pISSN 1976-913X
eISSN 2092-805X

J Inf Process Syst, Vol.9, No.1, March 2013 http://dx.doi.org/10.3745/JIPS.2013.9.1.117

Optical Character Recognition for Hindi Language Using a Neural-network Approach

Divakar Yadav*, Sonia Sánchez-Cuadrado** and Jorge Morato**

Abstract—Hindi is the most widely spoken language in India, with more than 300 million speakers. As there is no separation between the characters of texts written in Hindi as there is in English, the Optical Character Recognition (OCR) systems developed for the Hindi language carry a very poor recognition rate. In this paper we propose an OCR for printed Hindi text in Devanagari script, using Artificial Neural Network (ANN), which improves its efficiency. One of the major reasons for the poor recognition rate is error in character segmentation. The presence of touching characters in the scanned documents further complicates the segmentation process, creating a major problem when designing an effective character segmentation technique. Preprocessing, character segmentation, feature extraction, and finally, classification and recognition are the major steps which are followed by a general OCR.

The preprocessing tasks considered in the paper are conversion of gray scaled images to binary images, image rectification, and segmentation of the document´s textual contents into paragraphs, lines, words, and then at the level of basic symbols. The basic symbols, obtained as the fundamental unit from the segmentation process, are recognized by the neural classifier.

In this work, three feature extraction techniques-: histogram of projection based on mean distance, histogram of projection based on pixel value, and vertical zero crossing, have been used to improve the rate of recognition. These feature extraction techniques are powerful enough to extract features of even distorted characters/symbols. For development of the neural classifier, a back-propagation neural network with two hidden layers is used. The classifier is trained and tested for printed Hindi texts. A performance of approximately 90% correct recognition rate is achieved.

Keywords—OCR, Pre-processing, Segmentation, Feature Vector, Classification, Artificial Neural Network (ANN)

1. INTRODUCTION

The introduction is covered into three sub-sections. The first defines the OCR and its basic applications, the second is about OCR in general, and the third is about Devanagari script, the mother script of the Hindi language.

Manuscript received December 17, 2012; accepted January 17, 2013.
Corresponding Author: Divakar Yadav
* Department of Computer Science & Engineering, Jaypee Institute of Information Technology, Noida, India (divakar.yadav@jiit.ac.in)
** Department of Computer Science, Carlos III University, Leganes, Madrid, Spain (ssanchec@gmail.com, jorge.morato@gmail.com)

Copyright ⓒ 2013 KIPS

1.1 What is Optical Character Recognition (Ocr)?

Optical Character Recognition (OCR) is a process of converting printed or handwritten scanned documents into ASCII characters that a computer can recognize [1]. In other words, automatic text recognition using OCR is the process of converting an image of textual documents into its digital textual equivalent. The advantage is that the textual material can be edited, which otherwise is not possible in scanned documents in which these are image files. The document image itself can be either machine-printed or handwritten, or a combination of the two. Computer systems equipped with such an OCR system improve the speed of input operation, decrease some possible human errors and enable compact storage, fast retrieval and other file manipulations. The range of applications includes postal code recognition, automatic data entry into a large administrative system, banking, automatic cartography and, when interfaced with a voice synthesizer, reading devices for the visually handicapped.

1.2 Overview of Ocr

For certain language scripts (e.g. Roman script), it is not difficult nowadays to develop an OCR system that recognizes well-shaped and well-spaced characters with an accuracy of 99% or above. However, it is still challenging to design a system that can maintain such high recognition accuracy, regardless of the quality of the input document and character font style variation.

In this work, our concern is Devanagari script, which is the script for Hindi, the national language of India and also for other languages like Sanskrit, Marathi and Nepali. The script is used by more than 300 million people across the globe. The algorithms which perform well for other scripts can be applied for Devanagari only after extensive preprocessing which makes simple adaptation ineffective. Therefore, it was necessary to do the research independently for Devanagari script. To date, many works on Devanagari script OCR have been reported [5-18, 26-30]. However, only a few of them have considered real-life printed Hindi text consisting of character fusions and combined with a noisy environment.

Segmentation and classification are the two primary phases of a text recognition system. The segmentation process extracts recognition units from the text, which is usually a character. The classification process computes certain features for each segmented character and then they are assigned to a class that may be the true class (correct recognition), the wrong class (substitution error), or an unknown class (rejection error).

Accuracy, flexibility and speed are the main features that characterize a good OCR system. Performance of most OCR systems developed for Hindi, have been constrained by dependence on the font, size and orientation. The recognition rate of these algorithms depends on the choice of features. Most of the existing algorithms involve extensive processing on the image before the features are extracted that results in increased computational time.

In this work, we applied a neural network-based approach for recognizing Hindi script. Neural network based Hindi OCR effectively reduces the image processing time while maintaining efficiency and versatility. The parallel computational capability of neural networks ensures a high speed of recognition. Neural network approaches have been used for character recognition [2, 3, 5, 19, 20, 31-33], but a complete system that encompasses all the features of a practical OCR system is yet to be realized. The key factors involved in the implementation are an optimal selection of features that categorically define the details of the characters, the number of features, and low image processing. We identified three techniques for computing feature vectors for each

character, histogram for projection based on mean distance calculation, histogram for projection base on pixel value, and vertical zero crossing. Details about these feature computation techniques are discussed later in the paper.

The preprocessing step is used to adapt the input image into a normalized shape. It consists in turn of steps such as noise cleaning, skew detection and correction, and line, word and character segmentation. These pre-processing steps transform the document image into a set of normalized isolated Hindi characters that are trained and/or tested with ANNs. The ANN is trained to learn all Hindi characters, and after successful training, its weights/feature is used for the purpose of testing. After obtaining the recognition results from the ANN, some post-processing techniques are applied to improve the recognition rate.

We prepared the database having Hindi alphabets for the purpose of training the classifier. The individual alphabet characters in the database were saved in .bmp format of a fixed size of 48x57 pixels. Our database consists of 77 different symbols/characters in five different Hindi fonts. So in total there were 385 characters in the data set that was used for training.

An OCR system should usually perform the following tasks:

a) Text Digitization
b) Gray Tone to Two Tone Conversion
c) Noise clearing
d) Text Block Identification
e) Skew Correction
f) Line and Word Detection
g) Character Segmentation

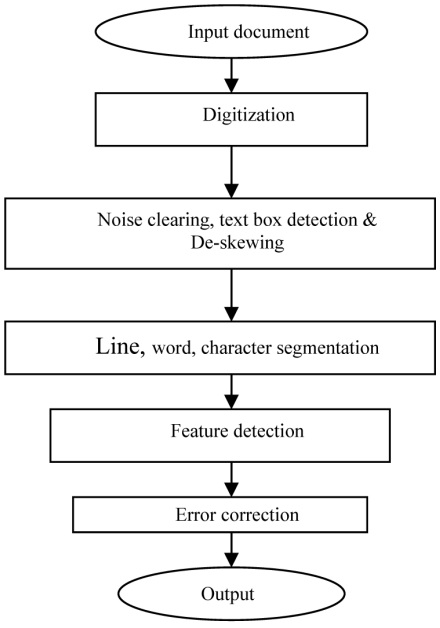


Fig. 1. Flow control for Hindi OCR System

h) Features Extraction
i) Character Recognition
j) Error Correction

In real life applications, OCR software accepts the document image as input and produces a text file as output. So we are required not only to develop the recognition process, but also to develop the pre-processing and post-processing parts. The flow control of processes for Hindi OCR is shown in Fig. 1.

1.3 Devanagari Script from Ocr Point of View:

Devanagari script is a logical composition of its constituent symbols in two dimensions. It is an alphabetic script. The writing style in Devanagari is horizontal, from left to right, and characters do not have any uppercase/lowercase. It has 12 vowels and 36 simple consonants. Besides the consonants and the vowels, other constituent symbols in Devanagari are the set of vowel modifiers called Matra (placed to the left, right, above, or at the bottom of a character or conjunct), pure consonants (also called half letters) which when combined with other consonants yield conjuncts.

Some of the Hindi vowels and consonants are shown in Fig. 2(a) and Fig. 2(d) respectively. In English as well as in Hindi, the vowels are used in two ways: (i) they are used to produce their own sounds; the vowels shown in Fig. 2(a) are used for this purpose in Devanagari. (ii) They are used to modify the sound of a consonant; in order to do this, an appropriate modifier from the symbols shown in Fig. 2(b) is attached to the consonants in an appropriate manner.

Some of the modifiers attached with the first consonant letter 'ka' of Hindi are shown in Fig.

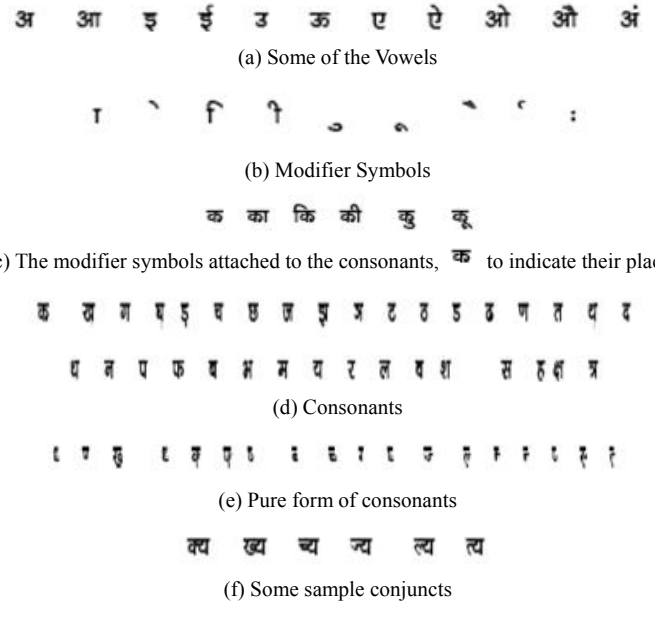


Fig. 2. Characters and Symbols of Devanagari Script

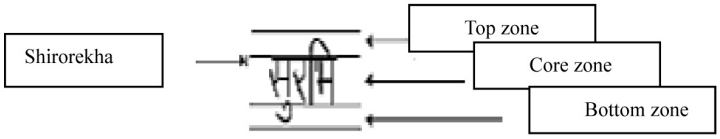


Fig. 3. Three Zones of Devanagari Script

2(c). A visual inspection of the figure reveals that some modifier symbols are placed next to the consonants (core modifier), some are placed above the consonants (top modifier) and some are placed below (lower modifier) the consonants. Some of the modifiers contain a core modifier and a top modifier; the core modifier is placed before or next to the consonant and the top modifier is placed above the core modifier.

In Devanagari script, there exists a pure form called a half character for almost every consonant. A consonant in pure form always touches the next character occurring in the script and thus yielding conjuncts, touching characters or fused characters. Fig. 2(f) shows some of the conjuncts formed by writing pure form consonants followed by consonants.

A horizontal line is drawn on the top of all characters of a word and is known as the header line, or shirorekha. Words written in Devanagari can be visualized in the form of three horizontal strips/zones: a core strip, a top strip and a bottom strip. The middle and upper zones are separated by the header line. Fig. 3 shows a word that consists of three characters, one top modifier, and one lower modifier. The three strips and the header line have been marked.

2. RELATED WORK

A brief review of the previous research work done in the field of OCR for Hindi is discussed in following paragraphs.

In the previous section it was mentioned that Devanagari is the script for Hindi, the national language of India. Further we also discussed that apart for Hindi, it is also script for Sanskrit, Marathi, Nepali and many more Indian languages. Devanagari is derived from ancient Brahmi script through various transformations. During the past 40 years, substantial research effort has been devoted to OCR for the Devanagari script [4-5], [8-16], [26-31]. But no complete OCR for Devanagari is yet available which works in a noisy environment. One of the major causes for the poor recognition rate in an OCR system is error in character segmentation [4]. Existence of touching characters in the scanned documents is another major problem faced while designing an effective character segmentation procedure. In [4], a new technique is presented for identification and segmentation of touching characters. The technique is based on fuzzy multi factorial analysis. A predictive algorithm is developed for effectively selecting possible cut columns for segmenting the touching characters. The proposed method has been applied to printed documents in Devanagari and Bangla.

The technique for character segmentation treats characters touching each other as a single character and leads to the failure in the character recognition phase. Faxed documents, photocopies, old books, newspapers, etc., contain a considerable number of touching characters. A module for automatic separation of touching characters is essential for successful OCR of such documents.

3. SEGMENTATION

Segmentation is one of the most important phases in OCR development. It directly affects the efficiency of any OCR. So a good segmentation technique can increase the performance of OCR. Segmentation is process of extracting the basic constituent symbols of the script, which are individual characters. It is necessary to segment the script at character level, as classifier works at character level only. Preprocessing in form of text digitization and skew correction is performed before applying segmentation.

3.1 Text Digitization and Preprocessing Technique

During implementation of the work text digitization is done using a flatbed scanner having a resolution between 100 and 600 dpi. The digitized images are usually in gray tone, and for a clear document, a simple histogram-based thresholding approach is sufficient for converting them to two-tone (black & white) images. The histogram of gray values located between the peaks is a good choice for the threshold.

When a document page is fed to the scanner, it may be skewed by a few degrees. The pages of thick books create more problems since the scanned image may both be distorted and skewed. There exists a wide variety of skew detection algorithms based on projection profile. Among these, Hough transform [21] is one of the most important methods usually used in OCR for most languages, but Chaudhari et al. [19] proposed a new approach that employs the headline of Devanagari script. For this work we applied the latter approach due to its suitability for Devanagari script as this script contains a header line (shirorekha) with each of its words.

3.2 Line, Word and Character Segmentation

The OCR system developed, initially detects the text blocks from the input provided in form of scanned images and thereafter it automatically segments the detected blocks into lines, lines into words and finally words into characters by using the methods discussed below.

3.2.1 Segmentation of Lines

A horizontal scanning method is applied for segmenting the text paragraphs into lines. While performing the segmentation to extract the lines from the text blocks, it performs horizontal scanning starting from the top of the scanned document till it locates the last row containing all white pixels, before a black pixel row is encountered. It continues the scanning further, till it

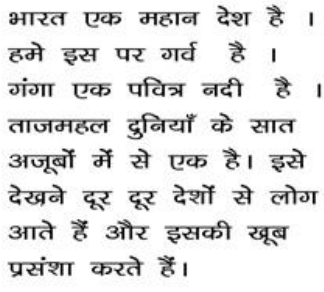

भारत एक महान देश है ।
हमे इस पर गर्व है ।
गंगा एक पवित्र नदी है ।
ताजमहल दुनियाँ के सात
अजूबों में से एक है। इसे
देखने दूर दूर देशों से लोग
आते हैं और इसकी खूब
प्रसंशा करते हैं।

Fig. 4. Text block of Devanagari Script

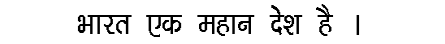
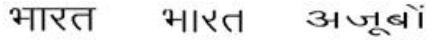


Fig. 5. Segmented line from text block of Fig. 4

locates the first row containing all white pixels, just after the end of last row of black pixels. This determines a line, and is eventually extracted. This whole process is repeated on the entire text page to segment all the text lines present in that particular page/paragraph. Fig. 5 shows the output for first line when this algorithm is applied on the input text block shown in Fig. 4

3.2.2 Segmentation of Words

After segmenting the lines it segments the individual words embedded in each line. To perform this operation a vertical scanning method is applied. The vertical scanning is applied to the width of the line only. One output example of this phase is shown in Fig. 6.



(a) (b) (c)

Fig. 6. (a) Segmented word, (b) and (c) word after removing Shirorekha

3.2.3 Segmentation of Characters

Once the words are segmented from text lines using the method described here, a further segmentation process is applied to achieve the individual characters out of the segmented words. Before segmenting words at character level, the header line or shirorekha is identified and removed. This process is done by finding the rows with the maximum number of black pixels in a word. Here we applied a heuristic approach to locate the shirorekha as sometimes, not all the rows of shirorekha contain same number of black pixels. In this paper we have taken black pixel difference counts among rows of shirorekha as 10, though it is not the same value for all other scripts. This difference count for Hindi was decided after an exhaustive analysis of rows of many shirorekhas. After locating the shirorekha, it is removed, i.e. converted into white pixels, shown in Fig 6 (b) and (c). Once the shirorekha is properly removed, the word is divided into three horizontal zones known as upper, middle and lower zones. Individual characters are separated from each zone by applying vertical scanning. The vertical scanning is performed for the width of zones and length of the word only. As stated earlier, only modifiers are present in upper and lower zones. So before performing vertical scanning in these zones, it is checked whether any modifier exists or not. This is required because in several cases words may contain only middle zone or middle and upper zone or middle and lower zone only as shown in Fig. 7

(a) (b) (c) (d)

Fig. 7. (a) word with only middle zone, (b) word with all three zones, (c) word with middle and upper zone (d) word with middle and lower zone

4. Proposed Method for Feature Extraction and Classification

In the current section we discuss the methods for recognition of the basic symbols. For the classification of symbols in Hindi fonts, we developed artificial neural network based classifiers. The classification is performed on the basis of the extracted features of individual symbols/characters. To ensure the correct classification of each symbol, we applied three different methods to compute their features.

4.1 Feature Extraction Methods

Feature extraction is another major step in developing a classifier system. One should study in detail and should become well aware of the characteristics of the scripts to be classified before selecting the feature detection methods. This helps in computing the features that discriminate the character sample in better way. For initial classification of characters, we considered following three feature extraction methods:

1. Histogram of projection based on mean distance
2. Histogram of projection based on pixel value.
3. Vertical zero crossing

The final feature vector for each character is computed by concatenating the sub vectors computed separately from the above three methods. The computed feature vector is used as input for the neural network. Since each character has its own unique feature vector, thus it helps them to be classified correctly.

4.1.1 Histogram of Projection Based on Mean Distance

Using this method the feature vector for the segmented symbols/characters is computed as follows:

Steps:

1. Find the x and y coordinate for every black pixel of the character and add them all to compute X and Y.
2. Divide both X and Y separately by total number of black pixels of the characters to compute a central point.
3. Compute the distances of each black pixel from the central point to complete a matrix and normalize it.

4.1.2 Histogram of Projection Based on Pixel Value

Using this method the feature vector for the segmented symbols/characters is computed as follows:

Steps:

1. Find the outer boundary of character.
2. Scan it horizontally as well as vertically to find the number of black pixels.
3. Compute vertical and horizontal projection counts.
4. Normalize it.

4.1.3 Vertical Zero Crossing

The image of a character is treated as an array of pixels where white pixels are expressed by 1 and black pixels by 0. Under the Vertical Zero Crossing the entire pixel array is traversed column wise and the number of transitions from black to white pixels is recorded for each column.

4.2 Classifier Tool

As stated earlier, an Artificial Neural Networks (ANN) approach was used to develop the classifier. ANN approach has been used for classification and recognition on varying problems from a long time. It is a computational model widely used in situations where the problem is complex and data is subject to statistical variation. The training and recognition phase of the neural network has been performed using a conventional back propagation algorithm with two hidden layers. The architecture of a neural network determines how a neural network transfers its input into output. This transfer can be viewed as a computation.

4.2.1 Scaling

Since ANN receives input image in fixed size, a normalization process is required to change each segmented character to the same size. There are two major issues that we should consider at this stage, first the suitable fixed size of input matrix for ANN, and second a method to transform from the original character to the normalized one. We conducted a study on the characteristics of printed Hindi characters (up-to size 16), and found that the matrix size 48x57 is enough to fit any character. An example is given in Fig. 8, where we scale an image of an arbitrary size to a fixed size. As can be seen, the scaling process does not introduce any distortions or loss of information.

Original 22x30 image Scaled 48x57 image

Fig. 8. Normalization from random size to fixed size

4.2.2 Training

A neural network maps a set of input to a set of output. This nonlinear mapping can be thought of as a multidimensional mapping surface. The objective of learning is to mold the mapping surface according to a desired response. We used four layered perceptrons, and two hidden layers, one as input and one as output layer. In the input layer we used 169 neurons, equal to the number of elements in the feature vector computed for each Hindi character. The first hidden layer consists of 250 neurons and the second hidden layer, 200. Finally in the output layer there were 77 neurons, equal to the number of different Hindi characters used in the training set. Selection of number of neurons in each layer is purely based on practical observation. As mentioned earlier we adopted back propagation for learning. This is the most popular learning algorithm in multilayer network [3]. It employs descent method in weight space. Thresholding is done by using '*sigmoid'* and '*tansig'* function. For the output of the first hidden layer we used the 'sigmoid' function and at the second hidden layer we used *'tansig'* function. We trained our neural network for fives set of Hindi fonts (Kruti10, Kruti025, Kruti040, Kruti060 and DevLys020) each having 77 characters. The diagrammatic representation of the ANN is shown in

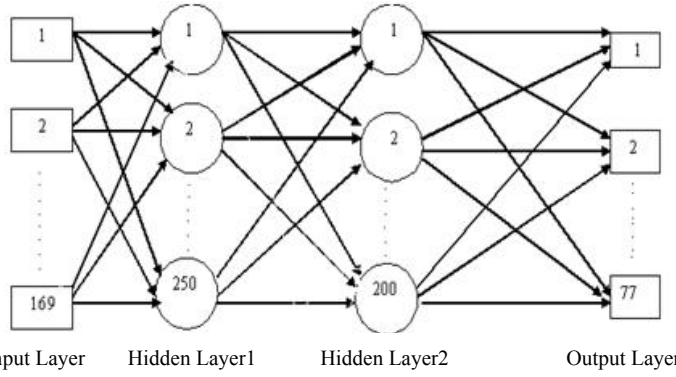


Fig. 9. A Neural Network with 4 layers, 169 neurons in I/P layer, 250 and 200 neurons in hidden layers and 77 neurons in O/P Layer

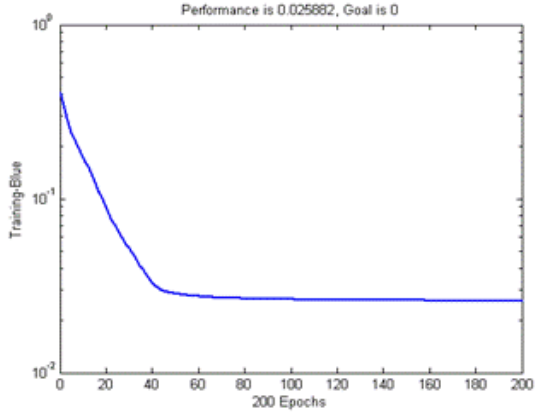


Fig. 10. ANN learning curve using learning rate 0.9 for first 200 epochs

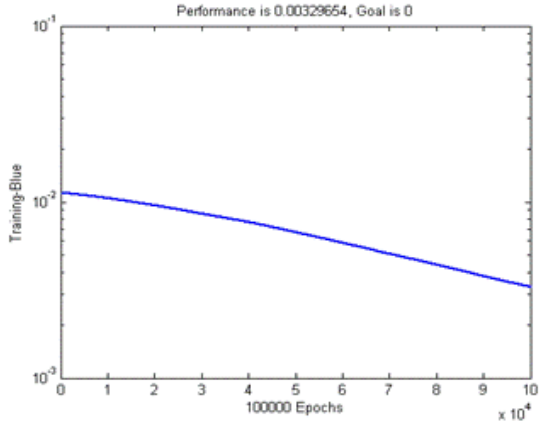


Fig. 11. ANN learning curve using learning rate 0.9 for 100000 epochs (from 50001epochs to 150000 epochs)

Table 1. Actual values obtained for all 77 symbols while training the alphabet 'Ka'

0.9550	0.0011	0.0000	0.0578	0.0027	0.0025	0.0250	0.0012	0.0002	0.0004	0.0116
0.0001	0.0000	0.0328	0.0005	0.0032	0.0026	0.0001	0.0004	0.0020	0.0000	0.0000
0.0064	0.0000	0.0011	0.0122	0.0012	0.0012	0.0000	0.0000	0.0000	0.0000	0.0001
0.0013	0.0100	0.0008	0.0016	0.0382	0.0049	0.0013	0.0002	0.0004	0.0044	0.0022
0.0004	0.1386	0.0001	0.0083	0.0065	0.0219	0.0143	0.0000	0.0135	0.0002	0.0007
0.0022	0.0005	0.0022	0.0036	0.0183	0.0022	0.0003	0.0016	0.0037	0.0001	0.0129
0.0001	0.0089	0.0065	0.0008	0.0000	0.0028	0.0004	0.0018	0.0003	0.0074	0.0045

Table 2. Values after normalization

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0																		

Fig. 9. The training was done for .5 million epochs in total to achieve the training goal. The learning curves for the first 200 epochs, and later between 50001 to 150000, are shown in Fig. 10 and Fig. 11 respectively. The X axis represents the number of epochs and the Y axis represents the training performance in terms of errors where zero errors means the training goal is achieved.

Table 1 shows the actual values obtained for training set sample 'Ka' after completion of the training. As can be seen, the first value is close to 1 while all the others are < 0.06. The values after normalization are shown in Table 2. This agrees quite well with the expected values for the character. Similar sets of values were obtained for other letters too.

4.2.3 Efficiency

In neural networks, time required to converge is highly dependent on the learning parameter [20]. If it is too high, then the training does not converge. On the other hand, if learning parameter is assumed very low, then it will take excessive time to converge. Adjusting the learning parameter dynamically during the process of training has given better result. Starting with a higher value, we decreased the parameter as it approached convergence. We started the training with an initial learning parameter value 0.9 and later reduced it to 0.1 after approximately .2 million epochs when we observed that the learning curve became somewhat stable. As mentioned in the previous paragraph, we trained the neural network for five fonts having 77 characters each. So there are 385 total characters in the training set. Each character is represented by a feature vector of length 169. So, the amount of training data is very huge. After training for several hours and for thousands of epochs it was possible to train 383 out of 385 characters, i.e. two, ड and रू , both of DevLys020 font were not trained. So we can see that efficiency of training of our neural network is near to complete, i.e. 100%.

5. RESULTS ANALYSIS

We tested our classifier on various printed Hindi documents and gathered the results. In fact the testing was done at two levels: individual letter level and paragraph level. The character

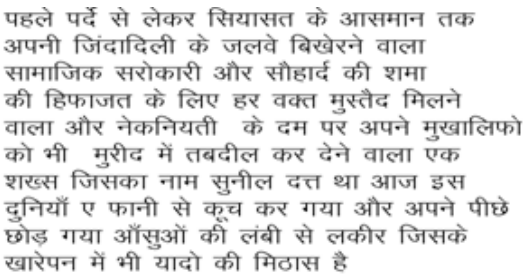


Fig. 12. Scanned Devanagari Script as Input

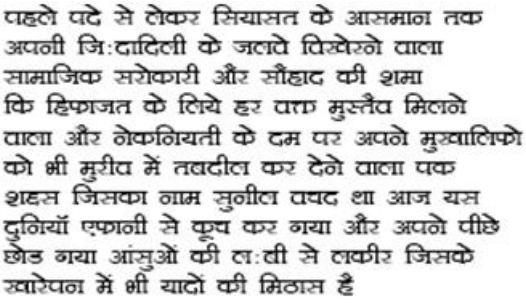


Fig. 13. OCR Output

level testing was performed on approximately 1000 individual characters. Paragraph level testing was performed on approx 15 paragraphs of different Hindi fonts consisting of approximately 650 words. The character level performance was excellent with a correct recognition rate of 98.5%. However, at paragraph level the performance dropped and an average accuracy of ap-

Table 3. Result comparison of OCR Systems for Printed Devanagari Characters

Method	Feature	Classifier	Data Set Size	Accuracy (%)
Govindraju et al. [32]	Gradient	Neural Networks	4,506	84
Kompalli et al. [33]	GSC	Neural Network	32,413	84.77
Bansal et al. [14]	Statistical Structural	Statistical Knowledge Sources	Unspecified	87
Huanfeng Ma et al. [34]	Statistical Structural	Hausdorff image comparison	2,727	88.24
Natrajan et al. [40]	Derivatives	HMM	21,727	88.24
Bansal et al. [8]	Filters	Five filters	Unspecified	93
Dhurandhar et al. [38]	Contours	Interpolation	546	93.03
Kompalli et al. [37]	GSC	K-nearest neighbor	9,297	95
Kompalli et al. [35]	SFSA	Stochastic finite state automaton	10,606	96
Dhingra et al. [39]	Gabor	MCE	30,000	98.5
Divakar et al.	Mean distance, pixel value and vertical zero crossing	Neural Network	1000	98.5

Table 4. Result comparison of OCR Systems for Printed Devanagari words

Method	Feature	Classifier	Data Set Size	Accuracy (%)
Govindraju et al. [32]	Gradient	Neural Networks	4,506	53
Kompalli et al. [37]	GSC	K-nearest neighbor	1,882	58.51
Kompalli et al. [33]	GSC	Neural Network	14,353	61.8
Ma et al. [34]	Statistical Structural	Hausdorff image comparison	2,727	66.78
Kompalli et al. [35]	SFSA	Stochastic finite state automaton	10,606	87
Divakar et al.	Mean distance, pixel value and vertical zero crossing	Neural Network	650	90

proximately 90% was achieved. One such input sample paragraph used for testing the performance of the classifier is shown in Fig. 12. Results obtained therefrom are shown in Fig. 13. It can be observed that the recognition rate is higher for individual than for continuous characters. The discrepancy between the performance for isolated and continuous characters may have happened due to errors in the segmentation procedure. The performance of our classifier both at printed character level and at word level for Devanagari was compared with that of many other existing methods [30] and is shown in Table 3 and Table 4 respectively.

5.1 Robustness of Classifier

The performance of the classifier at alphabet level is shown in Table 5. To test the robustness of the classifier we gradually increased the distortion in the input character images and observed the recognized characters thereof. Table 5 shows the original character, the distorted character, provided as an input to the classifier, and the output character recognized by the classifier.

By observing the table, we can say that the classifier is robust enough and is able to correctly classify even highly distorted characters too. The only instances of misclassifications are at the level when even a human expert would guess incorrectly. For example, for the Hindi letter ‘ka’ continuous distortion was incorporated, and the classification results observed thereafter are

Table 5. Robustness of the classifier

Original Character	Distorted Character	Character Recognized
क	क	क
क	क	क
क	क	क
क	क	क
क	क	क
क	क	क
क	क	क
क	क	क
क	क	फ

Original Character	Distorted Character	Character Recognized
ಅ	ಅ	ಅ
ಅ	ಅ	ಅ
ಅ	ಅ	ಅ
ಅ	ಅ	ಅ
ಅ	ಅ	ಎ
ಕ್ಷ	ಕ್ಷ	ಕ್ಷ
ಕ್ಷ	ಕ್ಷ	ಕ್ಷ
ಕ್ಷ	ಕ್ಷ	ಕ್ಷ
ಕ್ಷ	ಕ್ಷ	ಕ್ಷ
ಕ್ಷ	ಕ್ಷ	೯
ಂ	ಂ	ಂ
ಂ	ಂ	ಂ
ಂ	ಂ	ಂ
ಂ	ಂ	೧
ಘ	ಘ	ಘ
ಘ	ಘ	ಘ
ಘ	ಘ	ಘ
ಘ	ಧ	ಧ

shown from rows number 1 to 9 of Table 5. Until row number 8 the classifier was perfectly able to recognize it as ‘ka’, but it failed at row number 9, recognizing it as ‘ph’. It is almost certain that even if human experts are provided with the same distorted letter, as shown in row number 9 under the column ‘Distorted Character’ then most of the time they will recognize it as ‘ph’ too. Similar results were obtained for many other letters too such as ‘chha’ and ‘dwa’, as shown at rows number 14 and 27 respectively of Table 5. So this all proves the robustness of the classifier developed using the ANN approach.

5.2 Result Details of Feature Extraction

From the results presented in above table we can conclude that the error at classification level is negligible. This is, indeed, due to good selection of the features. To validate the point, we present some detailed results of the feature values obtained with respect to some characters (Fig. 14 to Fig. 28). As mentioned, we used a set of robust and efficient feature extraction techniques. These feature extraction techniques are not affected by the type of fonts or their sizes.

Similarly we can draw feature diagrams for every character. The above figures show that each character has unique feature values. It indicates the discriminating power of the feature extraction techniques, and is the primary reason for the robustness of the recognition.

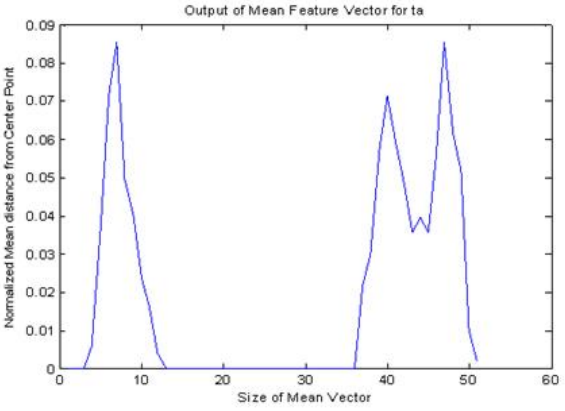


Fig. 14. Feature vector for character 'Ta' using mean distance

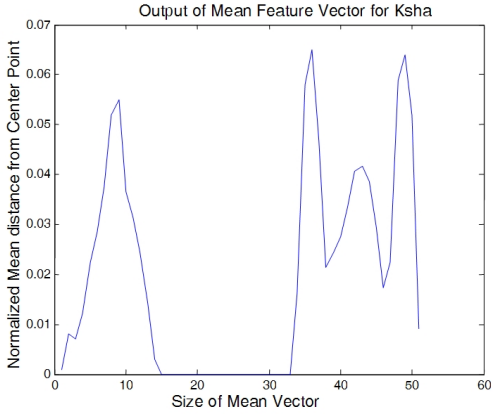


Fig. 15. Feature vector diagram for character 'Ksha' using mean distance

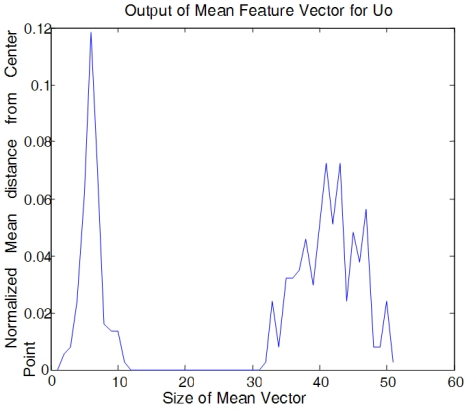


Fig. 16. Feature vector diagram for modifier 'Uo' using mean distance

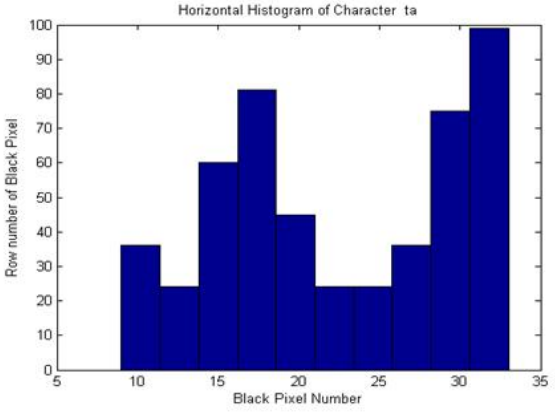


Fig. 17. Feature vector diagram for character 'Ta' using horizontal projection based on spatial position

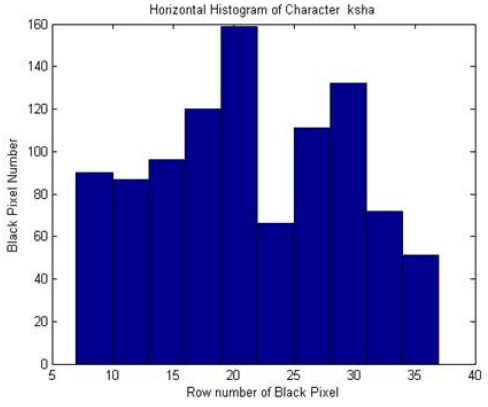


Fig. 18. Feature vector diagram for character 'Ksha' using horizontal projection based on spatial position

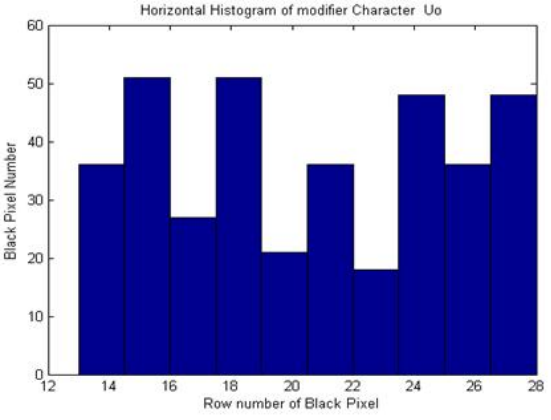


Fig. 19. Feature vector diagram for modifier 'Uo' using horizontal projection based on spatial position

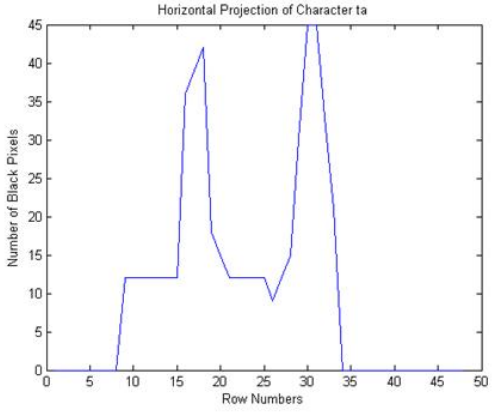


Fig. 20. Feature vector diagram for character 'Ta' using horizontal projection based on pixel value

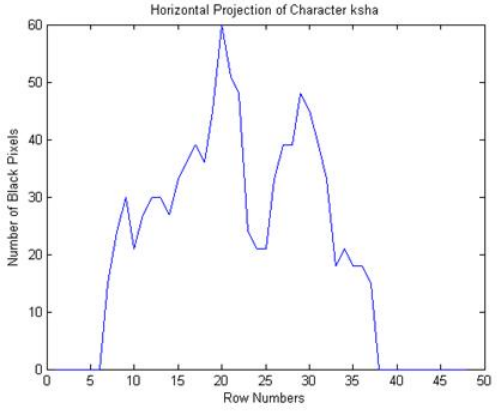


Fig. 21. Feature vector diagram for character 'Ksha' using horizontal projection based on pixel value

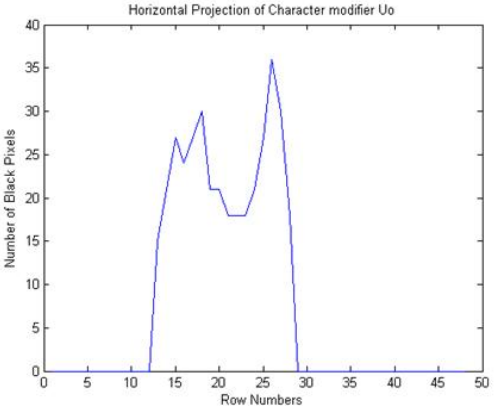


Fig. 22. Feature vector diagram for modifier ´Úo´ using horizontal projection based on pixel value

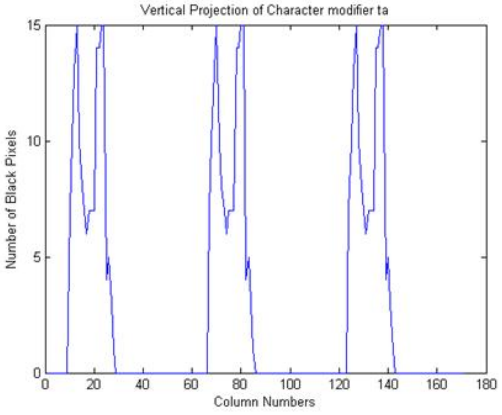


Fig. 23. Feature vector diagram for character 'Ta' using Vertical projection based on Pixel value

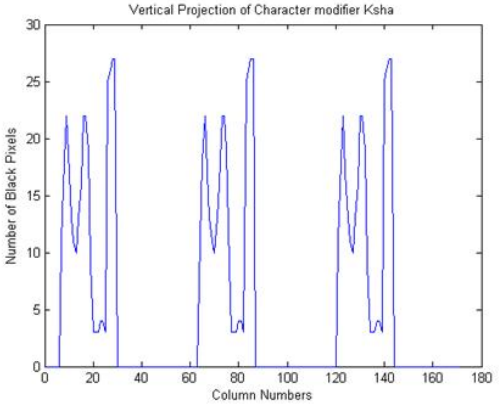


Fig. 24. Feature vector diagram for character 'Ksha' using vertical projection based on pixel value

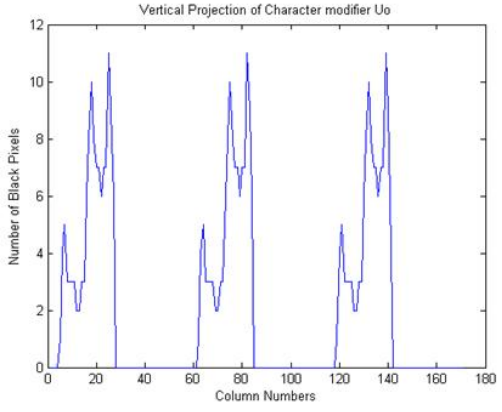


Fig. 25. Feature vector diagram for modifier ´Uo´ using vertical projection based on pixel value

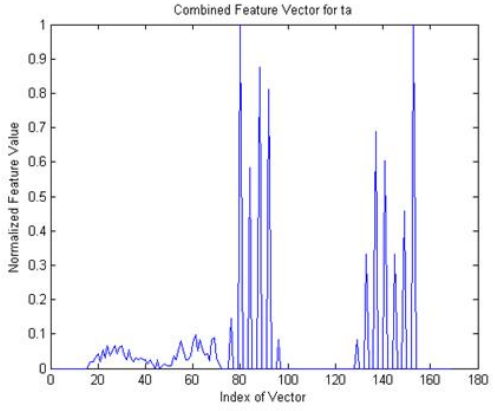


Fig. 26. Combined feature vector diagram for character 'Ta'

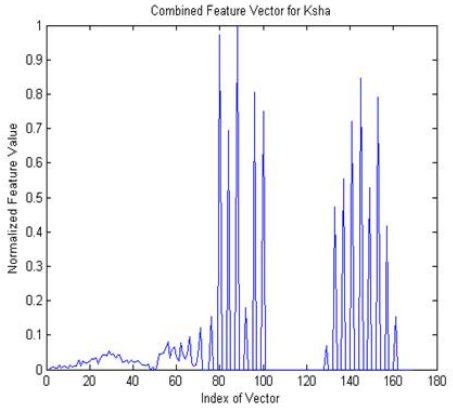


Fig. 27. Combined feature vector diagram for character 'Ksha'

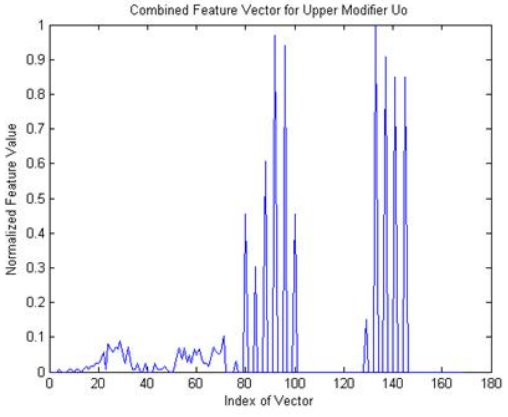


Fig. 28. Combined feature vector diagram for modifier 'Uo'

5.3 Segmentation Analysis

In the paper, we presented a complete method for segmentation of printed Hindi texts. A preliminary segmentation process extracts the header line and delineates the upper-strip from the rest of the character. This yields vertically separated character boxes that may be conjuncts, touching characters, characters with lower modifier attached to them, characters fused with rakar symbols (a special case of a ligature constituted by the adjunction of a consonant followed by a halanta to 'ra'), or a combination of these. In phase-I of the segmentation process, statistical information about these boxes is collected. In phase-II, this statistical information is used to select the boxes on which further segmentation is attempted. Besides separating the lower modifier from the core strip and identifying the rakar symbol within the core strip, the constituent character boxes in the conjuncts are also identified in the core strip. We extensively used the structural properties of the characters in the segmentation process. The presence and relative location of vertical line, and the nature of the constituent pure consonant forms in the conjunct, provide valuable clues about segmentation boundaries. Our segmentation approach is a hybrid approach, wherein we try to recognize the parts of the conjuncts that form part of a character class.

6. CONCLUSION AND FUTURE WORK

The experimental results illustrate that the artificial neural network (ANN) concept can be applied successfully to solve the Hindi optical character recognition problem. There are many variations of factors that affect the performance of the developed Hindi OCR software. We found that the input matrix of size 48x57 gives better results than other choices. The recognition rate of the OCR with the real Hindi document is quite high as shown in the results section. However, other kinds of preprocessing and neural network models may be tested for a better recognition rate in the future. The character segmentation method can be improved to handle larger variety of touching characters that occur often in images obtained from inferior-quality documents. The test set used in this experiment is of 77 characters of five different types of fonts. This can be tested for a greater number of fonts. The toughest phase in the experiment is getting a good set of characters for classification. This highlights the need for generation of a large ground-truthed set of characters of various resolutions so that more research can be performed for recognition of languages from Indian subcontinent. Also, the characters used for experiment were enclosed in the bounding region of a fixed size. Although this may not be always the case, attention is drawn toward the fact that mere shape based recognition will not always perform well. Different font families represent the same character differently and correlation between similar characters varies from font to font. This preliminary research helped us focus our attention on these matters so that issues for building robust character recognition can be studied.

The other future enhancement that can be incorporated in the work is to use a dictionary of words to correct the output [26]. Certainly this will improve the performance. Further speech synthesizer can be integrated with the OCR with the aim of making a system for reading aids to the blind. We can also implement the neural network method for classifying hand-written texts. In hand-written documents, the fragmentation of characters and the variation in shape of characters are considerably greater compared to printed documents. The higher levels can be used to provide clues for a hypothesization system, which learns from the text it recognizes. We have

not dealt with punctuation marks and numerals in this work. The set of the punctuation marks and the numerals need special treatment right from the point of preliminary segmentation of words into characters and symbols. For Devanagari script, the header line is removed from the character before an attempt is made to classify it. However, a numeral or punctuation mark cannot be dealt with the same manner. There is no header line to be removed, even though some such marks may have a horizontal line which resembles the header line. We implemented the work for the scripts with only Hindi characters. However, we can extend it to classify the document with characters of more than one script.

REFERENCES

[1] Mori, S. et. al.: Historical Review of OCR Research and Development. Proceeding IEEE, Vol.80, No.7, 1992, pp.1029-1058.

[2] Chaudhari, A.A., Ahmad, E.A. S., Hossain, S., Rahman, C.M.: OCR of Bangla Character Using Neural Network: A better approach. 2nd International Conference on Electrical Engineering (ICEE 2002.), khuln, Bangladesh, 2002.

[3] Mahmud, J.U.; Raihan, M.F., Rahman, C.M.: A complete OCR System for Continuous Bengali Characters. TENCON 2003, IEEE conference on convergent Technologies for Asia-Pacific Region Vol.4, 2003, pp.1372-1376.

[4] Garain, U., Chaudhuri, B. B.: Segmentation of Touching Character in Printed Devanagari and Bangla Script Using Fuzzy Multifactorial Analysis. IEEE Transaction on System, Man and Cybernetics -Part C: Applications and Reviews, Vol.32, No.4, 2002, pp.449-459.

[5] Jawahar, C.V., Pavan Kumar, M.N.S.S.K., Ravi Kiran, S.S.: A Bilingual OCR for Hindi-Telugu Documents and its Application. Document Analysis and Recognition. IEEE Proceedings Seventh International Conference on, Vol.1, 2003, pp.408-412.

[6] Lakshmi, C.V., Patvardhan, C.: A High Accuracy OCR System for printed Telugu text. Conference on Convergent Technology for Asia-pacific Region Vol.2, 2003, pp.725-729.

[7] Ashwin, T.V., Sastry, P.S.. A Font and size independent OCR for printed Kannad documents using support vector machines.

[8] Bansal, V., Sinha, R.M.K.: A Complete OCR for Printed Hindi Text in Devanagari Script. Sixth International Conference on Document Analysis and Recognition, IEEE publication, Seatle USA., 2001, pp.800-804.

[9] Bansal, V., Sinha, R.M.K.: Integrating Knowledge Source in Devanagari Text Recognition System. IEEE transaction on Systems, MAN and Cybernetics-Part A: System and Humans, Vol.30, No.4, 2000, pp.500-505.

[10] Bansal, V., Sinha, R.M.K.: On How to describe Shape of Devanagari Characters and Use them for Recognition. 5th International Conference on document Analysis and recognition (ICDAR'99), Bangalore, India, 1999.

[11] Bansal, V., Sinha, R.M.K.: A Devanagari OCR and A Brief Overview of OCR Research for Indian Script, PROC Symposium on Transaction support System (STRANS 2001), Kanpur, India, 2001.

[12] Setlur, S., Kompalli, S., Ramanaprasad, V., Govindaraju, V.: Creation of data resources and design of an evaluation test bed for Devanagari Script recognition. Research Issues in Data Engineering Multilingual Information Management. IEEE Proceeding 13th International Workshop on, 2003, pp.55-61.

[13] Bansal, V., Sinha, R.M.K.: Segmentation of touching and fused Devanagari characters. Pattern Recognit., Vol.35, 2002, pp.875-893.

[14] Bansal, V., Sinha, R.M.K.: Integrating knowledge sources in Devanagari text recognition. IEEE Trans. Syst. Man Cybern. A: Syst. Hum., Vol.30, No.4, 2000, pp.500-505.

[15] Chaudhuri, B. B., Pal, U.: An OCR System to Read Two Indian Language Script Bangla and Devanagari (Hindi. Document Analysis and recognition. IEEE Proceeding of the Fourth International Conference on, Vol.2, 1997, pp.1011-1015.
[16] Bansal, V., Sinha, R.M.K.: Designing a Front End OCR System for Indian Script for Machine Translation- A Case Study for Devanagari. Symposium on Machine Aids for Translation and Communication, New Delhi, India, 1996.
[17] Bansal, V., Sinha, R.M.K.: On Devanagari Document Processing. IEEE International Conference on Systems, Man and Cybernetics, Intelligent Systems for the 21st Century., Vol.2, pp.1621-1626, 1995.
[18] Pal, U., Chaudhuri, B.B.: Automatic Separation of Machine-printed and Hand-Written Text Lines. Document Analysis and Recognition. IEEE Proceedings of the Fifth International Conference on, 1999, pp.645-648.
[19] Chaudhuri, B.B., Pal, U.: Skew angle Detection of Digitized Indian script Documents. Pattern Analysis and Machine Intelligence, IEEE Transactions on Vol.19, Issue 2, 1997, pp.182-186.
[20] Faaborg, Alexander j.: Using Neural Network to Create an Adaptive Character Recognition System.
[21] Hinds, S.C., Fisher, J.L., D'Amato, D.P.: A document skew detection method using run-length encoding and the Hough transform. Pattern Recognition, IEEE Proceedings., 10th International Conference on Vol.1, 1990, pp.464-468.
[22] Tellache, M., Sid-Ahmaed, M., Abaza, B.: Thinning algorithms for Arabic OCR. Communications, Computers and Signal Processing, 1993, IEEE Pacific Rim Conference on Vol.1, 1993, pp.248-251.
[23] Shimizu, M., Fukuda, H., Nakamura, G.: A thinning algorithm for digital figures of characters. Image Analysis and Interpretation, 2000. Proceedings. 4th IEEE Southwest Symposium, 2000, pp.83-87.
[24] Pal, U., Chaudhuri, B.B.: Automatic identification of English, Chinese, Arabic, Devanagari and Bangla script line. Document Analysis and Recognition. Proceedings. Sixth International Conference on, 2001, pp.790-794.
[25] Bansal, V., Sinha, R.M.K.: Partitioning and searching dictionary for correction of optically read Devanagari character strings. Document Analysis and Recognition. Proceedings of the Fifth International Conference on, 1999, pp.653-656.
[26] Aradhya, V.N. Manjunath, Kumar, G. Hemantha, Noushath, S.: Multilingual OCR system for South Indian scripts and English documents: An approach based on Fourier transform and principal component analysis. Engineering Applications of Artificial Intelligence 21, 2008, pp.658-668.
[27] Bhattacharya, U., Chaudhuri, B.B.: Handwritten Numeral Databases of Indian Scripts and Multistage Recognition of Mixed Numerals. IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.31, No.3, 2009, pp.444-457.
[28] Desai, Apurva A.: Gujarati hand written numeral optical character reorganization through neural network. Pattern Recognition 43, 2010, pp.2582-2589.
[29] Pal, U.; Roy, P. P., Tripathy, N., Josep, L.: Multi-oriented Bangla and Devanagari text recognition. Pattern Recognition 43, 2010, pp.4124-4136.
[30] Jayadevan, R., Kolhe, Satish R., Patil, Pradeep M., Pal, U.: Offline Recognition of Devanagari Script: A Survey. IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews, Vol.41, No.6, 2011, pp.782-796.
[31] Apurva A. Desai: Gujarati handwritten numeral optical character reorganization through neural network. Pattern Recognition, Vol.43, Issue 7, 2010, pp.2582-2589.
[32] Govindaraju,V., Khedekar, S., Kompalli, S., Farooq, F., Setlur, S., Vemulapati, R.: Tools for enabling digital access to multilingual Indian documents. in Proc. 1st Int. Workshop Document Image Anal. Libraries, 2004, pp.122-133.
[33] Kompalli, S., Nayak, S., Setlur, S., Govindaraju, V.: Challenges in OCR of Devanagari documents. In Proc. 8th Conf. Document Anal. Recognition, 2005, pp.1-5.
[34] Ma, H., Doermann, D.: Adaptive Hindi OCR using generalized Hausdorff image comparison. ACM

Trans. Asian Lang. Inf. Process., Vol.2, No.3, 2003, pp.193-218.

[35] Kompalli, S., Setlur, S., Govindaraju, V.: Devanagari OCR using a recognition driven segmentation framework and stochastic language models. Int. J. Document Anal. Recognit., Vol.12, 2009, pp.123-138.

[36] Kumar, S.: An analysis of irregularities in Devanagari script writing: A machine recognition perspective. International Journal of Computer Science Eng., Vol.2, No.2, 2010, pp.274-279.

[37] Kompalli, S., Setlur, S., Govindaraju, V.: Design and comparison of segmentation driven and recognition driven Devanagari OCR. In Proceeding 2nd Int. Conf. Document Image Anal. Libraries, 2006, pp.1-7.

[38] Dhurandhar, A., Shankarnarayanan, K., Jawale, R.: Robust pattern recognition scheme for Devanagari script. Comput. Intell. Security, Part I, Lecture Notes in Artificial Intelligence 3801, 2005, pp.1021-1026 .

[39] Dhingra, K. D., Sanyal, S., Sanyal, P. K.: A robust OCR for degraded documents. In Lecture Notes in Electrical Engineering. Huang et al., Eds. New York: Springer-Verlag, 2008, pp.497-509.

[40] Natarajan, P., MacRostie, E., Decerbo, M.: The BBN Byblos hindi OCR system. In Guide to OCR for Indic Scripts, V. Govindaraju and S. Setlur, Eds. New York: Springer-Verlag, 2009, pp.173-180.

Divakar Yadav

He received his PhD. degree in Computer Science and Engineering from Jaypee Institute of Information Technology, Noida, India in Feb 2010. He spent one year, from Oct 2011 to Oct 2012, in Carlos III University, Leganes, Madrid, Spain as a post doctoral fellow. He has published more than 25 research papers in reputed international/national journals and presented at conferences. His areas of interest are Information retrieval and soft computing.

Sonia Sánchez-Cuadrado

She is an Associate Professor in the Department of Computer Science at Carlos III University, Madrid, Spain. She received her PhD in Library Science and Digital Environment in 2007, designing a methodology for the automatic construction of knowledge organization systems and NLP. She has published many research papers in reputed international and national journals and presented at conferences. Her areas of interest are information retrieval and natural language processing.

Jorge Morato

He is an Associate Professor in the Department of Computer Science at Carlos III University, Madrid, Spain. He received his PhD in Library Science from Carlos III University in 1999. His current research interests include text mining, information extraction and pattern recognition, NLP, information retrieval, web positioning, and knowledge organization systems. He has published many research papers in reputed international and national journals and presented at conferences.