A complete OCR for printed Tamil text

A.G. Ramakrishnan and Kaushik Mahata
Dept. of Electrical Engg, Indian Institute of Science, Bangalore 560 012, India

Abstract: A multi-font, multi-size Optical Character Recognizer (OCR) of Tamil Script is developed. The input image to the system is binary and is assumed to contain only text. The skew angle of the document is estimated using a combination of Hough transform and Principal Component Analysis. A multi-rate-signal-processing based algorithm is devised to achieve distortion-free rotation of the binary image during skew correction. Text segmentation is noise-tolerant. The statistics of the line height and the character gap are used to segment the text lines and the words. The images of the words are subjected to morphological closing followed by connected component-based segmentation to separate out the individual symbols. Each segmented symbol is resized to a pre-fixed size and thinned before it is fed to the classifier. A three-level, tree-structured classifier for Tamil script is designed. The net classification accuracy is 99.01%.

METHODOLOGY

OCR involves skew detection and correction followed by character segmentation and recognition of segmented symbols. Operations involved in each step are elaborated below.

Skew Correction

The input binary image is first corrected for skew. We have developed a precise skew detection algorithm [1], which estimates the skew angle in two steps. A coarse estimate of the skew is obtained through *interim line* detection using Hough Transform [2]. The *interim lines* are the lines that bisect the backgrounds in between the text lines. The coarse estimate is used to segment the text lines, which are superposed on each other and the direction of the principal axis [3] of the resulting image with the larger variance is taken as the fine skew direction. The accuracy of the final estimate is $\pm$ 0.06°. A multi-rate-signal-processing based algorithm is devised to achieve distortion-free rotation of the binary image during skew correction [4].

Text Segmentation

The text lines are segmented using the horizontal projection profile of the document image [5]. Subsequently, the words are segmented using the vertical projection profile. The statistics of line-height and symbol-gap are extracted first. During text line segmentation, the average line height is used to split those pairs of text lines, which cannot be segmented separately due to noise. Since some of the Tamil characters are made up of 2 or 3 disconnected symbols, we use the term symbol to denote each connected component, as different from a character. The symbol-gap statistics is used to distinguish a word boundary from a symbol boundary. From the segmented words, individual symbols are separated by successive application of the morphological closing and connected component-based segmentation algorithm [2]. Morphological closing helps in filling the gaps in the broken characters. Connected Component Analysis is useful when the symbols cannot be segmented using vertical projection profile only.

The case for a tree structured classifier for Tamil characters

The segmented symbols are fed to the classifier for recognition. We use a classification strategy, which first identifies the individual symbols, and in a subsequent stage, combines the appropriate number of successive symbols to detect the character. It is desirable to divide the set of 154 different symbols into a few smaller clusters, so that the search space while recognition is smaller, resulting in lesser recognition time and smaller probability of confusion. The above objective is accomplished by designing a three level, tree structured classifier to classify Tamil script symbols.

First Level Classification based on Height

The text lines of any Tamil text will have three different segments. We name them *Segment-1*, *Segment-2*, and *Segment-3*, as shown in Fig.1. Since the segments occupied by a particular symbol are fixed and remain invariant from font to font, a symbol can be associated with one of the four different *classes* depending upon its occupancy of these *segments*. Symbols occupying *segment-2* only are labeled as *Class-0* symbols. Those occupying *segment-2* and *segment-1* are termed as *Class-1* symbols. Those occupying *segment-2* and *segment-3* are named as *Class-2* symbols. Symbols occupying all of them are called as *Class-3* symbols. Almost all the symbols

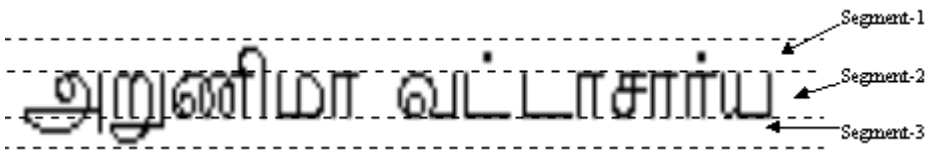
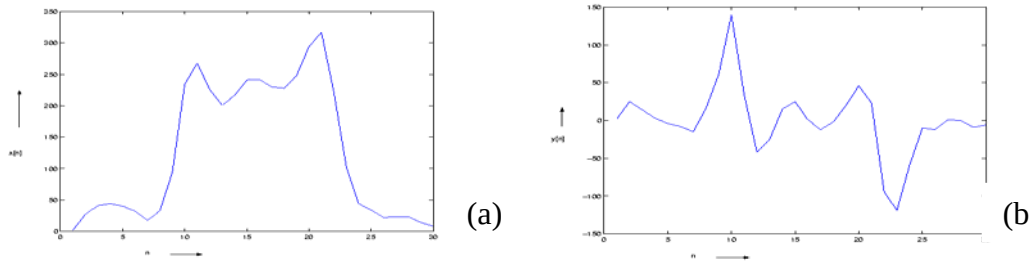


Figure.1. Segments in a Tamil



(a) (b

Figure 2. (a) Horizontal Projection Profile of a text line.
(b) First difference of the profile in (a).

in Tamil occupy the *segment-2* and about 60% of the symbols belong to *Class-0*. Thus, the horizontal projection value of any row in the *segment-2* is large compared to that of a row of the *segments 1* or *3*. The sharp rise and the fall in the horizontal projection profile p[n] indicate the transition from segment-1 to segment-2 and the transition from segment-2 to segment-3 respectively (Refer Fig.2.). These correspond to the sharp maximum and the minimum in its first difference q[n], which is given by

$$q[n] = p[n] - p[n-1] \text{ , } n>0 \qquad (1)$$
$$p[0] = q[0].$$

The line-boundary between the *segments 1 & 2* denoted by *Line_1* is given by the value of n for which q[n] is maximum in the upper half of the text line. Similarly, the boundary between the

segments 2 & 3 denoted by *Line_2* is given by the value of n for which q[n] is minimum in the lower half of the text line. An unknown symbol segmented from the text line under consideration can now be classified accordingly.

Second Level Clustering based on matra/extensions

Symbols of *class-1* and *class-3* have their extensions in *segment-1*. The set of symbols in *class-1* is divided into three groups (*Groups 1, 2, and 3*) based on their extensions in *segment-1* (Refer Fig. 3.). Similarly, *Class-2* symbols are clustered into five groups (*Groups 4, 5, 6, 7, and 8*) based on their extension in the *segment-3* (Refer Fig.4.). No further script dependent clustering is performed among the *Class-0* and *Class–3* symbols.

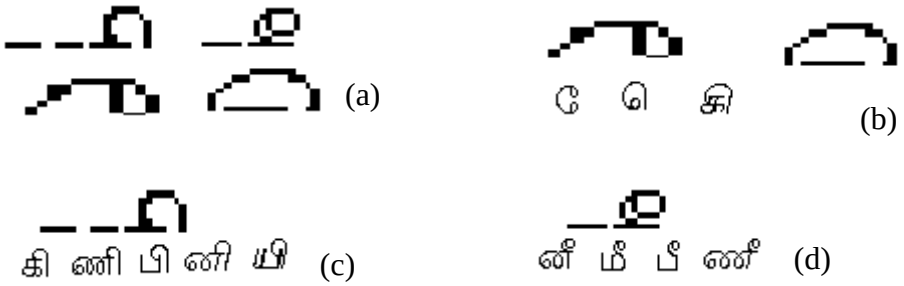


Figure 3. Illustration of second level classification in Class-1. (a) Different types of extensions of Class-1 symbols captured in segment-1 (b) Group-1 symbols and the corresponding extensions. (c) Group-2 symbols and corresponding extensions. (d) Group3 symbols and extensions

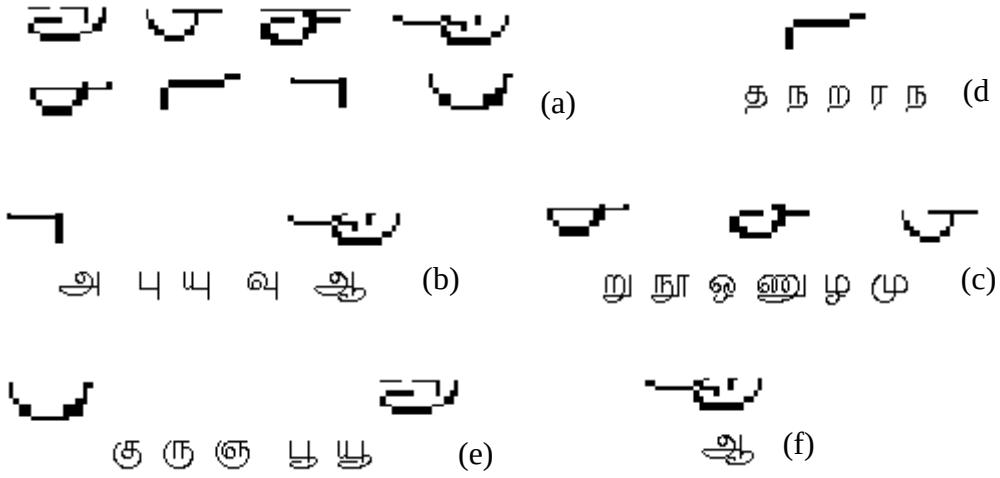


Figure 4. Illustration of second level classification in Class-2. (a) Different types of extensions of Class-2 symbols captured in segment (b) Group-4 symbols and the corresponding extensions. (c) Group-5 symbols and corresponding extensions. (d) Group-6 symbols and extensions. (e) Group-7 symbols and corresponding extensions. (f) Group-8 symbols and the corresponding extensions.

The rectangle containing the thinned symbol is found out. The portion of the rectangle captured in the *segment-1* or *3* (as applicable) is resized to a 30×30 image. This image is thinned and divided into four 15×15 blocks. Second moments [2] are calculated from each block to obtain

the 12-dimensional feature vector. Nearest neighbor classifier [6] using Euclidean distance is used for classification. Thinning algorithm proposed by Zhung and Suen [7] is employed.

The tree structure of the classifier is shown in Fig.5.

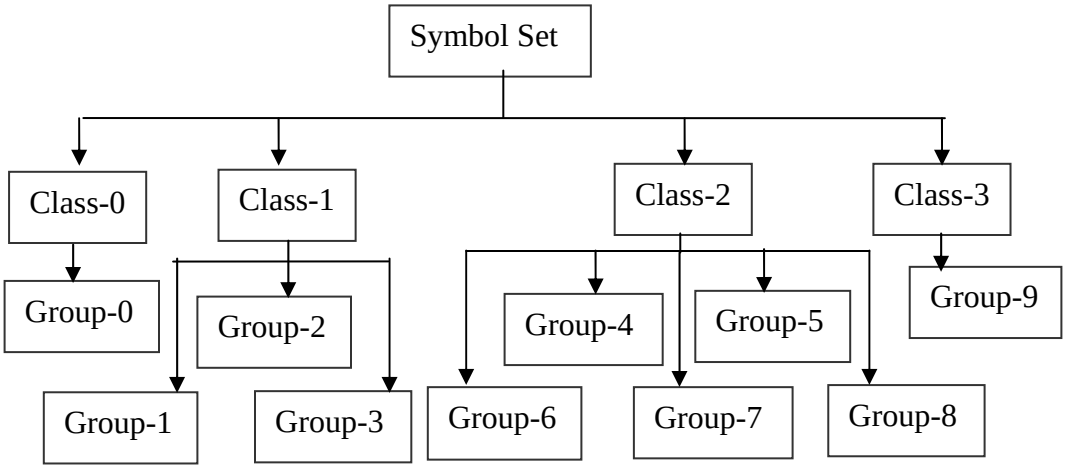


Figure.5. Tree structure of the classifier.

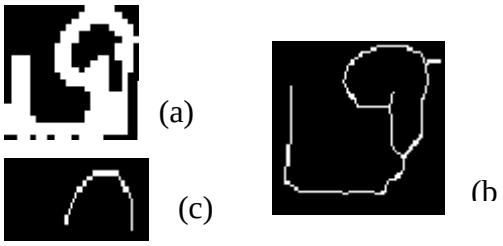


Fig.6. Example of class-1 normalization
(a) Class-1 symbol (b) Normalized symbol
(c) segment-1 extension separated.

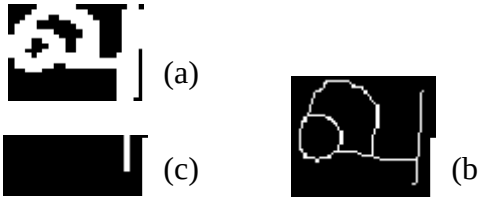


Fig.7. Example of class-2 normalization
(a) Class-2 symbol. (b) Normalized Symbol.
(c) segment-2 extension separated.

Recognition at the third level

In the third level, feature-based recognition is performed. The symbols are to be normalized first to a predefined size to make it possible to compare them with those in the training set. The normalization strategy varies from group to group. First, the rectangle containing the symbol is cropped. The cropped rectangle is interpolated to a size of 45×60 and thinned if the symbol belongs to *Class-0*. For a symbol belonging to *class-1, 2 or 3,* the portion of the cropped rectangle captured in the *segment-1* or 3 is normalized to a rectangle of height 10. The portion of the rectangle captured in the *segment-2* is normalized to a rectangle of height 50, keeping the same normalized width. These individual images are concatenated back and thinned to get the normalized symbol (Refer Figs. 6 & 7). The normalized width is 45 for *group-1*. It is 60 for the *groups 3, 4, 6, 7, 8, 9*. The width for *groups 2 and 5* is 75. This normalization strategy helps to bring in the font independence in the OCR. Geometric moment features are extracted from the normalized symbols. The normalized symbols are split into 15×15 non-overlapping blocks and from each block, second order geometric moments are calculated. Nearest neighbour classifier

using Euclidean distance is employed to recognize the symbols. A symbol is rejected if the distance to its nearest neighbour is larger than a predefined threshold. The value of the threshold is taken as 30.

Classification Results

Training set is generated form the symbols extracted from regular Tamil texts appearing in books. The algorithm is tested on some other pages of the same texts. Some of the symbols are very rare in regular Tamil texts. These symbols belong to Group-3, Group-5 and Group-9. Computer generated font is used for both the training and the test set for these symbols. The summery of the results is given in the following table. The classification accuracy is calculated based on the number of symbols correctly recognized.

	No of test patterns	No of training patterns	Percentage Recognition Accuracy	Percentage Rejection
Class-0	1832	69	99.4	0.3
Class-1	423	45	98.3	0.3
Class-2	983	69	99.3	0.4
Class-3	122	21	95.2	0.2

Net Classification accuracy is 99.01%.

References

[1] Kaushik Mahata and A. G. Ramakrishnan, *Precision Skew Detection through Principal Axis*. Proc. International Conference on Multimedia Processing and Systems, Chennai, Aug. 13-15, 2000, pp. 186-188.

[2] R.C.Gonzalez & R.E.Woods, *Digital Image Processing*. Addison-Wesley.

[3] G.Strang, *Linear Algebra and its Applications*. Academic press.

[4] Kaushik Mahata and A. G. Ramakrishnan, *A Novel Scheme for Image Rotation for Document Processing*, Proc. IEEE Intern. Conf. on Image Processing 2000, Vancouver, BC, Canada, Sept 10-13, 2000, Vol. 2, pp. 594-596.

[5] T.Akijama & N.Hagita, *Automatic entry system for printed documents.* Pattern Recognition, vol 23, pp 1141 - 1154, 1990

[6] R.O.Duda & P.E.Hart, Pattern Classification and Scene Analysis. John Wieley & Sons.

[7] T.Y.Zhung & C.Y.Suen, *A fast parallel Algorithm for thinning digital patterns*. Comm ACM, vol. 27, no. 3, pp. 337-343.