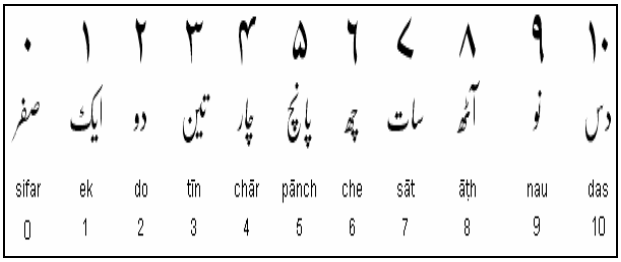

Fig. 1(b) Urdu Numerals

III. Technical Overview

In the proposed system, the document image is captured using a flatbed scanner and passed through training, and testing modules. These modules have been developed by combining conventional and newly proposed techniques. Supervised learning has been used to train the Feed Forward Neural Networks. [4, 5, 6, 9] Next, individual characters are recognized by our proposed method.

IV. Methodology

A. Network Formation

The MLP Network[3,12] implemented for the purpose of this project is composed of 3 layers, one input, one hidden and one output layer. The input layer constitutes of 150 neurons which receive pixels, binary data from a 10x15 symbol pixel matrix. The size of this matrix was decided taking into consideration the average height and width of character image that can be mapped without introducing any significant pixel noise. The hidden layer constitutes of 250 neurons whose number is decided on the basis of optimal results on a trial and error basis. The output layer is composed of 16 neurons corresponding to the 16-bits of Unicode encoding. To initialize the weights a random function was used to assign an initial random number which lies between two preset integers named **± weight bias**. The weight bias is selected from trial and error observation to correspond to average weights for quick convergence (See Fig. 2).

B. Symbol Image Detection

The process of image analysis to detect character symbols by examining pixels is the core part of input set preparation in both the training and testing phase. Symbolic extents are recognized out of an input image file based on the color value of individual pixels, which for the limits of this project is assumed to be either black **RGB(255,0,0,0)** or white **RGB(255,255,255,255)**. The input images are assumed to be in bitmap form of any resolution which can be mapped to an internal bitmap object in the Microsoft Visual Studio (.Net) environment.

The procedure also assumes the input image is composed of only characters and any other type of bounding object like a boarder line is not present. It also assumes that the size of the .bmp and font will not vary and all character lies in a single line. The procedure for analyzing images to detect characters is listed in the following algorithms:

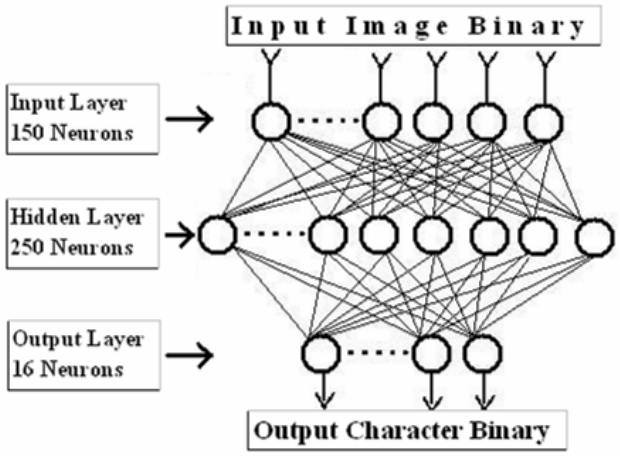

Fig. 2 Implemented MLP Network

C. Determining Character/Feature Extraction

All the characters are detected [13] and pixels are copied to a matrix in two passes only. In first pass, left, right and top (3 extreme points) of all characters are detected and in second pass bottom (extreme) is discovered.

1) Algorithm

1. start at left top of the picture[.bmp]
2. scan up to image height on the same x-component
   a. if black pixel is detected register x as left of the character, and y as top, Increment x, y
   b. if not continue to the next pixel
3. Scan the image(in the same character space), if y> top , update top
4. If y is equal to height register x as right of character. Increment Number of Characters.
5. Repeat step 1 to 4 till x is equal to image width.
6. Using left, top and right of each character scan character for bottom.

D. Training

Once the network has been initialized and the training input space prepared the network is ready to be trained. Some issues that need to be addressed upon training the network are:

- How complex are the patterns for which we train the network? Complex patterns are usually characterized by feature overlap and high data size.
- What should be used for the values of:
  - Learning rate
  - Sigmoid slope
  - Weight bias

Most common activation functions are the logarithmic and hyperbolic tangent sigmoid functions. The project used the **Hyperbolic tangent function:** $(2 / (1+e^{-\lambda x}))-1$ **and derivatives:** $f'(x)=f(x)(1-f(x))$

- How many Iterations (Epochs) are needed to train the network for a given number of input sets?
- What error threshold value must be used to compare against in order to prematurely stop iterations if the need arises?

For the purpose of this project the parameters used are:

- Learning rate = 150
- Sigmoid Slope = 0.026(for Urdu Characters)
- Weight bias = 30 (determined by trial and error)
- Number of Epochs = 300 (Maximum)
- Mean error threshold value = 0.0002 (determined by trial and error)

1. Algorithm

The training routine implemented the following basic algorithm

1. Form network according to the specified topology parameters
2. Initialize weights with random values within the specified ± weight bias value.[7]
3. load trainer set files (both input image and desired output text)
4. analyze input image and map all detected symbols into linear arrays
5. read desired output text from file and convert each character to a binary Unicode value to store separately
6. for each character :
    a. calculate the output of the feed forward network
    b. compare with the desired output corresponding to the symbol and compute error
    c. back propagate error across each link to adjust the weights
7. move to the next character and repeat step 6 until all characters are visited
8. compute the average error of all characters
9. repeat steps 6 and 8 until the specified number of epochs
    a. Is error threshold reached? If so abort iteration
    b. If not continue iteration

E. Testing

The testing phase of the implementation is simple and straightforward. Since the program is coded into modular parts the same routines that were used to load, analyze and compute network parameters of input vectors in the training phase can be reused in the testing phase as well. The basic steps in testing input images for characters can be summarized as follows:

1. Algorithm

- load image file
- analyze image for characters
- for each character
    - analyze and process symbol image to map into an input vector
    - feed input vector to network and compute output
    - convert the Unicode binary output to the corresponding character and render to a text box

V. Result and Discussion

The network has been trained and tested for Ariel font type in the Urdu alphabet set. Since the implementation of the software is open and the program code is scalable, the inclusion of more number of fonts like Pak-nastaleeq (Microsoft Beta-1) is easily implementable.

Our system identifies individual character with an accuracy of 98.3%

The necessary steps are preparing the sequence of input symbol images in a single image file (*.bmp [bitmap] extension), typing the corresponding characters in a text file (*.utc [Urdu trainer character] extension). The application will provide a file opener dialog for the user to locate the *. utc text file and *.bmp file. The software is tested in 72pt font size but it can be converted to any font size very easily.

VI. Future Direction

The Urdu character recognition system that is developed is only able to recognize the single/isolated Urdu character .Further research is needed to develop a system that recognize the connected/joined characters of Urdu, Arabic and other languages having the same properties.

VII. Conclusion

We have presented our new approach to text segmentation and text recognition for Urdu characters. Our proposed character recognition algorithms operate on input image and efficiently recognize the individual characters. More work is needed to have a system that also recognize the compound/ joined characters of Urdu script.

Acknowledgment

All glory is to ALLAH Almighty, whose blessing has always been a source of encouragement, patience and understanding for us, who gave us the ability to complete this difficult task successfully.

We greatly acknowledge the supervision of Mr. Awais Adnan, who was always very kind to extend their valuable guidance during this project. He was always there to help us find our way out of both major and minor problems.

In the end greatly thanking our parents for their incessant commitment to provide us with all the possible facilities throughout our academic career, which has made all this possible.

REFERENCES

[1] Y. LeCun, B. Boaer, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Handwritten zip code recognition with multilayer networks," *International Conference on Pattern* Recognition, 1990, pp. 35-44.

[2] K. Fukushlma, T. Imagawa, and E. Ashida, "Character recognition with selective attention ," 1991 *International Joint Conference on Neural Networks* (I), pp. 593-598.

[3] K. Fukushima and N. Wake,. "Handwritten alphanumeric character recogmtlon by the neocognitron," *IEEE 11-mw.* on *Neurral Networks,* Vol. 2, No. 3, May 1991, pp. 355-365.

[4] W. H. Joerding and J. L. Meador, "Encoding a priori information in feedforward networks," *Neural Networks,* Vol. *4, No. 6,* December 1991, pp. 847-856.

[5] J. S. N. Jean and J. Wang, 'Weight smoothing to improve network generalization," to appear in *IEEE tins. On Neural Networks.*

[6] J. Wang and J. S. N. Jean, "Multirexolution neural work for omni font character recognition, "submitted to 1999 *IEEE International Conference on Neural Networks.*

[7] A. Rajavelu, M. T. Muaavi, and M. V. Shirvaikar, "A neural network approach to character recognition," *Neuml Networks,* Vol. 2, No. 5, 1989, pp. 387-389.

[8] Y. Hayashi, M. Sakata, T. Nakao, T. Ohno, and S. Ohhashi, "Alphanumeric character recognition using a connectionist model with the pocket algorithm," 1989 *International Joint Conference on Neural Networks* (II), pp. 606.

[9] S. Kahan, T. Pavlidis, and H. S. Baird, "On the recognition of printed characters of any font and size," *IEEE I%ans. on Pattern Analysis and Ma*chine Intelligence, Vol. PAMI-9, No. 2, March 1987, pp. 274288.

[10] C. Wu, J. Wang, and W. Wu, 'A shunting multilayer perception network for confusing/composite pattern recognition," Pattern *Recognition,* Vol. 24, No. 11, 1991, pp. 1093-1103.

[11] M. Maier, 'Separating characters in scripted documents," 1986 International *Conference on Pattern* Recognition, pp. 1056-1058.

[12] S. Harmalkar and R. M. K. Sinha, "Integrating word level knowledge in text recognition, "1990 In*ternational* Conference on *Pattern* Recognition, pp. 758-760.

[13] R. G. Casey and G. Nagy, "Recursive segmentation and classification of composite character pattern," 1982 *International Joint Conference on Pa*ttern *Recognition,* pp. 1023-1026.

