ISSN No. 0976-5697

Volume 4, No. 6, May 2013 (Special Issue)

International Journal of Advanced Research in Computer Science

REVIEW ARTICAL

Available Online at www.ijarcs.info

Script Identification of Text Words from Indian Document through Discriminating Features

Miss. Priyanka P. Yeotikar
M.E. 2nd yr I.T. SIPNA's COET
Amaravati, India
priyapy27@gmail.com

Prof. P. R. Deshmukh
Computer & I.T. department
SIPNA's COET Amaravati, India
pr_deshmukh@yahoo.com

Abstract: In a multi script environment, majority of the documents may contain text information printed in more than one script/language forms. For automatic processing of such documents through Optical Character Recognition (OCR), it is necessary to identify different script regions of the document. In this context, this paper proposes to develop a model to identify and separate text words of Kannada, Hindi and English scripts from a printed tri-lingual document. The proposed method is trained to learn thoroughly the distinct features of each script and uses the simple voting technique for classification. Experimentation conducted involved 1500 text words for learning and 1200 text words for testing. Extensive experimentation has been carried out on both manually created data set and scanned data set. The average success rate is found to be 99% for manually created data set and 98.5% for data set constructed from scanned document images.

Keywords: Multi-lingual document processing, Script Identification, Feature Extraction, Binary Tree classifier.

I. INTRODUCTION

In India, a single document page may contain words in two or more language scripts. So, multi-script Optical Character Recognition (OCR) is necessary to read these documents for such a country. In most Indian script alphabet system apart from vowel and consonant characters, called basic characters, there are compound characters formed by combining two or more basic characters. The shape of a compound character is usually more complex than the constituent basic characters. Many researchers have developed character recognizers tuned to specific applications, but multilingual capability has not received much attention. The capability of recognizing multilingual documents is both novel and useful.

Script identification is an important problem in the field of document image processing, with its applications to sort document images, as pre processor to select specific OCRs, to search online archives of document images for those containing a particular language, to design a multi-script OCR system and to enable automatic text retrieval based on script type of the underlying document. Automatic script identification has been a challenging research problem in a multilingual environment over the last few years. All existing works on automatic language identification are classified into either local approach or global approach. In recent years, the growing use of physical documents has made to progress towards the creation of electronic documents to facilitate easy communication and storage of documents. However, the usage of physical documents is still prevalent in most of the communications. For instance, the fax machine remains a very important means of communication worldwide. Also, the fact that paper is a very comfortable and secured medium to deal with, ensures that the demand for physical documents continues for many more years to come. So, there is a great demand for software, which automatically extracts, analyses and stores information from physical documents for later retrieval. All these tasks fall under the general heading of document image analysis, which has been a fast growing area of research in recent years.

One important task of document image analysis is automatic reading of text information from the document image. The tool Optical Character Recognition (OCR) performs this, which is broadly defined as the process of reading the optically scanned text by the machine. Almost all existing works on OCR make an important implicit assumption that the script type of the document to be processed is known beforehand. In an automated multilingual environment, such document processing systems relying on OCR would clearly need human intervention to select the appropriate OCR package, which is certainly inefficient, undesirable and impractical. If a document has multilingual segments, then both analysis and recognition problems become more severely challenging, as it requires the identification of the languages before the analysis of the content could be made [10]. So, a pre-processor to the OCR system is necessary to identify the script type of the document, so that specific OCR tool can be selected. The ability to reliably identify the script type using the least amount of textual data is essential when dealing with document pages that contain text words of different scripts. An automatic script identification scheme is useful to (i) sort document images, (ii) to select specific Optical Character Recognition (OCR) systems and (iii) to search online archives of document image for those containing a particular script/language.

II. RELATED WORK

Automatic script identification is a challenging research problem in a multi script environment over the last few years. Major work on Indian script identification is by Pal, Choudhuri and their team [1, 3, 5]. Pal and Choudhuri [1] have proposed an automatic technique of separating the text lines from 12 Indian scripts (English, Devanagari, Bangla, Gujarati, Tamil, Kashmiri, Malayalam, Oriya, Punjabi, Telugu and Urdu) using ten triplets formed by grouping English and Devanagari with any one of the other scripts. This method works only when the triplet type of the document is known. Script identification technique explored by Pal [3] uses a binary tree classifier for 12 Indian scripts using a large set of features. The method suggested in [3] segments the input image up to character level for feature extraction and hence complexity increases. Lijun Zhou et. al. [9] has developed a method for Bangla and English script identification based on the analysis of connected component profiles. Santanu Choudhuri, et al. [4] has proposed a method for identification of Indian languages by combining Gabor filter based technique and direction distance histogram classifier considering Hindi, English, Malayalam, Bengali, Telugu and Urdu. Gopal Datt Joshi, et. al. [6] have presented a script identification technique for 10 Indian scripts using a set of features extracted from log-Gabor filters. Ramachandra Manthalkar et.al. [19] have proposed a method based on rotation-invariant texture features using multichannel Gabor filter for identifying seven Indian languages namely Bengali, Kannada, Malayalam, Oriya, Telugu and Marathi. Hiremath et al. [20] have proposed a novel approach for script identification of South Indian scripts using wavelet based co-occurrence histogram features.

Sufficient work has also been carried out on non-Indian languages [2, 17, 18]. Tan [2] has developed a rotation invariant texture feature extraction method for automatic script identification for six languages: Chinese, Greek, English, Russian, Persian and Malayalam. Lijun Zhou et. Al. [9] has developed a method for Bangla and English script identification based on the analysis of connected component profiles. Peake and Tan [10] have proposed a method for automatic script and language identification from document images using multiple channel (Gabor) filters and gray level co-occurrence matrices for seven languages: Chinese, English, Greek, Korean, Malayalam, Persian and Russian. Wood et al. [14] have proposed projection profile method to determine Roman, Russian, Arabic, Korean and Chinese characters. Hochberg et al. [15] have presented a method for automatically identifying script from a binary document image using cluster-based text symbol templates. Andrew Bhush [17] has presented a texture-based approach for automatic script identification. Spitz has [18] proposed method to discriminate between the Chinese based scripts and the Latin based scripts.

III. ANALYSIS OF PROBLEM

Pal and Choudhuri[1] have proposed an automatic technique of separating the text lines from 12 Indian scripts (English, Devanagari, Bangla, Gujarati, Tamil, Kashmiri, Malayalam, Oriya, Punjabi, Telugu and Urdu) using ten triplets formed by grouping English and Devanagari with any one of the other scripts. This method works only when the triplet type of the document is known. Script identification technique explored by Pal uses a binary tree classifier for 12 Indian scripts using a large set of features. The method suggested in segments the input image up to character level for feature extraction and hence complexity increases. Some considerable amount of work has been carried out on specifically the three languages - Kannada, Hindi and English. Basavaraj Patil et. al. [7] have proposed a neural network based system for script identification of Kannada, Hindi and English languages. Vipin Gupta et. al. [13] have presented a novel approach to automatically identify Kannada, Hindi and English languages using a set of features- cavity analysis, end point analysis, corner point analysis, line based analysis and Kannada base character analysis. Word level script identification in bilingual documents through discriminating features has been developed by Dhandra et. al. [8]. Padma et. al. [11] have presented a method based on visual discriminating features for identification of Kannada, Hindi and English text lines. Though a great amount of work has been carried out on identification of the three languages Kannada, Hindi and English, very few works are reported in literature at word level. Also, the great demand for automatic processing of tri-lingual documents shows that much more work needs to be carried out on word level identification. So, this paper focuses on word wise identification of Kannada, Hindi and English scripts.

IV. PROPOSED WORK

A. Data Collection:

Standard database of documents of Indian languages is currently not available. In this paper, it is assumed that the input data set contains text words of Kannada, Hindi and English scripts and English numerals. For the experimentation of the proposed model, three sets of database were constructed, out of which one database was used for learning and the other two databases were constructed to test the system. The text words of Kannada and English scripts, and English numerals were created using the Microsoft word software. These text words were imported to the Micro Soft Paint program and saved as black and white bitmap (BMP) images. The font type of Times New Roman, Arial, Bookman Old Style and Tahoma were used for English language. The font type of Vijaya, Kasturi and Sirigannada were used for Kannada language. The font size of 14, 20 and 26 were used for both Kannada and English text words. However, the performance is independent of font size.

The text words of Hindi language were constructed by clipping only the text portion of the document downloaded from the Internet. So, the data set constructed using Microsoft word software and by clipping the text portion from the downloaded documents is called manually created data set. Thus the data set of 500 text words from each of the four classes (Kannada, Hindi, English and English Numerals) was constructed to train the proposed system. To test the proposed model, two different data sets were constructed. One dataset of size 300 text words was constructed manually similar to the data set constructed for learning and the other data set was constructed from the scanned document images. The printed documents like newspapers and magazines were scanned through an optical scanner to obtain the document image. The test document image of size 600x600 pixels were considered such that each text line would contain text words in mixture of the three languages. Manually constructed dataset is considered as good quality dataset and the data set constructed from the scanned document images are considered poor quality data set. The test data set was constructed such that 200 text words were incorporated from each of the three scripts - Kannada, Hindi and English, and English numerals.

B. *Preprocessing:*

Any script identification method used for identifying the script type of a document, requires conditioned image input of the document, which implies that the document should be noise free, skew free and so on. In this paper, the preprocessing techniques such as noise removal and skew correction are not necessary for the manually constructed data sets. However, for the datasets that were constructed from the scanned document images, preprocessing steps such as removal of non-text regions, skew-correction, noise removal and binarization is necessary. In the proposed model, text portion of the document image was separated from the non-text region manually. Skew detection and correction was performed using the existing technique proposed by Shivakumar [16]. Binarization can be described as the process of converting a gray-scale image into one, which contains only two distinct tones, that is black and white. In this work, a global thresholding approach is used to binarize the scanned gray scale images where black pixels having the value 0’s correspond to object and white pixels having value 1’s correspond to background.

C. *Overview Of The Proposed Model:*

The proposed model is inspired by a simple observation that every script/language defines a finite set of text patterns, each having distinct visual discriminating features. Hence, the new model is designed by using the distinct features of the scripts under consideration. Scripts are made up of different shaped patterns to produce different character sets. Individual text patterns of one script are collected together to form meaningful text information in the form of a text word, a text line or a paragraph. The collection of the text patterns of the one script exhibits distinct visual appearance and hence it is necessary to thoroughly study the discriminating features of each script that are strong enough to distinguish from other scripts. To arrive at the distinct features of each script under consideration, the complete character set of those scripts should be thoroughly studied. Sometimes, a text word may even contain only two characters thus making the feature extraction process too complex. As a result, a large number of features have to be considered to develop word level script identification model. The properties of the three scripts - Kannada, Hindi and English, and English Numerals are described below.

a. *Some Discriminating Features of Kannada Script:*

Modern Kannada character set has 47 basic characters, out of which the first 13 are vowels and the remaining 34 characters are consonants [10]. Some books report 14 vowels and 36 consonants. By adding vowels to each consonant, modified consonants are obtained. A consonant or a modified consonant is combined with another consonant to form a compound character. As a result, Kannada text words consists of combination of vowels, consonants, modified consonant and/or compound characters. The compound characters have descendants called ‘vathaksharas’ found at their bottom portions. Some examples of Kannada compound characters with descendants are given in Figure 1.

The presence of these descendants is one of the discriminating features of Kannada script, which is not present in the other two scripts – Hindi and English and hence, it could be used as a feature named bottom-component to identify the text word as a Kannada script. It could be observed that most of the Kannada characters have either horizontal lines or hole-like structures present at the top portion of the characters. Also, it could be observed that majority of Kannada characters have upward curves present at their bottom portion. Some characters have double-upward curves found at their bottom portions. In addition, left curve and right curve are also present at the left and right portion of some characters. Thus, the presence of the structures such as – horizontal lines, hole-like structures, bottom-up-curves, descendants, left-curves and right-curves could be used as the supporting features to identify Kannada scripts. Some examples of Kannada characters with the above said features are given in Figure 1. The probability of presence of these features is thoroughly studied from a large collection of documents. The density of the occurrence of these features is thoroughly studied and the features with maximum density are considered in the proposed model.

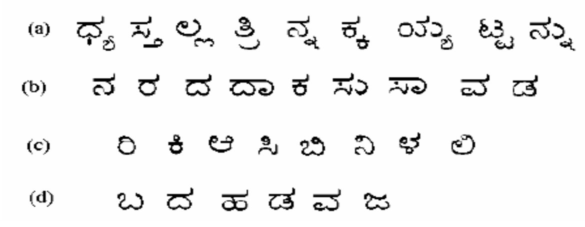


Figure 1. Some characters of Kannada script (a) Characters with descendents, (b) Characters with horizontal lines (c) Characters with holes at the top portion and (d) Characters with double-upward curves

b. *Some Discriminating Features of Hindi Script:*

It could be noted that many characters of Hindi script have a horizontal line at the upper part called sirorekha [1], which is generally called a headline. It could be seen that, when two or more characters are combined to form a word, the character headline segments mostly join one another and generates one long headline at the top portion of each text word. These long horizontal lines are present at the top portion of the characters. The presence of such horizontal lines is used as supporting features for identifying Hindi script. Another strong feature that could be noticed in Hindi script is the presence of vertical lines. Some examples of Hindi text words are given in Figure 2.

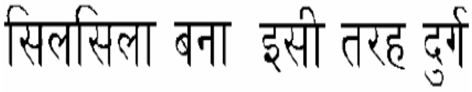


FIGURE 2. Some text words of Hindi script

c. Some Discriminating Features of English Script:

English character set has 26 alphabets in both upper and lower cases. One of the most distinct and inherent characteristics of most of the English characters is the existence of vertical line-like structures. It could be observed that the upward-curve and downward-curve shaped structures are present at the bottom and top portion of majority of English characters respectively. So, it was inspired to use these distinct characteristics as supporting features in the proposed script identification model.

d. Text word Partitioning:

By thoroughly observing the structural outline of the characters of the three scripts, it is observed that the distinct features are present at some specific portion of the characters. So, in this paper, the discriminating features are well projected by partitioning the text line using the four lines that are obtained from the top-profile and the bottom-profile of each text line. The top-profile (bottom profile) of a text line represents a set of black pixels obtained by scanning each column of the text line from top (bottom) until it reaches a first black pixel. Thus, a component of width N gets N such pixels. The row at which the first black pixel lies in the top-profile (bottom-profile) is called top-line (bottom-line). The row number having the maximum number of black pixels in the top profile (bottom-profile) is called the attribute top-max-row (bottom-max-row). Using these four lines – top-line, bottom-line, top-max-row and bottom-max-row as the reference lines, the features are extracted from each text line of the respective script. A sample partitioned Kannada text word is shown in Figure 3. The attribute 'x-height' represents the difference between top-max-row and bottom-max-row and the attribute 'text-height' represents the difference between top-line and bottom-line.

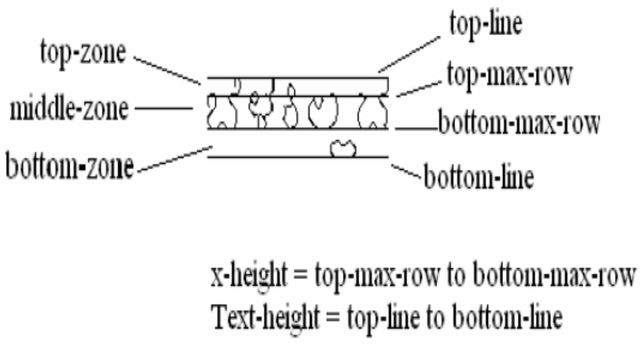

top-zone
middle-zone
bottom-zone
top-line
top-max-row
bottom-max-row
bottom-line
x-height = top-max-row to bottom-max-row
Text-height = top-line to bottom-line

FIGURE 3. Partitioned Kannada Text Word.

Different text words are partitioned in different ways, as different shaped characters are present in text words. A sample Kannada text word that is partitioned into three zones namely top-zone, middle-zone and bottom-zone is shown in Figure 3. A partition with at least three pixels height (fixed through experimentation) is considered as a top-zone or bottom-zone. A text word can be partitioned into three zones only when the four reference lines namely top-line, top-max-row, bottom-max-row and bottom-line are obtained. However, for some text words where top-line and top-max-row occur at the same location, top-zone is not obtained. Similarly, for some txt words if bottom-max-row and bottom-line occur at the same location, then bottom-zone is not obtained. This is because of the absence of ascendants or descendants. Ascendants are the portion of the characters that are protruded above the top-max-row and descendants are the portions of the characters that are protruded below the bottom-max-row. In Kannada script the presence of 'vothaksharas' could be considered as a descendant. So, for a Kannada text word with descendant, the two reference lines – bottom-max-row and bottom-line are present and the space between the bottom-max-row and bottom-line could be called as the bottom-zone. The partitioning of the typical Kannada text word with the descendant is shown in Figure 3. However, if the text word without the descendant is partitioned, then the bottom-zone is not obtained, since the bottom-max-row and the bottom-line occur at the same row. Such a text word in Kannada script without the descendant is shown in Figure 4. Similarly, a text word without ascendants does not possess top-zone.

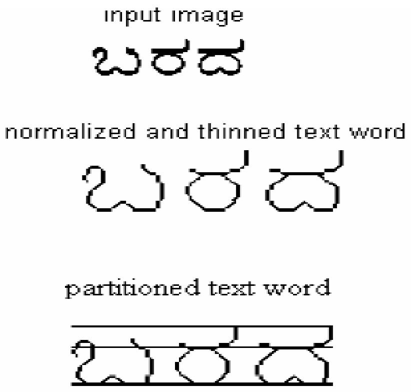


FIGURE 4. Kannada Text Word Without bottom-zone.

English text words are partitioned in a similar way as that of Kannada text word partitioning. This is because, some English characters like 'b, d, f, h, k, l and t' have ascendants and some characters like 'g, j, p, q and y' have descendants. So, if the characters of the text word have ascendants, then the top-zone is obtained and if the characters of the text word have descendants then the bottom-zone is obtained. For some other characters like 'a, c, e, m, n, o, r, s, u, v, w, x, z', there are no ascendants and descendants. For the text words having these characters, top-zone and also bottom-zone are not obtained. So, only middle zone is obtained for such text words. Different partitioned English text words are shown in Figures 5 and 6.

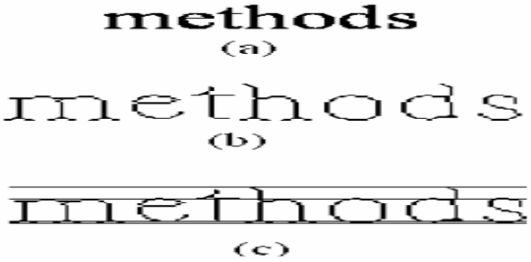


FIGURE 5.Partitioned English Text Word Without Descendant (a) Input Word (b) Preprocessed Word(c) Partitioned Word without bottom-zone

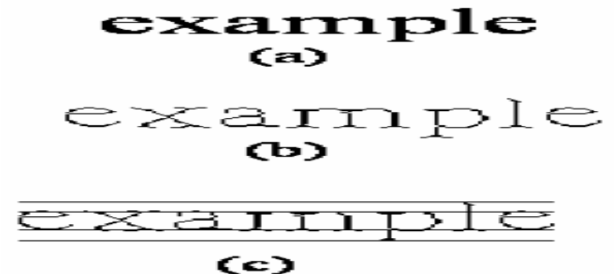


Figure 6. Partitioned English Text Word With Descendant (a) Input Word (b) Preprocessed Word (c) Partitioned Word with top-zone and bottom-zone

It is observed that all English numerals are equal in height. So, the partitioning of a text word containing only English numerals results in middle-zone only. A sample image of English numeral and its partitioned image are shown in Figure 7.

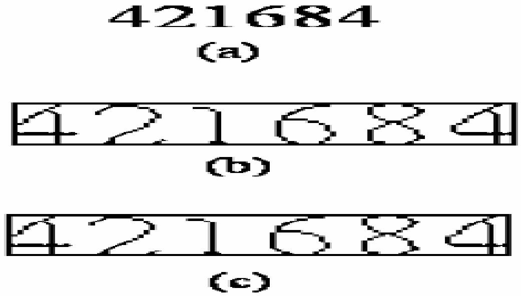
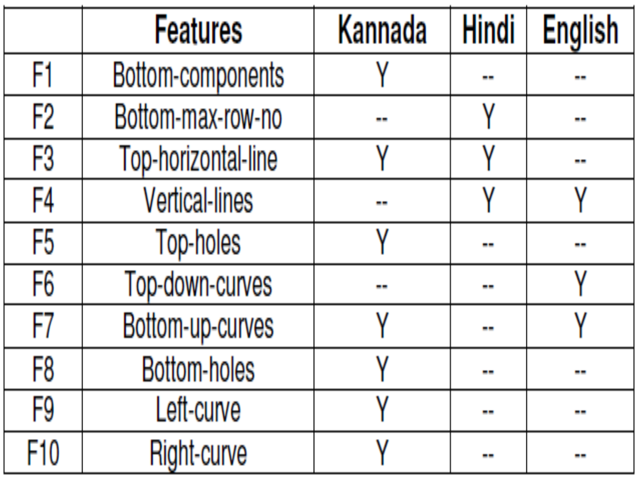


Figure 7. Partitioned English Numeral (a) Input Word (b) Preprocessed Word with Bounding Box (c) Partitioned Word without top-zone and bottom-zone.

e. *Feature Extraction:*

The distinct features useful for identifying the three scripts – Kannada, Hindi and English are shown in Table 1. The entry 'Y' in the Table 1 means that the feature in the corresponding row is used for identifying the script in the corresponding column. Thus, seven features for Kannada, three features for Hindi and three features for English are used. It is observed in Table 1 that the features used for identifying English numerals are not listed. The method of identifying English numerals is explained in the later Section.

TABLE 1. Features of Kannada, Hindi and English languages.



	Features	Kannada	Hindi	English
F1	Bottom-components	Y	--	--
F2	Bottom-max-row-no	--	Y	--
F3	Top-horizontal-line	Y	Y	--
F4	Vertical-lines	--	Y	Y
F5	Top-holes	Y	--	--
F6	Top-down-curves	--	--	Y
F7	Bottom-up-curves	Y	--	Y
F8	Bottom-holes	Y	--	--
F9	Left-curve	Y	--	--
F10	Right-curve	Y	--	--

The method of extracting the distinct features, which are used in the proposed model, is explained below:

a) *Feature 1: Bottom-component:*

The presence of vathaksharas or descendants found at the bottom portion of Kannada script could be used as a feature called bottom-component. The feature named 'bottom-component' is extracted from the bottom-portion of the input text line. Bottom-portion is computed as follows:

Bottom-portion = f(x,y) where x=bottom-max-row to m and y=1to n ; where f(x,y) represent the matrix of the preprocessed input image of size (m x n).

Through experimentation, it is estimated that the number of pixels of a descendant is greater than 8 pixels and hence the threshold value for a connected component is fixed as 8 pixels. Any connected component whose number of pixels is greater than 8 pixels is considered as the feature bottom-component. Such bottom-components extracted from Kannada script are shown in Figure 11.

b) *Feature 2: Bottom-max-row-no:*

It is observed through experimentation that for Kannada and English script, the two attributes topmax- row and bottom-max-row occur at a distance of x-height as shown in the partitioned Kannada text word shown in Figure 3 and partitioned English text words shown in Figure 5 and 6. But, for a partitioned Hindi text word, the x-height is 0 or 1 pixel, as the top-max-row and bottommax- row occur at the same location. This is because when the bottom profile of Hindi script is computed the pixels of the headline happen to be the pixels of bottom profile. So, the top-maxrow and bottom-max-row occur at the same location in a Hindi text word. This is the distinct property of Hindi script that is not so in the other two scripts English and Kannada. Hence, if the bottom-max-row is equal to the top-max-row, then the value of the attribute bottom-max-row could be used as a strong feature named 'bottom-max-row-no' to separate Hindi script from the other two scripts. A typical Hindi text word with the feature 'bottom-max-row-no' is shown in Figure 8.

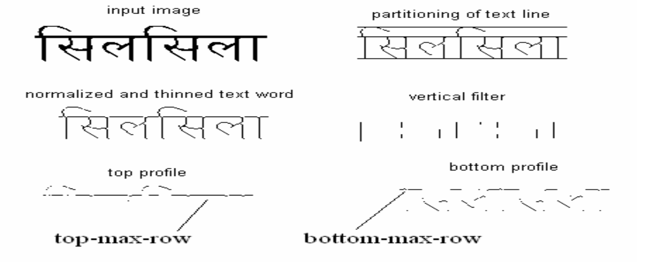


Figure 8. Hindi text word with bottom-max-row and vertical lines

c) *Feature 3: Top-horizontal-line:*

It could be noted that the horizontal line like structures are present at the top-max-row of Kannada and Hindi scripts. The connected components present at the top-max-row of the text word are analyzed. If the number of pixels of these connected components is greater than the 75% of the x-height, then such components are used as the feature top-horizontal-line. The probability of presence of this feature is calculated from the complete Kannada character set. Also, the distribution of this feature is analyzed using 500 text words from all the three languages Kannada, Hindi and English. From the experimental analysis, it is observed that the presence of top-horizontal-line is more in Kannada and Hindi script and it is almost absent in the case of English script. So, using the feature named top-horizontal-line, Kannada and Hindi scripts could be separated from English script. If the length of the horizontal line (length of the horizontal line is measured with the number of pixels of that component) is greater than two times the x-height, then Hindi text word can be separated from Kannada word. So, using the length of the feature top-horizontal-line, Hindi can be well separated from Kannada script. The feature top-horizontal line is shown in the output images of Hindi and Kannada text words in Figures 8 and 11 respectively.

d) Feature 4: Vertical lines:

It is noticed that the Hindi and English scripts have vertical line segments. To extract these vertical lines, the middle-zone of the text line is extracted as below:

Middle-zone = g(x,y) where x = top-max-row to bottom-max-row and y = 1 to n

Where g(x,y) is the input matrix size (m,n). By convolving a vertical line filter over the image of the middle-zone, vertical lines are extracted. Typical vertical lines extracted from English script are shown in Figure 9. The presence of these vertical lines more in Hindi and English script, whereas it is absent in Kannada script. Hence, the feature vertical lines are used to identify Hindi and English script.

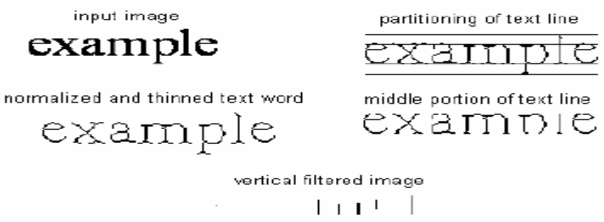

input image
example
partitioning of text line
example
normalized and thinned text word
example
middle portion of text line
example
vertical filtered image

FIGURE 9. English text word with vertical lines

e) Feature 5: Top-holes:

Hole is a connected component having a set of white pixels enclosed by a set of black pixels (black pixels having the value 0's correspond to object and white pixels having value 1's correspond to background). By thoroughly observing the words of Kannada scripts, it is noticed that hole-like structures are found at the top portion. To compute the hole-like structures, the attribute top-pipe is obtained from the matrix of the pre-processed image as follows:

Top-pipe (x1, y1) = f (x, y) where x=top-max-row – t to top-max-row+t and y=1 to n where f(x,y) and n represents the input image and number of columns of the input image. The variable 't' is used as a threshold value and t=round(x-height/3), where the term 'x-height' represents the difference between top-max-row and bottom-max-row. Presence of holes at the top-pipe is used as the feature top-holes and it is used to identify the text word as Kannada script as this feature is not present in the other two anticipated languages. Hole-like structures extracted from the sample Kannada script text word is shown in Figure 11.

f) Features 6 & 7: Top-down-curves and Bottom-up-curves:

By thoroughly observing the structural shape of the two scripts – Kannada and English, it is observed that the upward and downward shaped components are present at the region of topmax- row and bottom-max-row. This inspired us to extract the two attributes top-pipe and bottompipe as follows:

Top-pipe = g(x,y) where x=top-max-row – t to top-max-row+t and y=1 to n and Bottom-pipe = g(x,y) where x=bottom-max-row – t to bottom-max-row + t and y=1 to n where g(x,y) and n represents the input image and number of columns of the input image. The variable 't' is used as a threshold value and t=round(x-height/3), where the term 'x-height' represents the difference between top-max-row and bottom-max-row..

Detecting the curve shaped portion from a character is the key for extracting the features named top-down-curves and bottom-up-curves. The presence of a curve is obtained by verifying the variation between two pixels of a connected component that appear on the same scan line for the complete scan of the component. The increasing variations of the two pixels for the entire scan of the component results in top-down-curves and decreasing variations of the two pixels for the entire scan of the component results in bottom-down-curves. Components having the shape upward curve and downward curve are shown in Figure 10.

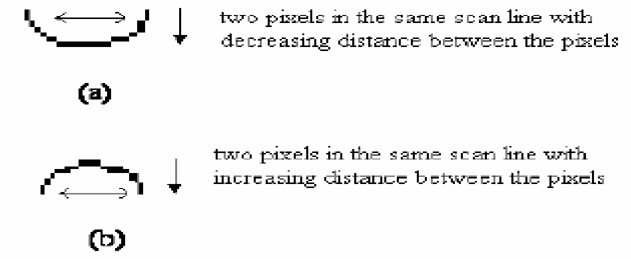

two pixels in the same scan line with decreasing distance between the pixels
(a)
two pixels in the same scan line with increasing distance between the pixels
(b)

FIGURE 10. (a) upward curves (b) downward curves

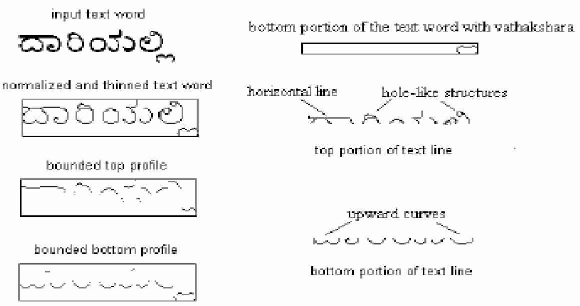


FIGURE11. Output image of Kannada text word

g) Feature 8: Bottom-holes:

By thoroughly observing the text words of Kannada scripts, it is observed that some characters have hole-like structures at their bottom portion. To compute the hole-like structures, the attribute bottom-pipe is obtained from the matrix of the pre-processed image as follows:

Bottom-pipe (x1, y1) = f (x, y) where x=bottom-max-row – t to bottom-max-row + t and y=1 to n where f(x,y) and n represents the input image and number of columns of the input image. The variable ‘t’ is used as a threshold value and t=round(x-height/3), where the term ‘x-height’ represents the difference between top-max-row and bottom-max-row..

Presence of holes at the bottom-pipe is used as the feature bottom-holes and it is used to identify the text word as Kannada script as this feature is not present in the other two anticipated scripts.

h) Feature 9 and 10: Left-curve and Right-curve:

The structural shape of the two scripts – Kannada and English is observed thoroughly and noticed that the left-curve and right-curve shaped components are present at the middle-zone of a partitioned text word. This inspired us to extract the middle-zone as follows:

Middle-zone = g (x, y)

Where x = (top-max-row + t) to (bottom-max-row – t) and y = 1 to n, g (x, y) represents the input image and ‘n’ is the number of columns of the input image. The variable ‘t’ is used as a threshold value determined through experimentation and t = 2.

Detecting the curve shaped portion from a character is the key for extracting the features named top down-curves and bottom-up-curves. The presence of a left curve is obtained by verifying the distance between two pixels of a connected component that appear on the same vertical scan line of a component. The increasing variations of the two pixels for the entire vertical scan of the component results in the left-curve and decreasing variations of the two pixels for the entire vertical scan of the component results in right-curve. Components having the shape left curve and right-curve are shown in Figure 12.

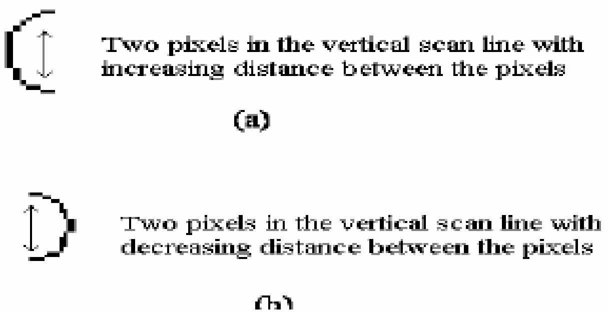

Two pixels in the vertical scan line with increasing distance between the pixels
(a)
Two pixels in the vertical scan line with decreasing distance between the pixels
(b)

Figure 12. (a) left-curve (b) right-curve

f. Simple voting technique for classification:

A simple voting technique is an approach used for classifying a test sample based on the maximum number of votes obtained for a class type. In this chapter, a simple voting technique is used to classify the given test sample into any of the four classes- Kannada, English, Hindi and English Numerals. The overview of the working principle of the simple voting technique is explained below:

Here, a simple voting technique is used for classifying the test word. For each class type, a particular accumulator, which is initialized to zero, is used to count the number of features present in a test word. For the present problem, there are four script classes – Kannada, English, Hindi and English Numerals. So, four accumulators namely KA, HA, EA and NA are used for the four script classes - Kannada, Hindi, English and English Numerals respectively. In this method, the test word is segmented into several blocks through vertical projection profiles. A block may contain one or more than one character with one or more connected components. So, in this method, the term character or block may be used interchangeably. After segmenting the text word, the number of blocks present in a test word is calculated. Each segmented block is tested for the presence of the features given in Table 1. For example, if the feature F3 is present in a block, then the accumulator KA is incremented, as only Kannada characters possess this feature. Then the presence of the feature F4 is tested and if this feature is present only English accumulator is incremented. This process is repeated for all the blocks of the given test sample. At the end of this procedure, all the four accumulators are stored with some value. The highest value is found in one of the accumulator and the label of highest value shows the class label of the test sample.

The features are extracted from the partitioned image in the order given in Table 1. Later, a simple voting technique is used to classify the test word into the respective script type. The voting technique used in the word level script identification model works in three stages as explained below:

Algorithm Stage 1:

```
Input: Document Image containing text words of Kannada, Hindi, English and English Numerals.
Output: Script type of each text word.
    a)  Preprocess the input document image.
    b)  Segment the document image into several text lines.
    c)  Repeat for each text line
    d)  { Segment the text line into words
    e)  Repeat for each text word
    f)  { Partition the text word
    g)  If (bottom-component)
        { Then Classify the text word as “Kannada” script and Return}
    h)  If (Bottom-max-row)
        { Then Classify the text word as “Hindi” script and Return.}
    i)  If (only one zone) then call stage 3 Else call stage 2.
        } }
```

Algorithm Stage 2:

```
    a)  Segment the text word into characters
    b)  Initialize the four accumulators -KA, HA, EA, NA to zero.
    c)  Repeat for each character
    d)  { Repeat for each feature F3 through F10
        { If (feature present)
        Then increment the corresponding accumulator } }
    e)  Find the accumulator with maximum value.
    f)  Classify the script type of the text word as the corresponding accumulator.
    g)  Return
```

Algorithm Stage 3:

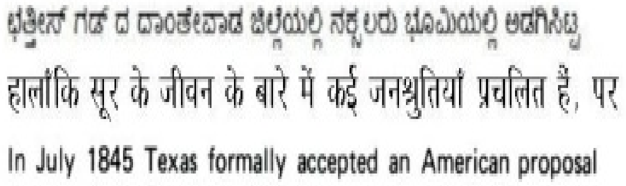
```
    a)  Segment the text word into characters/blocks
    b)  Initialize the four accumulators -KA, HA, EA, NA to zero.
    c)  Repeat for each character
    d)  { Repeat for each feature F3 through F10
```

```
      { If (feature present)
      Then increment the corresponding accumulator } }
e)    Find the accumulator content with maximum value.
f)    If (accumulator value >= n/2)
      { Then Classify the script type of the text word as
      the corresponding accumulator.
      Else If (width of 90% of the characters is same)
      Then Classify the script type as English Numerals
      Else Reject
g)    Return
```

V. RESULTS

The proposed algorithm has been tested on a test data set of 300 document images containing about 500 text words from each script. The test data set is constructed such that the English text words contain characters that possess ascendants (for example b, d, f, h, k, l, t) and descendants (for example g, j, p, q, y). The English text word without any ascendants and descendants (for example words like ‘cow’, ‘man’, ‘scanner’) are also considered in the test data set. The performance of classification is encouraging when tested with all kinds of words having the characters with and without ascendants and descendants. Similarly, the test data set of Kannada and Hindi scripts were constructed such that all characters of the two scripts are included in the test words. The algorithm is tested for text words containing two to six characters. The success rate is sustained even for the text words with only two characters. This is because all the features present in one or the other character are used in the proposed model. Satisfactory success rate was achieved even in classifying the English Numerals.

The proposed algorithm has been implemented using Matlab R2008b. The average time taken to identify the script type of the text word is 0.1846 seconds on a Core 2 Duo with 1GB RAM based machine running at 2.60 GHz. A sample manually constructed test document containing text words of all the four classes- Kannada, Hindi, English and English numerals are given in Figure 13.



ಛತ್ತೀಸ್ ಗಢ ದ ದಾಂತೇವಾಡ ಜಿಲ್ಲೆಯಲ್ಲಿ ನಕ್ಸಲರು ಭೂಮಿಯಲ್ಲಿ ಅಡಗಿಸಿಟ್ಟ

हालांकि सूर के जीवन के बारे में कई जनश्रुतियां प्रचलित हैं, पर

In July 1845 Texas formally accepted an American proposal

Figure 13. Manually created test image containing Kannada, Hindi, English and Numeral words

The algorithm is tested for various font types and the results are given in Table 2. The proposed method is independent of font type and size. Since the features are considered to be at specific region of the partitioned text word, the variation in the font size does not affect the performance of the algorithm. From the experimentations on the test data set, the overall accuracy of the system has turned out to be 98.8%.The performance of the proposed algorithm falls down for English text words printed in italics. This is one limitation. However, for the Kannada text words printed in italics, the performance is sustained. The performance of the proposed model was evaluated from the scanned document images also. The overall accuracy of the system reduces to 98.5% due to noise and skew-error in the scanned document images. However, if the scanned document images undergo suitable Preprocessing techniques, the performance can be improved.

Table 2. Percentage of Recognition on manually created data set for different font styles.

Script type	**Font Style**	**Number of samples**	**Correct recognition**	**Recognition rate**
Kannada	Sirigannada	150	147	98%
	Kasturi	160	157	98.13%
	Vijaya	170	167	98.23%
Hindi	Vijaya	200	200	100%
English	Times New Roman	100	97	97%
	Arial	100	98	98%
	Verdana	100	98	98%
	English Text: Upper Case only	100	100	100%
English Numerals	Times New Roman	100	97	97%
	Arial	100	96	96%
	Verdana	100	97	97%

VI. CONCLUSION

In this paper, a new method to identify and separate text words of the Kannada, Hindi and English scripts and also English numerals is presented. Experimental results show performance of the proposed model. The performance of the proposed algorithm is encouraging when the proposed algorithm is tested using manually created data set. However, the performance slightly comes down when the algorithm is tested on scanned document images due to noise and skew error. Our future work is to identify the numeral information printed in the Kannada and Hindi scripts and also to reach higher rate of success (100%). Further, it is planned to identify the scripts from a degraded document images.

VII. ACKNOWLEDGMENT

I express my sincere gratitude to Dr. A. D. Gawande Head of the Department, Information Technology and guide Prof. P. R. Deshmukh for providing their valuable guidance and necessary facilities needed for the successful completion of this seminar throughout. I am also obliged to our principal, Dr. S. A. Ladhake who has been a constant source of inspiration throughout.

VIII. REFERENCES

[1]. U.Pal, B.B.Choudhuri, “Script Line Separation From Indian Multi-Script Documents”, 5th Int. Conference on

Document Analysis and Recognition(IEEE Comput. Soc. Press), 406-409, (1999).

[2]. T.N.Tan, “Rotation Invariant Texture Features and their use in Automatic Script Identification”, IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 20, no. 7, pp. 751-756, (1998).

[3]. U. Pal, S. Sinha and B. B. Chaudhuri, “Multi-Script Line identification from Indian Documents”, In Proceedings of the Seventh International Conference on Document Analysis and Recognition (ICDAR 2003) 0-7695-1960-1/03 © 2003 IEEE, vol.2, pp.880-884, (2003).

[4]. Santanu Choudhury, Gaurav Harit, Shekar Madnani, R.B. Shet, “Identification of Scripts of Indian Languages by Combining Trainable Classifiers”, ICVGIP, Dec.20-22, Bangalore, India, (2000).

[5]. S. Chaudhury, R. Sheth, “Trainable script identification strategies for Indian languages”, In Proc. 5th Int. Conf. on Document Analysis and Recognition (IEEE Comput. Soc. Press), pp. 657–660, 1999.

[6]. Gopal Datt Joshi, Saurabh Garg and Jayanthi Sivaswamy, “Script Identification from Indian Documents”, LNCS 3872, pp. 255-267, DAS (2006).

[7]. S.Basavaraj Patil and N V Subbareddy, “Neural network based system for script identification in Indian documents”, Sadhana Vol. 27, Part 1, pp. 83–97. © Printed in India, (2002).

[8]. B.V. Dhandra, Mallikarjun Hangarge, Ravindra Hegadi and V.S. Malemath, “Word Level Script Identification in Bilingual Documents through Discriminating Features”, IEEE – ICSCN 2007, MIT Campus, Anna University, Chennai, India. pp.630-635. (2007).

[9]. Lijun Zhou, Yue Lu and Chew Lim Tan, “Bangla/English Script Identification Based on Analysis of Connected Component Profiles”, in proc. 7th DAS, pp. 243-254, (2006).

[10]. G. S. Peake and T. N. Tan, “Script and Language Identification from Document Images”, Proc. Workshop Document Image Analysis, vol. 1, pp. 10-17, 1997.

[11]. M. C. Padma and P.Nagabhushan, “Identification and separation of text words of Karnataka, Hindi and English languages through discriminating features”, in proc. of Second National Conference on Document Analysis and Recognition, Karnataka, India, pp. 252-260, (2003).

[12]. Rafael C. Gonzalez, Richard E. Woods and Steven L. Eddins, “Digital Image Processing using MATLAB”, Pearson Education, (2004).

[13]. Vipin Gupta, G.N. Rathna, K.R. Ramakrishnan, “A Novel Approach to Automatic Identification of Kannada, English and Hindi Words from a Trilingual Document”, Int. conf. on Signal and Image Processing, Hubli, pp. 561-566, (2006).

[14]. S. L. Wood, X. Yao, K. Krishnamurthy and L. Dang, “Language identification for printed text independent of segmentation”, Proc. Int. Conf. on Image Processing, pp. 428–431, 0-8186- 7310-9/95, 1995 IEEE.

[15]. J. Hochberg, L. Kerns, P. Kelly and T. Thomas, “Automatic script identification from images using cluster based templates”, IEEE Trans. Pattern Anal. Machine Intell. Vol. 19, No. 2, pp. 176–181, 1997.

[16]. Shivakumar, Nagabhushan, Hemanthkumar, Manjunath, 2006, “Skew Estimation by Improved Boundary Growing for Text Documents in South Indian Languages”, VIVEK International Journal of Artificial Intelligence, Vol. 16, No. 2, pp 15-21.

[17]. Andrew Busch, Wageeh W. Boles and Sridha Sridharan, “Texture for Script Identification”, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 27, No. 11, pp. 1720- 1732, Nov. 2005.

[18]. A. L. Spitz, “Determination of script and language content of document images”, IEEE Trans. On Pattern Analysis and Machine Intelligence, Vol. 19, No.3, pp. 235–245, 1997.

[19]. Ramachandra Manthalkar and P.K. Biswas, “An Automatic Script Identification Scheme for Indian Languages”, IEEE Tran. on Pattern Analysis And Machine Intelligence, vol.19, no.2, pp.160-164, Feb.1997.

[20]. Hiremath P S and S Shivashankar, “Wavelet Based Co-occurrence Histogram Features for Texture Classification with an Application to Script Identification in a Document Image”, Pattern Recognition Letters 29, 2008, pp 1182-1189.