OCR for Gujarati Numeral using Neural Network

[1]Manthan Khopkar
[1]M.tech Information Technology (4th sem)
[1]L.D.College Of Engineering, Gujarat Technological University, Ahmedabad, Gujarat, INDIA

Abstract— This papers functions within to reduce individuality popularity (OCR) program for hand-written Gujarati research. One can find so much of work for Indian own native different languages like Hindi, Gujarati, Tamil, Bengali, Malayalam, Gurumukhi etc., but Gujarati is a vocabulary for which hardly any work is traceable especially for hand-written individuals. Here in this work a nerve program is provided for Gujarati hand-written research popularity. This paper deals with an optical character recognition (OCR) system for handwritten Gujarati numbers. A several break up food ahead nerve program is suggested for variation of research. The functions of Gujarati research are abstracted by four different details of research. Reduction and skew- changes are also done for preprocessing of hand-written research before their variation. This work has purchased approximately 81% of performance for Gujarati hand-written numerals.

Key Words: Algorithm, Neural Networks, Supervised learning, Pattern Matching.

I. INTRODUCTION

A. *Introduction of numerals*

Numeral is the basic building block of any language that is used to build different structures of a language for calculation. Numerals are the numbers and the structures are the numbers group, natural Numbers, and operation etc.

B. *Optical Character Recognition*

Optical character recognition (OCR) is the process of converting an image of text or numerals, 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/numbers 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 numbers of Gujarati language input to the system, and later on they may be used for different purposes.

D. *Objective*

Since in practice there are very few projects of this type used: for Gujarati characters recognition, the primary objective is to develop a recognition system that efficiently recognizes Gujarati numbers 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.1and Fig. 2).

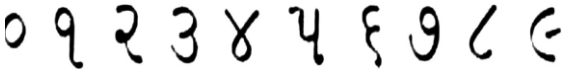

Fig. 1 : Gujarati digits 0–9.

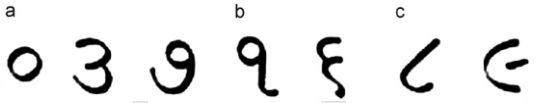

Fig. 2: Gujarati confusing sets of numerals

III. TECHNICAL OVERVIEW

In the proposed system, the document image of numerals 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. Prepositions (unless the title begins with such a word). Leave two 12-point blank lines after the title.

A. *Network Formation*

The FFBP Network[3,4] implemented for the purpose of this project is composed of 3 layers, one input, one hidden and one output layer. Earlier, 2000 Gujarati numeral samples from 200 Different writers were collected. Writers were provided with a plain A4 sheet and each writer was asked to write Gujarati numerals from 0-9 at one time [9]. Recently, we have again collected 2000 Gujarati numerals by 40 different writers. In this study the dataset size of 4000 Gujarati numerals is used. The database is totally unconstrained and has been created for validating the recognition system. Please note that the previous dataset is also included in the present dataset. The collected documents are scanned using the HP-scan jet 5400c at 300 dpi, which is usually a low noise and good quality image. The digitized images are stored as binary Images in the BMP format. A sample of Gujarati, handwritten numerals from the dataset is shown in Fig. 4. The preprocessing stage involves noise reduction, slant correction, size normalization and thinning. Among these, size normalization and thinning

are very important. The image size normalization is required, as the size of the numeral varies from person to person and even with the same person from time to time. The input numeral image is normalized to size 50×50 after finding the bounding box of each handwritten numeral image.

Thinning provides a tremendous reduction in data size; it extracts the shape information of the characters. Thinning is the process of reducing the thickness of each line of pattern to just a single pixel connectivity pattern. Thus, the reduced pattern is known as the skeleton and is close to the medial axis, which preserves the topology of the image. We have used the morphology-based thinning algorithm for better symbol representation. Detailed information about the thinning algorithm is available in [11]. Figure 5 shows the steps involved in our method as far as preprocessing is considered.

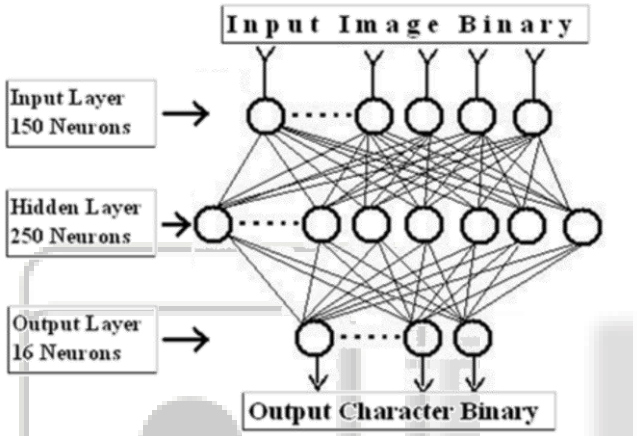

Fig. 3: Implemented FFBP Network

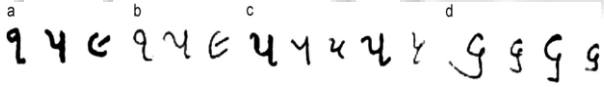

Fig. 4: Sample Handwritten Gujarati Numerals

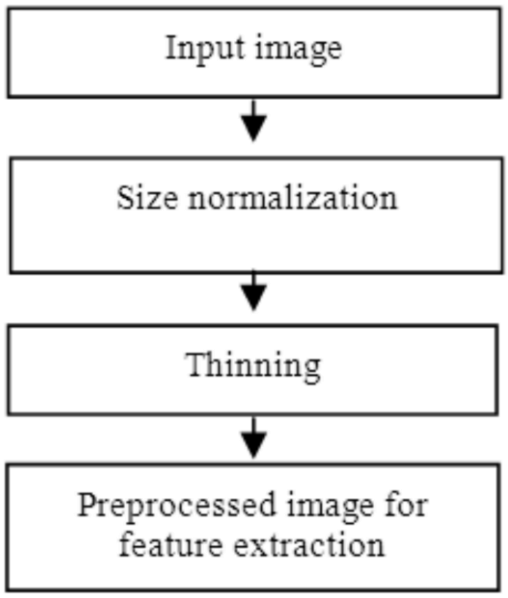

Fig.5: Preprocessing of the input numeral image

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 border 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

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:
  1) Learning rate
  2) Sigmoid slope
  3) Weight bias

Most common activation functions are the logarithmic and hyperbolic tangent sigmoid functions. The project used the *Hyperbolic tangent function:* (2 / (1+e-λx))-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:
  1) Learning rate = 150
  2) Sigmoid Slope = 0.026(for Gujarati Characters)
  3) Weight bias = 30 (determined by trial and error)

4) Number of Epochs = 300 (Maximum)
5) 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 converts 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

IV. RESULT AND DISCUSSION

The network has been trained and tested for Ariel font type in the Gujarati alphabet set. Since the implementation of the software is open and the program code is scalable, the inclusion of more number of fonts like shruti is easily implementable. Our system identifies individual character with an accuracy of 81.5% 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 [Gujarati 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. It can be easily seen in fig 4. Various result are trained in matlab neural network tool kit.

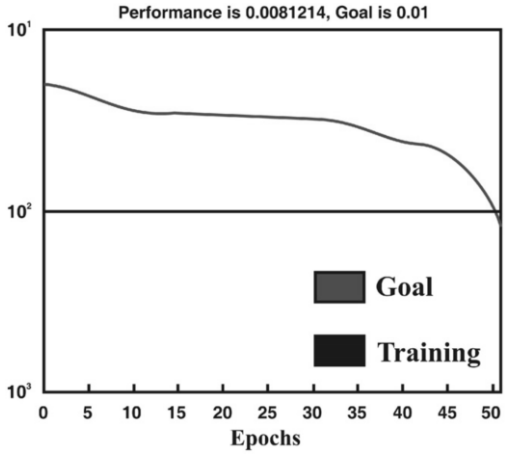

Fig.6 : Result of training in FFBP Network

V. FUTURE DIRECTION

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

VI. CONCLUSION

We have presented our new approach to zone based segmentation and numeral recognition for Gujarati numbers. 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 Gujarati script also work to be needed to get better accuracy upto 90 %.

ACKNOWLEDGMENT

All glory is to Almighty, whose blessing has always been a source of encouragement, patience and understanding for us, who gave us ability to review on this area. We greatly acknowledge the supervision of *Dr. J. S. Shah (Principal, Government Engineering college, Patan, Gujarat)* 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] U. Garain, B.B. Chaudhuri, Segmentation of touching characters in printed Devnagari and Bangla Scripts using fuzzy multifactorial analysis, IEEE Transactions on Systems, Man and Cybernetics, Part C 32 (4) (2002) 449–459.
[9] B.B. Chaudhhuri, U. Pal, M. Mitra, Automatic recognition of printed Oriya script, Saadhanaa 27 (1) (2002) 23–34.
[10] B. Chakravarthy, T. Ravi, S.M. Kumar, A. Negi. On developing high accuracy OCR systems for Telugu and other Indian scripts, in: Proceedings of Language Engineering Conference, 2002, pp. 18–23.
[11] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, "Digital Image Processing using MATLAB," Pearson Education. Dorling Kindersley, South Asia, 2004.