

Candidate Search and Elimination Approach for Telugu OCR

Atul Negi
Department of CIS
University of Hyderabad
Hyderabad 500046 India
atulcs@uohyd.ernet.in

Chandra Kanth Cherreddi
Department of CSE
Osmania University
Hyderabad 500007 India
chandra-kanth@ieee.org

Abstract—In this paper we propose an OCR system for *Telugu* based on the candidate search and elimination technique. The initial candidates for recognition are found by applying a zoning method on input glyphs. We propose cavities as a structural approach suited specifically for *Telugu* script, where cavity vectors are used to prune the candidates found by zoning. A final template matching stage using controlled non linear normalization is performed to conclude the search process. The search can be concluded when at any stage ever an unique candidate is found. A recognition accuracy of 97-98% was achieved on real images scanned from *Telugu* literature.

1. INTRODUCTION

Telugu is one of the prominent scripts in India and Asia, with more than 62 million speakers. While it is seen that OCR technology is in a mature stage of development for English and other Roman/Latin scripts, the progress of OCR in Asian and particularly Indian scripts is in a relatively nascent stage. One of the reasons is the complexity of the orthography, especially in *Telugu*. While potentially 10000 syllables are frequently used in the language, the orthographic units are composed by combinations of 36 consonants and 16 vowels. A practical OCR system for *Telugu* script was proposed and developed by Negi et al[3], where the complexity of *Telugu* script and methods for its reduction were proposed. Their approach consists of identification and recognition of connected components. Their recognition used a modification to the template matching approach called the fringe distance method proposed by Brown[1]. In this paper we propose an improved and robust recognition strategy which first uses the pixel distributions of the script and later exploits the structural information of *Telugu* orthography. In this paper we donot discuss layout related issues for the isolation of *Telugu* text regions, which is taken up elsewhere[4].

2. OUR APPROACH

Following the approach of Negi et al, we focus on recognizing from the order of 500 distinct glyphs, which are extracted as connected components from the input image. To do better than linear search of the template training set, we attempted a technique which is not greatly affected by the size of the training set. However, this would imply a system based on a candidate search and elimination technique. Our system consists of three stages, as shown in figure 1. The first stage uses the density of pixels in different zones for an initial candidate

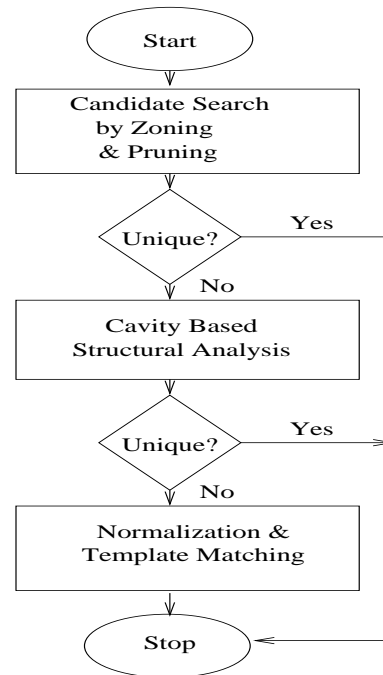
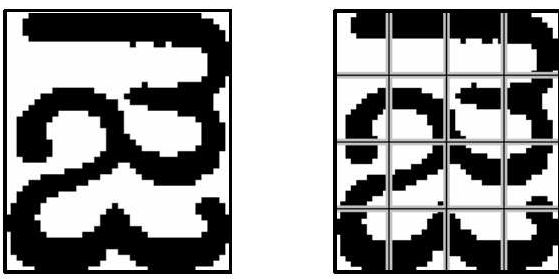


Figure 1. Schema of the OCR system.

search. The candidates obtained from this stage if found *inconclusive*, are passed to the next stage which analyzes the presence of cavities in the input image. The final stage performs template matching based on Euclidean distance on *controlled* nonlinear shape normalized characters. The normalization technique used is a modification of the techniques discussed by Lee and Park [2]. At any of these stages the search may be concluded if the pruning produces an unique result.

3. CANDIDATE SEARCH (ZONING)

For a candidate search we use the measure of density of pixel distribution in different zones of the input glyph as a feature vector. First the input glyph is broken into zones by super-imposing a grid and then the percentage of the number of foreground pixels is calculated as in figure 2. This produces a 16 (4x4) dimensional feature vector represented as $(i_1, i_2, \dots, i_{16})$ which corresponds to the grids from top left to the bottom right, in that order. A codebook of this feature vector is pre-computed from the training set. The feature vector of the input glyph is computed and searched in the codebook to obtain k (5 in our case) nearest neighbors (n). The distance measure is Euclidean Distance between the feature vectors.



(a) (b)

28.54	44.44	47.72	62.69
42.11	44.91	34.15	58.95
47.02	35.79	42.11	47.02
68.54	59.18	57.08	63.86

(c)

Figure 2. (a) Input glyph (b) Grid superimposed (c) Corresponding grid weights as percentages

After the top n results along with their distances are computed, we analyze them to eliminate a few candidates. Based on empirical evidence, candidates with a distance greater than a factor of 2.5 of the nearest neighbour distance are eliminated. The search concludes if a unique match is produced after pruning. If the search does not select a unique candidate, then the remaining candidates are passed to the next stage. This method is invariant under linear scaling as the percentage of pixels is unaffected by scaling. The present candidate search technique promises low computational complexity. The main component of time complexity in the search lies in the searching the codebook. We are presently considering techniques for reduction of complexity of the codebook search. This would facilitate good performance on multiple fonts.

4. CAVITY BASED STRUCTURAL ANALYSIS

Many *Telugu* characters have cavities (holes) in them. Cavities are used as structural features in our recognition. The existence and position of these cavities is a structurally distinguishing feature. We use cavities since they provide discrimination between glyphs which are could be very confusing for recognition. For example we show the glyphs in the figure 3.

Cavities are detected by generating a contour of the glyph and performing a connected component detection on the contour image, since cavities get disconnected from outer boundary in a contour image. The bounding box of the cavity contour should cover an area between threshold low (5%) and threshold high (25%) of the total glyph area, else it is discarded either as being too low or too large. To determine the position of the cavity we divide the total glyph area into 9 overlapping quadrants as shown in figure 4. They are numbered sequentially from 0 to 8. The existence of a cavity in these nine

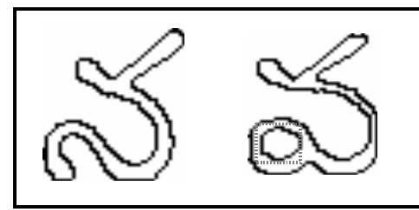


Figure 3. Glyphs which are very confusable without Cavity Analysis.

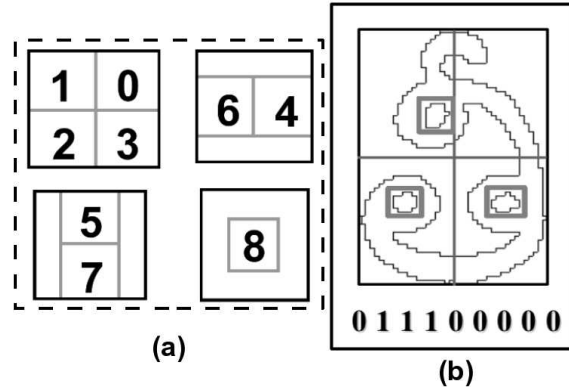


Figure 4. (a) Overlapping quadrants (b) Contour image and its cavity vector

quadrants is shown by a boolean value (since more than one cavity in a quadrant is not possible for the *Telugu* orthography) generating a 9 bit vector.

After the bit vector of the input glyph is generated, it is matched against the template bit vectors of the candidates obtained from the previous stage. If a unique match is found, the search is concluded, else the pruned results are passed onto the next stage. When no matches are found (possible if image is of poor quality or has breaks), we deal with the case explicitly. A match is performed using the logic $(\bar{x} + y)$ where the bit vector x is computed from the input glyph and y is from the template set. In the boolean operation, a 'dont care' condition exists for non-existence of cavities in certain quadrants for the input glyph. Again, if a unique result is obtained, the search is concluded, else the pruned results are passed to the final stage.

Since contour generation, and connected component analysis are low complexity algorithms, the overall complexity of this stage remains low. It is possible that more than one cavity vector (due to structural changes) is attributed to each template glyph in a multi font environment.

5. NORMALIZATION AND TEMPLATE MATCHING

We reach the final stage of the OCR, i.e template matching only if the previous stage donot conclude the search. This stage has two stages internally. The first stage is the nonlinear

normalization stage where image scaling is performed based on the image features such as projection profiles or crossing counts. This is helpful in enhancing the favorable features of the input image. We use a modification to the crossing count based technique for normalization as proposed by Lee and Park[2]. In the following we explain using their notation.

Nonlinear Shape Normalization

Let the binary input image be represented by $f(i,j)$ $i = 1,2,\dots,I$; $j = 1,2,\dots,J$ and $G(m,n)$, $m = 1,2,\dots,M$; $n = 1,2,\dots,N$ represents the output image. We normalize $f(i,j)$ to produce $G(m,n)$ which is of a desired output size. The size of the output image we have chosen to use is 64 x 64 pixels.

Feature Projection—The projection of the crossing count statistic is the basis for our normalization. Where $H(i)$ represents the horizontal crossing count while $V(j)$ represents the vertical crossing count. Where $f(i,0) = f(0,j) = 0$ and a_H and a_V are constants which affect the linearity of the statistical data.

$$H(i) = \sum_{j=1}^i \overline{f(i,j-1)} \cdot f(i,j) + a_H$$

$$V(j) = \sum_{i=1}^j \overline{f(i-1,j)} \cdot f(i,j) + a_V$$

Non Linearity Control—Lee and Park proposed in their work that the values of a_H and a_V be used to control nonlinearity when it becomes too strong. They have chosen to use the same values for both a_H and a_V and to use the same a values for all images being normalized. This assumption has failed to produce proper results for the *Telugu* orthography. The solution to this lies in using different values for a_H and a_V , which are calculated dynamically based on the input glyph.

In order to obtain favourable values for a_H and a_V we first calculate the standard deviation (σ) and mean (μ) of the $H(i)$ and $V(j)$ data assuming $a_H = a_V = 0$. We use the ratio of standard deviation to mean as a measure of non-linearity. This is because standard deviation is immune to the addition of an extra value to the each of the statistical data, while the mean is sensitive to such an addition. An upper limit is fixed for this ratio, represented by (η). The a_H and a_V values are calculated based on this maximum allowable non-linearity ratio (η). So that

$$a_H = \left[\left(\frac{\sigma_H}{\eta} \right) - \mu_H \right] \quad ; \quad a_V = \left[\left(\frac{\sigma_V}{\eta} \right) - \mu_V \right]$$

The values of a_H and a_V are thus obtained by calculating the (σ) and (μ) of $H(i)$ and $V(j)$. These values are added to the existent $H(i)$ and $V(j)$ data.

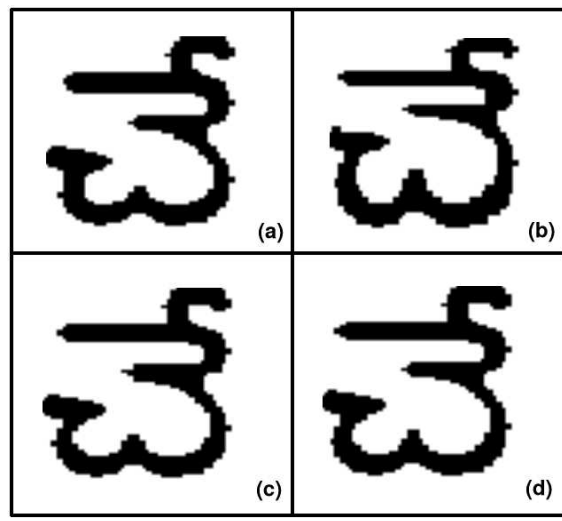


Figure 5. (a) Linear scaling (b) Nonlinear scaling (uncontrolled) (c) Nonlinear scaling $\eta=0.04$ (d) Nonlinear scaling $\eta=0.06$

Feature Density Equalization—Following Lee and Park, we resample our input image $f(i,j)$ based on the $H(i)$ and $V(j)$ to obtain the output image $G(m,n)$.

$$m = \sum_{k=1}^i H(k) \times \frac{M}{\sum_{k=1}^I H(k)}; \quad n = \sum_{l=1}^j V(l) \times \frac{N}{\sum_{l=1}^J V(l)}$$

The most important parameter which would affect all the above calculations is the value of the ratio (η). This value was chosen after careful evaluation of experimental results. We have chosen to use the value 0.06 as this produced the best overall result. Figure 5 shows a few experimental images.

The scaled input (64x64) is now passed to the second internal stage to be matched against the templates of the top n results which are also normalized using the same method as above.

Minkowski (L_2) Distance Matching

L_2 distance is more popularly known as the Euclidean distance and is a widely used measure of similarity. The summation of the L_2 distance between *each* foreground pixel of the scaled input image to the *nearest* foreground pixel in the template is called the *forward* L_2 distance. Similarly, *backward* L_2 distance is the summation of the L_2 distance between each foreground pixel of the template to the nearest foreground pixel in the scaled input image. The summation of the *forward* and *backward* L_2 distance is taken as the final comparison metric, least being the best match.

This stage has the high computational complexity owing to both the previous scaling step and the distance measure. But experimentally it is observed that not more than 20% of input

glyphs reach this stage, most get eliminated in the previous stage itself. Therefore it does not degrade the overall performance efficiency of the system.

6. FURTHER WORK

The OCR gave a good accuracy in the range of 97%-98%. In a test set of 1500 glyphs taken from a magazine (India Today, in *Telugu*) the OCR gave correct results for about 1463 glyphs. In majority of the mismatches the OCR had the correct match in the top 3 candidates. In another experiment using a different training set (from a different font) a recognition rate of 94% was obtained on the same test set. The larger errors here were because of the complete change in the structural style of a few glyphs (like non existence of a few cavities). The OCR design is such that it facilitates the use of a multi font training set. A more extensive analysis of the OCR is being performed to further fine tune all the parameters.

The true challenge for the *Telugu* OCR would be for it to work in a multi-font environment. We are presently considering the use of efficient code book search techniques and multiple cavity vectors to facilitate a multi-font OCR design. In *Telugu* orthography, the styles suffer a large variation in normal typesetting, further decorative fonts exhibit a very large variation. Future systems can be trained to keep this also in view.

REFERENCES

- [1] R.L Brown. The fringe distance measure: an easily calculated image distance measure with recognition results comparable to Gaussian blurring. *IEEE Trans. System Man and Cybernetics*, 24:111–116, January 1994.
- [2] S.W. Lee and J.S. Park. Nonlinear shape normalization methods for the recognition of large-set handwritten characters. *Pattern Recognition*, 27(7):895–902, July 1994.
- [3] Atul Negi, Chakravarthy Bhagavati, and B. Krishna. An OCR System for Telugu. In *Proceedings Sixth ICDAR, Seattle USA 2001*, 2001.
- [4] Atul Negi, Nikhil Shanker, and Chandra Kanth Chereddi. Localization, Extraction and Recognition of text in Telugu Document Images. In *Proceedings of the 7th ICDAR, Edinburgh, Scotland 2003*, 2003.