Journal of Information Systems and Communication
ISSN: 0976-8742, E-ISSN: 0976-8750, Volume 3, Issue 1, 2012, pp- 92-95.
Available online at http://www.bioinfo.in/contents.php?id=45

KANNADA LANGUAGE OCR SYSTEM USING SVM CLASSIFIER

ARVIND C.S.[1]*, NITHYA E.[2] AND NABANITA BHATTACHARJEE[3]

[1]Telibrahma Convergent Communication Pvt Ltd, Bangalore
[2]Computer Science Department, Visvesvaraya Technology University, Dr. Ambedkar Institute of Technology, Bangalore
[3]Computer Science Department, Meghnad Saha Institute of Technology, Kolkata
*Corresponding Author: Email- csarvind2000@gmail.com

Received: January 12, 2012; Accepted: February 15, 2012

Abstract- Optical Character Recognition (OCR) is the process of converting the textual image into the machine editable format. This paper proposes an OCR system for printed Kannada Document. The input to the system would be the scanned image of a page of text that containing complex Kannada characters and the output is a machine editable file. The system first pre-processes the input document containing the Kannada characters and converts it into binary form. Then the system extracts the lines from the document image and segments the lines into character and sub-character level pieces. Here projection profile technique and connected component method is used for character segmentation. Zernike 10th Order moment is used to extract kannada character features. SVM classifier is using for classification. Output of the classifier has to be transformed into a format (English) which can be loaded into a Kannada editing package. The method of composition of aksharas in all Kannada typesetting packages is similar. For basic Kannada characters like (Consonants & Vowels) can be obtained easily by loading the classification output to editing package. But for complex Kannada characters, it is slight difficult. So we propose a unique approach for complex character output generation and output a file which is compatible with the "Baraha" typesetting software.
Keywords- SVM, Connected Component method, Hough line, Zernike moments.

Citation: Arvind C.S., Nithya E. and Nabanita Bhattacharjee (2012) Kannada Language OCR System using SVM classifier. Journal of Information Systems and Communication. ISSN: 0976-8742, E-ISSN: 0976-8750, Volume 3, Issue 1,pp- 92-95.

Introduction

Optical Character Recognition is one of the oldest sub fields of pattern recognition with a rich contribution for the recognition of printed documents. OCR is a field of research in pattern recognition, artificial intelligence and computer vision. Though academic research in the field continues, the focus on OCR has shifted to implementation of proven techniques.

When one scans a paper page into a computer, it produces just an image file, a photo of the page. The computer cannot understand the letters on the page, so you cannot search for words or edit it or change the font, as in a word processor. You would use OCR software to convert it into a text or word processor file so that you could do those things. The result is much more flexible and compact than the original page photo.

The need for OCR arises in the context of digitizing the documents from the library, which helps in sharing the data through the Internet.

Currently there are many OCR systems available for handling printed English documents with reasonable levels of accuracy. Such systems are also available for many European languages as well as some of the Asian languages such as Japanese, Chinese, etc. However, there are not many reported efforts at developing OCR systems for Indian languages especially for a South Indian language like Kannada. Kannada is one of the South Indian languages, which has 16 vowels and 34 consonants. It includes the possible consonant-vowel combinations are 16*34 =544. The number of possible consonant-consonant-vowel (Complex characters) combinations is 16*34*34 = 18496. Some of the Kannada characters are shown in Figure 1.

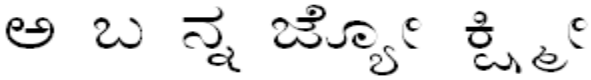

Fig. 1- Printed Kannada Characters

Sanjeev Kunte et.al has worked on Kannada character recognition with Hu's invariant moments and Zernike moments and Neural classifiers is used for the classification of simple characters based on moments features [1]. T.V. Ashwin et.al has used SVM for font and size independent for printed text, to output the results ascii values are taken and given to Kantex typesetting software [2]. B.M. Sagar et.al has used database approach for Kannada OCR system for simple Kannada characters [3]. B.Vijay Kumar et.al has used DCT,DWT, and Karhunen-Louve Transform for feature extraction and classification pattern using nearest neighbour, (ANN) [4].

In this paper a unique approach for complex character output generation is proposed where an English string is generated and translated to Kannada characters.

The string representing an akshara is composed from the English character codes corresponding to the different components of the akshara as follows: the codes for the base consonant appear first followed by the codes for the consonant conjuncts; the codes for the vowel modifier appear at the last and signify the end of an akshara. In our system we currently output a file which is compatible with the "Baraha" typesetting software, It recognized more than hundred Kannada character and more than hundred complex Kannada words. In the document, Here the system gives more than 95% accuracy.

Preprocessing

In Kannada OCR system, the sequence of Pre- processing is as follows. The page of the text is scanned through a flat bed scanner and binaries using local thresholding Ni-Black binarization technique. This image is first processed to remove any skew so that the text is aligned horizontally in the image. For skew correction we employed a hough transform technique (Gorman & Kasturi 1995). Some of the Indian scripts, such as Devanagari, have a dark top line linking all the characters in a word. The strong linear feature can be exploited for skew estimation, in Kannada, such a line linking all characters of the word is not present. However the Hough transform technique for extracting lines works well.

Segmentation

Line Segmentation

To separate the text lines, from the document image, the horizontal projection profile [2] of the document image is found. The horizontal projection profile is the histogram of the number of ON pixels along every row of the image. White space between text lines is used to segment the text lines. Figure 2 shows a sample Kannada document along with its horizontal projection. The projection profile will have valleys of zero height between the text lines. Line segmentation is done at these points.

Character segmentation

The letters in Kannada are composed by attaching to the glyph of a consonant, the glyphs of the vowel modifiers and the glyphs of the consonant conjuncts. If we considered all the combination, then building the classifier of these numbers of character is very difficult. So our strategy is that we will segment the word into its constituents, i.e. the base consonant, the vowel modifier and the consonant conjunct. It's very difficult to achieve this. If we have good look on the Kannada word, we will see that for extracting glyph of a consonant the glyphs of the vowel modifiers and the glyphs of the consonant conjuncts we can divide the character into two zones. Top zone: Top zone mainly consist of main portion of the character. It includes base consonant or vowels or some vowel modifiers. Bottom zone: Bottom zone consists of glyphs for the consonant conjuncts.

Fig. 2- Line segmentation

Here by using connected component method, we are first counting the number of consonants, vowels, vathu or vowel modifiers present in the text line. Connected component method is an algorithmic application of graph theory, where subsets of connected components are uniquely labeled based on a given heuristic. Connected component labeling is used in computer vision to detect connected regions in binary digital images, although color images and data with higher-dimensionality can also be processed. In connected component method connectivity checks are carried out by checking the labels of pixels that are North-East, North, North-West and West of the current pixel. Then we extracting that characters separately and send it for character recognition.

Feature Extraction

To extract the feature of the Kannada printed characters Zernike 10th order moment is been used. Zernike moments are due to Zernike polynomials introduced by Zernike (1934). Zernike polynomials are a set of complex polynomials which form a complete orthogonal set over interior of the unit circle.

The Zernike moments are projections of the input image onto the space spanned by the orthogonal V functions:

$$V_{nm}(x,y)=V_{nm}(r,\theta)=R_{nm}(r).\exp(jm\theta) \quad (1)$$

Where, n>=0,|m|<=n,n-|m| is even, r is the length of vector from origin to (x,y) pixel and is angle between vector r and the x-axis in counter-clock direction.

$$R_{nm}(r)=\sum_{s=0}^{(n-|m|)/2}(-1)^s g\frac{(n-s)!}{s!(\frac{n+|m|}{2}-s)!(\frac{n-|m|}{2})}g^{r^{(n-2s)}} \quad (2)$$

The Zernike moment of order of n with a repetition m of a continuous image function f(x,y) is given by:

$$Z_{nm}=\frac{n+1}{\pi}\sum_{x}\sum_{y}f(x,y)[V_{nm}(x,y)]^{*} \quad (3)$$

Where, $x^2+y^2<=1$ and the symbol * denotes the complex conjugate operator

In order to compute the zernike moments of a given image, the image is taken as the origin and image co-ordinates are mapped t o the unit circle $x^2+y^2<=1$. Only those pixels falling inside the unit circle are considered for the moment computation.

If the image is rotated by an angle α, the transformed Zernike moment function z'_{nm} are given by:

$$z'_{nm} = z_{nm} . e^{-jm\alpha} \quad (4)$$

The means that the magnitude of the moments stays the same after the rotation. Hence, the magnitude of the Zernike |moments of the image, | z_{nm} |, could be taken as rotation invariant feature. The advantage of zernike 10th order moment is that it produce 36 descriptor and the magnitude of zernike moments are invariant to rotation. Zernike moments are robust to noise and minor variations in shape.

Recognition

The feature vector extracted from the segment bitmap has to be assigned a label using a pattern classifier. There are many methods for designing pattern classifier such as Bayas classifier, neural network and correlation method. For classification this system uses Support Vector Machine (SVM).

SVM Classifier: The SVM classifier is a two-class classifier based on the use of discriminant functions. A discriminant function represents a surface which separates the patterns so that the patterns from the two classes lie on the opposite side of the surface. The SVM is essentially a separating surface which is optimal according to a criterion as explained below:

Consider a two-class problem where the class labels are denoted by +1 and -1. Given a set of l labelled (training) pattern. X={(xi,yi), 1<=i<=1}, xi $\in$ R^d ,yi $\in$ {-1,+1}, the hyper-plane represented by (w,b) where w $\in$ R^d represents the normal to the hyper-plane and b $\in$ R represents the normal to the hyper-plane and b $\in$ R the offset, forms a separating hyper-plane or linear discriminant function if the following separability are satisfied

$$\left(x_i^t x_j + 1\right)^p \quad (5)$$

Here, $w^t x_i$ denotes the inner product between the two vectors, and g(x) = wᵗx+b is the linear discriminant function.

SVM used different kernels like Polynomial kernel, Gaussian Kernel and Perceptron kernel

Polynomial kernel

$$w^t x_i + b > 0, for\ i: y_i = +1,$$
$$w^t x_i + b < 0, for\ i: y_i = -1. \quad (6)$$

Power 'p' is specified a priori by the user.

Gaussian kernel

$$exp\left(-\frac{1}{2\sigma^2}\|x_i - x_j\|^2\right) \quad (7)$$

The width σ^2, common to all the kernels is specified a priori

Perceptron kernel

$$\tanh\left(\beta_0 x_i^t x_j + \beta_1\right) \quad (8)$$

Mercer's condition satisfied only for values of β_0 and β_1.

The SVM method for learning two classifiers is as follows. We choose a kernel function and some value for the Ci

$$Minimize: \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{1}\xi_i \quad (9)$$

Then, we solve its dual which is same as

$$Maximixe: \sum_i \alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j z_i^t z_j,$$
$$Subject\ to: \alpha_i \geq 0, i = 1, 2, \ldots\ldots, 1,$$
$$\sum_{i=1}^{1}\alpha_i y_i = 0 \quad (10)$$

Except that the variables α_i also have an upper bound, namely, C. Once we solve this problem, all we need to store are the non-zero α_i and the corresponding xi (which are the support vectors). Using these, given any new feature vector x, we can calculate the output of SVM, namely, f(x) thorough

$$f(x) = \Phi(x)^t w^* + b^* = \sum_{i\in S}\alpha_i^* y_i \Phi(x_i)^t \Phi(x) + b^* \quad (11)$$

The classification of 'x' would be +1 if the output of SVM is positive; otherwise it is -1.

SVM Classifier for OCR

We have used SVM classifiers for labelling each segment of a word. As explaining earlier, we have trained a number of two-classifiers (SVM) each one for distinguishing one class from all others. Thus each of our class labels has on associated SVM. A test example is assigned the label of the class whose SVM gives the largest positive output. If no SVM gives a positive output then the example is rejected. The output of the SVM gives a measure of the distance of the example from the separating hyper-plane in the Φ space. Hence higher the value of the (positive) output for a given pattern higher is the confidence in classifying the pattern. We use Gaussian Kernel function for all SVM's and used a single value for the penalty constant C. The SVMs are trained using the SVM package.

The sequence of segments corresponding to a give akshara is not necessarily unique. This is because; during the segmentation some based consonants may be split into different number of segments for different fonts. Font's information is not known to the system. All possible modes of splitting of the aksharas are known. The sequence of labels output from the classifier is checked before final recognition of all the aksharas. If an inconsistency is observed an attempt is made to remove this inconsistency could

not be removed by this, then that piece is labelled as unrecognizable.

The output after the classification has to be transformed into a format (English) which can be loaded into a Kannada editing package. The method of composition of *aksharas* in all Kannada typesetting packages is similar. The basic Kannada characters (Consonants & Vowels) can be easily obtained by loading the classification output to editing package. But for complex Kannada characters, it is slight difficult. So here unique approach is used for output character generation. The output after the classification is usually in English string format. The English strings are such that the final text looks like transliteration of the Kannada word. The string representing an *akshara* is composed from the English character codes corresponding to the different components of the *akshara* as follows: the codes for the base consonant appear first followed by the codes for the consonant conjuncts; the codes for the vowel modifier appear at the last and signify the end of an *akshara*. In our system we currently output a file which is compatible with the "Baraha" typesetting software.

Results and Discussion

Experiment is carried out using MATLAB 7.9. We measured the performance of our system by scanning document that contains different complex Kannada character documents. The complex Kannada character means, which is the combination of vowel or consonants or vathu or vowel modifier. The system first segments the document into character level pieces and it is compared with the sample characters. We have trained 57 kannada characters and testing 200 complex Kannada words and documents of printed kannada characters. We have used different SVM kernel for classification.

Our system recognition accuracy is about 95%.

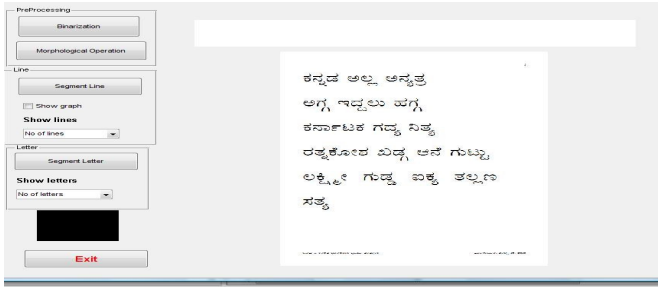

Fig. 3- Input printed Kannada Document

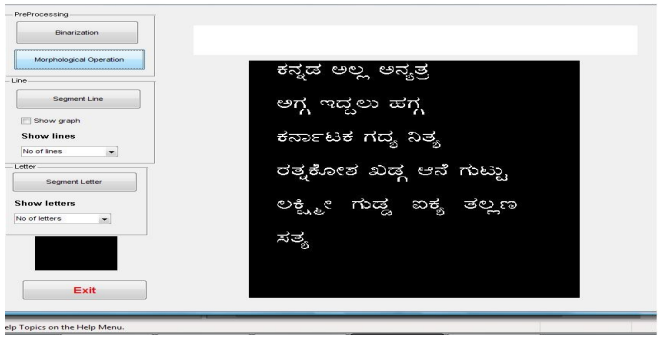

Fig. 4- Preprocessed Results

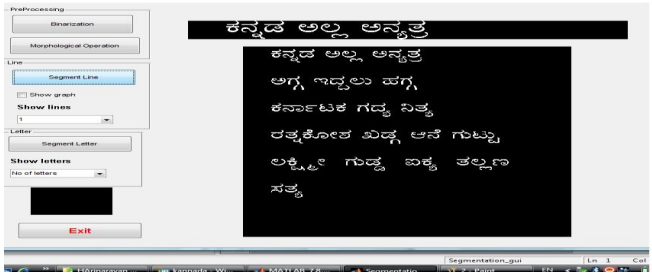

Fig. 5- Segmentation Results

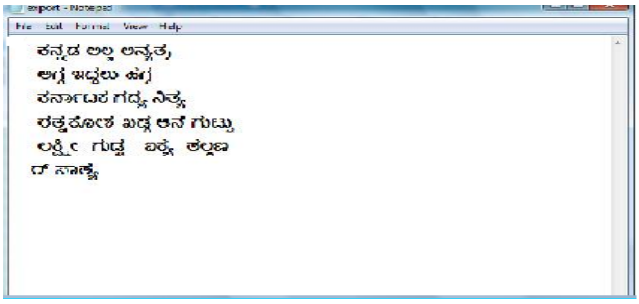

Fig. 6- Kannada character in Editable format

Table 1- Percentage of Accuracy Using SVM

Type of SVM kernel	Document Size	Recognition Rate	Recognition Time (min)
RBF kernel	25 words	95%	0.45
Linear Kernel	25 words	93%	0.3
Polynomial kernel	25 words	94%	0.35

Conclusion

This paper describes a simple and efficient OCR system for printed text documents in Kannada, a South Indian language. It takes printed Kannada character as input image and converts it into machine editable format. The system is designed to be independent of the font and size of text. At the end, the paper shows some results with the system, which delivers reasonable character recognition accuracy. By using this system we can restore available ancient Kannada documents or image into machine editable format, so that we can easily analyse and understand the ancient document. We can easily manage the Kannada documents once it is converted into machine editable format.

References

[1] Sanjeev Kunte R. and Sudhaker Samuel R.D. (2007) *A simple and efficient OCR, for basic symbols in printed Kannada text*, 32, (5), 521–533.

[2] Ashwin T.V. and Sastry P.S. (2002) *A font and size-independent OCR system for printed Kannada documents using support vector machines,* 27, (1), 35–58.

[3] Sagar B.M., Shobra G., Ramakanth Kumar (2008) *WSEA Transactions*, 4(3).

[4] Vijay Kumar B. and Ramakrishnan A.G. *Machine Recognition of Printed Kannada Text*.

[5] Balletti C., Guerra F. (2009) *Image matching for historical maps comparison*, *e-perimetron*, 4(3), 180-186.

[6] Gonzalez R.C., Woods R.E. (1993) *Digital image processing.*