Autonomous Bilingual Character Recognition 
and Writer Identification

B. Aditya Vighnesh, B. Manikanta Yadav, Abhishek Kumar, M.V. Raghunadh
Department of Electronics and Communication, NIT Warangal

Abstract—Writer Recognition (WR) has been an active area of research in the past few decades and remains a challenging topic due to its difficulty in implementing writer individual variations. This paper describes a complete system for the recognition of off-line handwriting and the writer. The database creation, pre-processing, segmentation, representation and classification steps involved in the recognition of handwritten scanned text images have been detailed.

Index Terms—Handwriting Recognition (HWR), Intelligent Character Recognition (ICR), k-NN Classifier, Neural Networks, Optical character Recognition (OCR), Pre-processing, Segmentation, Writer Recognition (WR).

I. INTRODUCTION

The purpose of this paper is to develop an automatic system for person identification based on their handwriting alone. There are many methods of person identification such as smart cards, person identification number (PIN) and passwords. Regardless of the fact that these means provide a highly secure access to authorized users, they can be easily stolen, forgotten or cracked. Handwriting is a behavioral characteristic of a person that is very reliable and common; however they are often restricted to fixed text, like a signature. This paper tries to widen the scope of hand-writer recognition to any text of English or Hindi script.

About a decade ago Srijari et.al[1] proved that handwriting is unique to an individual and later Tomai et.al[2] employed eleven global macro-features (document-level) and micro-features (character-level) to identify writers, however, the macro-features extracted from words presented very low performance. Then Hiremuth [3] used texture features while Martin et.al [4] used features from characters, words and sentences from offline text for improving the accuracy. Joulia Chapran[5] also worked on biometric writer identification systems using both online and offline features. Besides writer recognition, handwriting recognition is also gaining momentum. S. Madhavnath et al. [6] used contour chain codes to recognize handwritten words while Anshul et al. [7] used neural nets for solving the classification problem. Combinations of classifiers such as SVM + k-NN [8] were also used with improved results. For the writer classification k-NN classifiers have shown good results as described by Caroline and Horst [9].

While these papers concentrate in recognizing English script, Pal [10] published a comprehensive comparative study of Devnagari handwritten character recognition.

Besides character and writer recognition our paper focuses on script analysis as well. Ghosh [11] writes a review of various script recognition techniques available today. In this paper, we develop an automatic system for recognition of handwritten English and Hindi characters in words and finally identify the writer of those words for the given script. We first segment the paragraph into lines, these lines into words and finally words into individual characters. These characters are then represented by features which have high discriminative qualities. Next, the writer dependent features are extracted from segmented sentences to recognize the writer.

The process flow is as follows: The development of the database is first discussed in Section II followed by Section III which deals with the preprocessing that includes binarization, normalization and noise removal. Segmentation is described in Section IV while Section V deals with the feature extraction process of the individual characters for both English and Hindi languages. This section also includes extracting writer dependent features. Section VII presents the results, while the conclusions and possible extensions to this system are discussed in Section VIII.

II. DATABASE

Like any other recognition engine, the WR system also requires a database of training and testing samples. Although many online databases like CEDAR and IAM are available, we opted to develop a database of our own from ground up to check if the system works well in a real-world scenario. Besides the self-developed database we also used the IAM Handwriting database [17] to test our results against a standard database.

First, we developed a template containing fields where the writers were asked to write English and Hindi words in both constrained and unconstrained manner as shown in Fig. 1&2. The constrained box for Hindi was based on Diya’s design [12] and was complicated when compared to that of English in order to accommodate the ascenders and descenders.

Fig 1 English Constrained and Unconstrained writing

The fields in the template were segmented using connected component analysis, i.e. the blob areas were measured and the blobs having the areas under a specific range were indexed while others were rejected. The indexed blobs were then cropped out of the image.
Following the segmentation stage the images are cleared of any salt and pepper noise, using median filters, and are stored into either the train or test directory depending on their indices.

The training data was used to develop and train the classifier which is explained in section VI. The process flow for testing is similar to training and begins with pre-processing.

III. PREPROCESSING

Post scanning the document undergoes a series of processing steps that are very crucial. Any errors in these steps will propagate throughout the system. The steps involved are:

A. Binarization
The scanned text is converted into a black and white or binary image. We adopted the Otsu’s method [13] for global thresholding.

B. Noise Removal
The image is then filtered using a median filter to remove any salt and pepper noise while preserving the edges.

C. Skew Detection and Correction
While scanning for the orientation, the document may have deformed. This is known as skew of the document. In order to remove skew the image is modeled as a rotation transformation, i.e. if the skew of the image is found then it may be rotated using rotation transformations to its original orientation.

We used image morphology operations to create large connected components and analyzed their orientations individually. The orientation of the maximum number of blobs was taken as the skew angle.

D. Slant Detection and Correction
In order to identify characters individually they have to be normalized. This step detects and removes any slant or angle in the characters. The slant is modeled mathematically as a horizontal shear transformation.

To detect slant we first found the vertical edges in the words in a particular line and found their slopes. The slant angle for each edge was computed and their average was considered as the slant angle for that line. An example of slant detection is shown in figure (3) which is an image from Davide et al’s presentation [14].

IV. SEGMENTATION

In order to recognize characters they have to be first isolated from the entire text and to do so first the lines have to be segmented followed by words and finally the characters.

A. Line Segmentation
The line segmentation is performed using the popular vertical histogram algorithm where the image area between the significant minima in the vertical histogram of the text image is considered to be occupied by a line. An example of line segmentation is shown in figure (4) which is an image from Davide et al’s presentation [14].

B. Word Segmentation
The word segmentation is performed by using a variation of the horizontal projection algorithm which originally assumes that the image area between the significant minima in the horizontal histogram of the text image is considered to be occupied by a word. This assumption does not hold good if a person writes disjoint letters to form the word. Hence, we developed our algorithm using the repeated image dilation process that forms connected components of the entire word even if the letters are disjoint. The basic horizontal projections method is then followed to segment the words.

C. Character Segmentation
Many algorithms have been devised to segment connected characters, however many of them are very complex. In order to reduce the complexity we have implemented a novel approach to the problem. Our method is similar to the segmentation algorithm using vertical projections however the steps before the actual segmentation are very different.
1) **Algorithm for English:**
1. Segment the connected components
2. Try recognizing each component
3. If recognition passes $\Rightarrow$ END
4. Else,
   a. Thin the image to single pixel thickness,
   b. Thicken the image using a special structuring element such that only vertical sections are thickened while the horizontal sections are untouched.
   c. Finally perform the closing operation, now the image is ready for histogram segmentation.
   d. Estimate the number of characters in the connected component based on its length.
   e. Find the cut points by setting the threshold for setting as ‘1’.
   f. Find the distance between the cut points. If the distance is less than the average width of a letter ignore the cut point and repeat this step till the width is large enough.
   g. Finally check if the number of cut points match with the estimated number of characters.
   h. Finally pass the segmented characters through the recognition engine and END.

![Example of novel character segmentation algorithm](image)

2) **Algorithm for Hindi:**
1. Detect Sirorekha and baselines for appropriate zoning in to ascender, middle and descender regions
2. Segment the image into the three regions.
3. In the middle region, remove the sirorekha (algorithm detailed next) and then perform horizontal histogram segmentation algorithm.

3) **Sirorekha Removal Algorithm:**
1. Thin the Hindi word image to single pixel thickness.
2. Normalize the image both horizontally and vertically.
3. Find the horizontal edges by using a Prewitt filter.
4. Remove any open pixel areas of less than 25 pixels.
5. Compute the vertical projection.
6. Find the peak in the histogram and verify that this peak lies in the first one-third of the image. If verified then,
   a. Remove the pixels of 5 lines around the line with the peak.
   b. Check for a peak in the top one-third and then and verify that it is greater than the 80% of the global peak value. If verified, remove the pixels of 5 lines around the line with the peak else, there exists no sirorekha.

![Example of Hindi Segmentation](image)

**V. FEATURE EXTRACTION**

An Intelligent Recognition engine is characterized by a block that extracts intelligent features which are useful for successful recognition. In literature many different kinds of features have been computed, some are – geometric, statistical, zoning, point features, etc. This paper deals with recognition of script, character and writer and the features extracted for each of them are different as explained below.

**A. Features for Script Recognition**
To distinguish between the Latin and Devnagari script the presence of sirorekha (headline) is detected. The algorithm to detect the sirorekha has already been detailed previously.

**B. Features of English Character Recognition**
The English ICR uses the following features –
1. Point features – Character image is normalized to [70x50] and then divided to [7x5] by checking the number of white pixels in every 10x10 pixel matrix.
2. Geometric Features – The number of loops, endpoints, junctions and dots in the character image are extracted.
3. Zone features – The presence of white pixels is detected in the top-left, middle-left, bottom-left, top-centre, middle-centre and bottom-centre.

![Example of zones](image)

**C. Features of Hindi Character Recognition**
The Hindi ICR uses the following features –
1. Fourier Descriptor – encode the shape by taking Fourier transform of the boundary. Inverse Fourier Transform
reconstructs the shape and the higher the frequency component the more accurate the reconstruction.

2. Geometric Features - The number of loops, endpoints, junctions and dots in the character image are extracted.

![a) b) c)](image)

Fig 8 a) Endpoints b) Junctions c) Loops

3. Statistical Features- the centroid of the character image is extracted. This feature is used as Hindi Characters are spread across the image area.

D. Features of Writer Recognition

The following features are computed over a handwritten sentence -
1. Distance between consecutive words.
2. Average Orientation of the words.
3. Average of the Form Factor of the words.
4. Average Roundness measure of the words
5. Average size of the words
6. Average word height
7. Average upper zone height
8. Average middle zone height
9. Average lower zone height
10. Slant angle
11. Skew angle

![Handwritten text](image)

Fig 9 Example of handwriting and its computed features

Feature Vector = \{18.85 3.96 0.74 17.17 269.77 102 26 32 44 88.79 1.2107\}

VI. CLASSIFIER

We used two classifiers namely the Probabilistic Neural Network and the Kth Nearest Neighbor to recognize the characters and writers respectively. A probabilistic neural network (PNN) is a feed-forward neural network, which is a combination of the Bayesian network and the Kernel Fisher discriminant analysis. PNN have been proved previously to have a good classification of pattern and offer many advantages over the MLP network.

In k-Nearest Neighbor classifier a Euclidean distance measure is used to calculate how close each member of the training feature vector is to the target feature vector that is being examined.

VII. RESULTS

The results and performance figures reported in this section were obtained by implementing the proposed model on MATLAB 2010a on a laptop having 1.2GHz Intel Core i7-640U processor and 4GB RAM.

A. Self-developed Database

Database Files: 18,000 test & train files
Number of Writers: 50 writers
Training Samples: 800 samples per writer
Test Samples: 158 samples per writer
2-Letter Words: 17 English + 43 Hindi per writer
3-Letter Words: 18 English + 12 Hindi per writer
4-Letter Words: 13 English + 2 Hindi per writer
5-Letter Words: 2 English per writer

As mentioned in section (II) we have also used the IAM database to standardize our results.

B. Line Segmentation

The accuracy of segmentation depends on the line spacing of the writer; the greater it is the cleaner the segmentation. With our database having 50 text paragraphs each with 16 lines the accuracy is found to be-

Line Segmentation Accuracy: 97%.

C. Word Segmentation

The accuracy of segmentation depends on the character spacing in a word, the lesser it is the cleaner the word segmentation. With our database having 640 sentences each with an average of 7 words the accuracy is found to be-

Word Segmentation Accuracy: 92%

The algorithm was also tested over the IAM Handwriting Database and accuracy over the range of 1000 sentences was Accuracy over 1000 IAM database sentences: 95%

D. Character Segmentation

The character segmentation depends highly on the handwriting of the writer. The conjoint characters hamper the accuracy.

1. English Character Segmentation

Accuracy over 4480 words: 87%

Accuracy over 7000 IAM database words: 83%

This is better than projection based segmentation which fails for extending characters.

2. Hindi Character Segmentation

Accuracy over 2000 words: 90%

E. Pattern Classification

The time taken for character recognition through PNN model is 1.2 seconds and writer recognition through KNN model is 0.5 seconds.

1. Script Recognition

<table>
<thead>
<tr>
<th>Table I Script Recognition Results</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Script</strong></td>
</tr>
<tr>
<td>Latin (English)</td>
</tr>
<tr>
<td>Devnagari (Hindi)</td>
</tr>
</tbody>
</table>

2. Character Recognition

<table>
<thead>
<tr>
<th>Table II ICR Results</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Language</strong></td>
</tr>
<tr>
<td>English</td>
</tr>
<tr>
<td>Hindi</td>
</tr>
</tbody>
</table>
3. Writer Recognition

Table III Writer Recognition Results

<table>
<thead>
<tr>
<th>Language</th>
<th>Training Writers</th>
<th>Test Writers</th>
<th>kNN Accuracy</th>
<th>PNN Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>80</td>
<td>60</td>
<td>96%</td>
<td>87%</td>
</tr>
<tr>
<td>Hindi</td>
<td>30</td>
<td>20</td>
<td>87%</td>
<td>70%</td>
</tr>
</tbody>
</table>

For every writer 7 sentences were trained and tested.

Table IV False Acceptance and Rejection Ratios

<table>
<thead>
<tr>
<th>Language</th>
<th>FAR</th>
<th>FRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>10%</td>
<td>3%</td>
</tr>
<tr>
<td>Hindi</td>
<td>15%</td>
<td>4%</td>
</tr>
</tbody>
</table>

The writer recognition is more accurate for English script as the English handwriting for the writers was more uniform and continuous than Hindi. Moreover, the low accuracy of the PNN in WR is due to a very small sample space.

VIII. CONCLUSION

This paper therefore, highlights the development of a Writer Recognition (WR) system that can also recognize the handwritten characters in English and Hindi. Further, the paper also discusses the steps involved in developing your own database and also a novel algorithm for character segmentation. Keeping in mind the applications of WR system a lot of improvements can be made to better the accuracy of the Hindi writer Recognition and also reduce the overall FAR and FRR.

ACKNOWLEDGMENT

We would like to acknowledge our Mentor, Sri M.V. Raghunad, without whose help and constant guidance this paper would not have been successfully completed.

REFERENCES


AUTHOR BIOGRAPHY

B. Aditya Vighnesh, Senior Year Bachelor of Technology degree student in Electronics and Communication Engineering in National Institute of Technology, Warangal & IEEE Student Member.

B. Manikanta Yadav, Senior Year Bachelor of Technology degree student in Electronics and Communication Engineering in National Institute of Technology, Warangal.
Abhishek Kumar, Senior Year Bachelor of Technology degree student in Electronics and Communication Engineering in National Institute of Technology, Warangal

M.V. Raghunadh, Associate Professor, Department of Electronics and Communication Engineering, National Institute of Technology, Warangal