2009 10th International Conference on Document Analysis and Recognition

An Improved Online Tamil Character Recognition Engine using Post-Processing Methods

Suresh Sundaram, A G Ramakrishnan
Indian Institute of Science, Bangalore, India
suresh@ee.iisc.ernet.in, ramkiag@ee.iisc.ernet.in

Abstract

We propose script-specific post processing schemes for improving the recognition rate of online Tamil characters. At the first level, features derived at each sample point of the preprocessed character are used to construct a subspace using the 2DPCA algorithm. Recognition of the test sample is performed using a nearest neighbor classifier. Based on the analysis of the confusion matrix, multiple pairs of confused characters are identified. At the second level, we use script specific cues to sort out the ambiguities among the confused characters. This strategy reduces the recognition error among the confused character sets handled, by more than 4%. This approach can be applied irrespective of the nature of the classifier used for the first level of recognition, though the nature of the confusion set might vary.

1. Introduction

Tamil is a popular classical language spoken by a significant population in South East Asian countries. There are 156 distinct symbols in Tamil [1]. For the recognition of online Tamil characters, Deepu [2] uses class specific subspaces, while Niranjan et al. [1] have employed elastic matching schemes. Dinesh et al. [3] have recently proposed 'star -based' features for the same. Hidden Markov models for recognition have also been reported in [4] [5].

In this paper, we propose a two level scheme for recognizing online Tamil symbols. For the first level of classification, we propose an adaptation of the 2DPCA algorithm [6] for the extraction of features from online Tamil characters in a subspace. For the classification of a test character, we employ a nearest neighbor classifier.

It is an established fact that one way of assessing the performance of any given classifier depends on how well it can perform on an unknown test sample. To this effect, a confusion matrix is constructed with the training samples of all the 156 classes by employing the leave-one-out cross-validation (LOOCV) technique. The nearest neighbor classifier in the first stage fails to capture finer nuances between certain structural shapes that form the basic cues in making certain characters distinct. To further improve the classification accuracy of the system, it becomes imperative to design a robust, post-classification scheme (at the next level) to distinguish between visually similar misclassified characters. Hence, usage of script dependent cues becomes a necessity in developing these post-processing methods

2. Feature Extraction

The strokes of multistroke Tamil characters are first combined into a single trace, retaining the stroke order. Prior to feature extraction and recognition, the input raw character is smoothed to reduce noise. Dehooking algorithms are applied to remove any spurious hooks at the start of the character. The character is then resampled along the trace length to obtain a constant number of points, following which it is normalized by centering and rescaling.

Let the number of sample points in the preprocessed character be N_p. At each sample point (x_i, y_i), we extract a set of local features. Let F_j^i represent the j^{th} feature derived from the i^{th} sample point of the character.

2.1 Character Feature Matrix

Features corresponding to each sample point are stacked to form the rows of a matrix, referred to as the character feature matrix (CFM). The following are the features extracted.

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.65

- The normalized x and y coordinates of the sample points are used as features and are denoted by F_1^i and F_2^i.
- The distance and angle of the sample point with respect to the centroid of the character are computed to form the features F_3^i and F_4^i.
- We divide the length of the preprocessed character into 4 equal segments. The radial distance and polar angle of the sample point of the character with respect to the mean of the segment in which it lies are the features F_5^i and F_6^i.
- We relate the position of the sample point with respect to its immediate neighbors. We take a sliding window of size W (W odd) centered on the sample point and perform an n^{th} order polynomial fit on the samples within that window. We use the resulting n+1 polynomial coefficients as features. We use the values W=3, n=2 (quadratic fit) and accordingly denote the features as F_7^i, F_8^i and F_9^i.
- We separately model the x and y coordinates of the sample point by two n^{th} order autoregressive (AR) processes and use the resultant AR coefficients also as features. We employ a 2^{nd} order AR process and accordingly obtain the features F_{10}^i, F_{11}^i, F_{12}^i, F_{13}^i, F_{14}^i and F_{15}^i.

It is to be explicitly stated that for obtaining the polynomial fit and AR coefficients of the first and last sample points of the character, we have concatenated the last stroke with the first stroke. This ensures that the notion of neighborhood is not lost. The set of 15 features obtained at a sample point (x_i, y_i) are concatenated to form a feature vector FV^i of size 1×15

$$FV^i = \left[F_1^i \; F_2^i \; \ldots \; F_{15}^i \right] \tag{1}$$

We then construct the character feature matrix C by stacking the feature vectors of the sample points of the preprocessed character.

$$C = \begin{bmatrix} FV^1 \\ FV^2 \\ \ldots. \\ FV^{N_p} \end{bmatrix} \tag{2}$$

The i^{th} row of matrix C corresponds to the feature vector derived for the i^{th} sample point. Therefore, the size of C is $N_p \times 15$.

3. Recognition using 2D PCA

In the 2DPCA method [3], we project the character feature matrix C onto a set of projection axes $\{P_1, P_2, ..., P_d\}$ that maximize the total scatter of the projected samples. The projection axes are the orthonormal eigenvectors corresponding to the d largest eigenvalues of the character scatter matrix G_t defined below:

$$G_t = \frac{1}{N_T} \sum_{j=1}^{N_T} (C_j - M)^T (C_j - M) \tag{3}$$

where N_T is the total number of training samples, $\{C_1, C_2, ..., C_{N_T}\}$ are the N_T CFMs and M is the mean of the pooled training CFMs. Thus, the size of matrix G_t is 15×15.

On applying the 2DPCA technique to the character feature matrix C, we get a family of principal component feature vectors $\{Y_1, Y_2 \ldots, Y_d\}$ as defined below:

$$Y_k = CX_k, \quad k = 1, 2 \ldots d \tag{4}$$

The d principal component vectors can be stacked column-wise to form the projected feature matrix B of dimension $N_p \times d$.

$$B = [Y_1 \; Y_2 \ldots\ldots \; Y_d] \tag{5}$$

Let N_T be the total number of training CFMs. After transformation by 2DPCA, we get N_T projected feature matrices.

$$B_i = [Y_1^i, Y_2^i, \ldots.. Y_d^i] \quad i = 1, 2 \ldots N_T \tag{6}$$

Let B_t be the projected feature matrix for the test character. The Euclidean distance between the projected feature matrices B_t and B_i is

$$d(B_i, B_t) = \sum_{k=1}^{d} \left\| Y_k^i - Y_k^t \right\|_2 \tag{7}$$

The test character is assigned to the class of the training sample B, that satisfies the condition,

$$d(B, B_t) = \min_i d(B_i, B_t) \tag{8}$$

4. Analysis of the Confusion Matrix

The multi level recognition engines [1] generally pass the Top N choices from a previous classifier to the next level. This approach, however, may lead to redundancy, especially when at any given level; there is a high probability that none of the samples of the remaining 155 classes get easily confused with the estimated class.

In our work, we exploit prior knowledge of this probability to circumvent the aforementioned drawback of the top N choice approach. We construct a confusion matrix from the training samples using the leave-one-out cross-validation (LOOCV) technique as shown in Figure 1. The rows and columns of the confusion matrix correspond to the true and estimated class labels, respectively.

By scanning the j^{th} column of the confusion matrix, we obtain prior information on the classes that may get misrecognized as the j^{th} class. In fact, the (i,j) element in the confusion matrix represents the number of samples of i^{th} class that get recognized as the j^{th} class. If a particular column j has only one non-zero entry corresponding to the diagonal element (j,j), then it implies that none of the other classes get misclassified as the class j. In such a scenario, there is no need to post-process test data recognized as class j. A careful analysis of the confusion matrix revealed that if less than 2.5% of the total number of training samples of i^{th} class gets misrecognized to class j, they are regarded outliers that are produced mainly due to incorrect styles of writing. Accordingly, we do not consider class i in the post processing module designed for the j^{th} class.

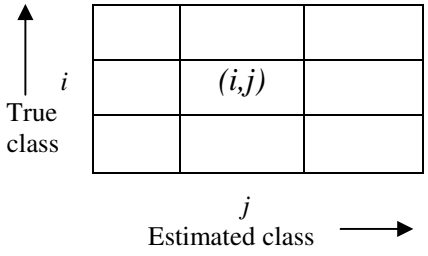

Fig. 1. Structure of the confusion matrix

Given a test character, we get its estimated class label from the first level classifier. If it corresponds to a class for which the first level classifier is highly discriminative, it is regarded as the final recognition label. Otherwise, the test data is fed to an appropriate post processing module. The output of the second classifier is the final recognition label.

5. Proposed Post-Processing Methods

The frequently confused pairs were manually grouped into two categories A and B as shown in Table 1. In this section, we propose appropriate post-processing techniques to each group of the confusion pairs.

Table 1. List of Confused Pairs

Group A	(கி,கீ) (ஙி,ஙீ) (சி,சீ) (ஞி,ஞீ) (ணி,ணீ) (தி,தீ) (நி,நீ) (பி,பீ) (லி,லீ) (வி,வீ) (ளி,ளீ) (னி,னீ) (ஜி,ஜீ)
Group B	(ஏ,ர) (ஐ,ஜ) (க,சு) (ல,வ) (ள,ன) (மு,மூ) (மு,ழு) (டு,டூ) (நு,ஞு) (நூ,ஞூ)

5.1 Disambiguating Group A Pairs

The confusions in this group appear between pairs of Tamil consonant-vowel combinations sharing the same base consonant but different vowel modifiers. The most frequently confused vowel modifiers contributing to such errors are the substrokes ◌ி and ◌ீ. Popular writing styles of Tamil script demand that the vowel modifier always forms the last stroke in any multistroke consonant-vowel combination character. However, for CV combinations written as a single stroke (where the vowel modifiers get attached to the base consonant), one can regard the subset of sample points traced before the final PEN UP to be the vowel modifier. The number of such sample points is chosen to be a function of the length of the character. It is worth re-emphasizing that the confused pairs in Group A correspond to CV combinations sharing the same base consonant (BC). Let ω_1 and ω_2 denote the class labels of BC+◌ி and BC+◌ீ combinations, respectively. We outline below the algorithm employed for distinguishing BC+◌ி and BC+◌ீ.

For a preprocessed character (BC+ Vowel modifier combination) resampled to N_p points, let $S = \{(x_i, y_i)\}_{i=b}^{N_p}$ denote the pen coordinates of the extracted vowel modifier. Here 'b' denotes the pen position of the start of the vowel modifier. A point (x_i, y_i) in S is said to be an

'interest point' if the following two conditions are satisfied.

$$\text{(i)} \quad y_i < y_{i-1} \text{ and } y_i < y_{i+1}$$
$$\text{(ii)} \quad x_{i+1} < x_i \qquad (9)$$

1) Find the sample point (x_s, y_s) satisfying the relation $y_s = \max_{i>=b} y_i$ (see Fig. 2)

2) Starting from (x_s, y_s), move along the trajectory to locate interest points, if any. Let N denote the number of interest points encountered. If $N > 0$, assign the character to class ω_2. If N=0, we invoke (3).

3) Locate the sample point (x_m, y_m) satisfying the relation $x_m = \max_{i>s} x_i$.

Define the ratio $$r = \frac{x_m - x_{N_p}}{x_m - x_b} \qquad (10)$$

If $r \geq \varepsilon$ and $y_{N_p} > y_b$ assign the character to class ω_2; else, assign it to class ω_1. ε is a threshold, empirically set to a value of 0.02.

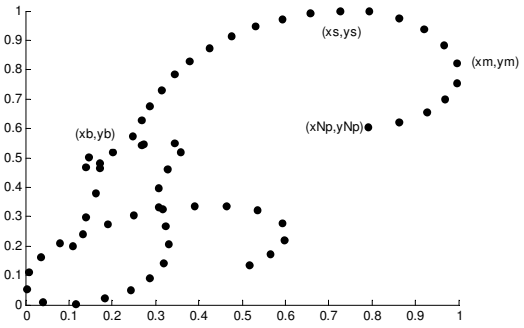

Figure 2: Extraction of script specific features from Group A. (This sample is assigned to class கீ)

To illustrate this scheme, consider the character (BC+ Vowel modifier combination) shown in Figure 2. Analysis of the vowel modifier alone indicates that N=0, r > 0.02 and $y_{N_p} > y_b$. Accordingly, the character is assigned to class கீ.

We now give an intuitive reasoning for the proposed post-classification scheme. It is observed that in modern Tamil script, there are many lexemic styles for the sub strokes ி and ீ as shown in Fig. 3. Samples of top row correspond to writing styles of கி, while those of the bottom correspond to கீ. For these cases, mere elastic / rigid matching schemes may not be good enough in distinguishing finer nuances between the sub-strokes ி and ீ. In such scenarios, the proposed technique outperforms conventional matching schemes.

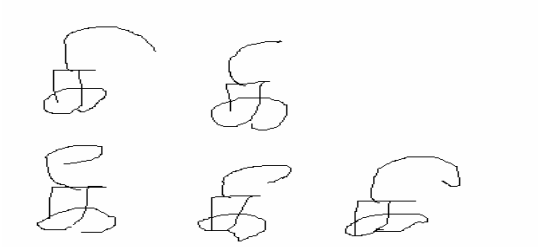

Figure 3: Lexemic styles of கி and கீ. Samples of the top and bottom rows correspond to கி and கீ, respectively.

5.2 Disambiguating Group B Pairs

For this group, Fourier descriptor features corresponding to parts of strokes that discriminate frequently confusing classes are fed to a second level classifier. Similar to Group A pairs, there exist other pairs in Group B that differ predominantly towards the end such as (ஐ,ஜ) (க,கு). However, the structure to be analyzed for these pairs is strikingly different from those treated in Group A. Moreover, there are certain characters that differ either at the start or middle of their trace such as (ள,ன) (ல,வ) (ரு,ஞ). Such pairs are also incorporated in Group B (See Table 1).

As an illustration, consider the characters ஐ and ஜ. Instead of feeding the (x,y) coordinates of these characters as a whole to the post-processing module, we focus on the shape of sub strokes forming the tails of these characters and extract Fourier descriptors from them, after resampling the extracted shape to 30 points. The number of Fourier coefficients chosen is set empirically to 10. A nearest neighbor classifier is used to obtain the final recognition label of a test character.

6. Experimental Results

The proposed two level recognition technique is tested on the IWFHR 2006 Tamil Competition dataset [7]. This dataset contains 26926 random test samples (approximately 177 test samples per character). We have used 270 training samples for each character. The characters are resampled to 60 points and normalized to [0, 1]. We compute features listed in Section 2.1 at each point of the resampled characters used for

training and construct character feature matrices of size 60 × 15. We then transform the features to an 8-dimensional subspace by performing 2DPCA on the training CFMs. A nearest neighbor classifier is used to classify the test character in the subspace. If the estimated class label is one of the confusion pairs in Table 1, we input the test character to an appropriate post-processing scheme at the next level.

Table 2 depicts the increase in the classification accuracy of a few frequently confused characters after the post processing step. The improvement in performance is observed in both the validation/ training and test sets. Validation on the training set is performed using the leave-one-out cross-validation (LOOCV) technique. On the average, there is an 8 % reduction in the recognition error among the confused characters in the validation set. On the test set, the improvement in recognition is around 5%.

On the IWFHR Test Set, we see that with 2DPCA+NN classifier alone, a recognition accuracy of 86.5% is achieved. However incorporation of post-processing schemes for confused pairs improves the performance by approximately 1%. The marginal improvement can be attributed to the fact; the current work has been concentrated solely on resolving pairs of confused characters. From the confusion matrix, there are many triplets and quadruples of confused characters, that are yet to be disambiguated such as (க,ச,சு) (ன,ள,ண) and (மு,மூ,மு,மூ) (கி,கீ,சி,சீ). Efforts are currently underway in this direction.

7. Conclusion

In this work, we have adopted the 2DPCA +NN classifier technique as the first stage in a two-stage recognition framework for online Tamil characters. We have employed structural cues to discriminate between each of the confused pairs in the second stage. There is significant improvement in the classification accuracy of the online recognition system.

8. References

[1] Niranjan Joshi, G Sita, A G Ramakrishnan and Sriganesh Madhavanath, Comparison of Elastic Matching Algorithms for Online Tamil Handwritten Character Recognition. *Proc. Intl Workshop Frontiers Handwriting Recog.* pp 444-449, 2004.

Table 2: Improvement in classification accuracy (in percentage) of some frequently confused characters on the validation and test sets after incorporation of the post-processing scheme.

	Validation Set		Test Set	
Confused Character	No post processing	With post processing	No post processing	With post processing
கி	90.6	94.8	79.2	86.3
ஞி	84.7	94.7	89.2	91.5
ணீ	83.2	95.0	86.4	91.2
ஐ	89.1	93.0	97.7	99.4
ஜ	81.3	99.8	87.2	98.7
ழ	82.7	86.4	75.6	81.9

[2] Deepu V and Sriganesh Madhavanath. Principal Component Analysis for Online Handwritten Character Recognition, *Proc. Intl Conf .Pattern Recog*. 2: pp 327-330, 2004.

[3] Dinesh M, Sridhar M K, A Feature based on Encoding the Relative Position of a Point in the Character for Online Handwritten Character Recognition. *Proc. Intl. Conf. Doc. Anal. Recog.* Vol 2, pp 1014-1017, 2007.

[4] Alejandro H Toselli, Moises Pastor, Enrique Vidal, On-Line Handwriting Recognition System for Tamil Characters, *Proc. Iberian conf. Pattern Recog. Image Anal.,* Lecture Notes Comp. Science Vol. 4477(1), pp 370-377, 2007.

[5] Bharath A, S Madhvanath, Hidden Markov Models for Online Handwritten Tamil Word Recognition. *Proc. Intl. Conf. Doc. Anal. Recog.* Vol 1, pp 506-510, 2007.

[6] Jiang Y, David Z, Alejandro FF and Jing yu Yang, Two Dimensional PCA: a New Approach to Appearance based Face. Representation and Recognition. *IEEE Trans. PAMI*, 26 (1), pp.131-137, 2004.

[7] HP Labs Isolated Handwritten Tamil Character Dataset. http://www.hpl.hp.com/india/research/penhw-interfaces-1linguistics.html#datasets