A novel hierarchical classification scheme for online Tamil character recognition

Abstract

In this paper we propose a novel three level hierarchical classification scheme for online character recognition for Tamil, a classical Indian language. We make use of the prior knowledge of the writing rules of a Tamil character to build the first level of the classifier for which we outline two methods. The first method utilizes the quantized slope information while the other relies on the trajectory of pen motion for grouping. The number of strokes in the preprocessed character is used for grouping characters at the second level while a k-Nearest Neighbor classifier is employed at the final level. The method that uses the trajectory of the pen motion information is robust to variations in the length of the character and therefore outperforms the method using quantized slope information at the first level of the classifier thereby leading to an increase in the final classification accuracy at the third level from 85% to 94%.

1. Introduction

In online handwriting recognition a machine is made to recognize the writing as a user writes on a pressure sensitive screen with a stylus. The stylus captures information about the position of the pen tip as a sequence of points in time. This spatio-temporal information of the character being traced is the only input available to the online recognition system. Online handwriting recognition can be easily extended to customize an individual's writing. Extensive research in the past two decades has led to the development of online handwritten script recognition systems for languages like English [1], Chinese [2] and Japanese [3]. However not much attention has been given to develop similar systems appropriate to Indian languages. Symbols requiring several key strokes to define a character are a common feature of Indian languages. This attribute can be well utilized for online handwriting recognition in the Indian scenario.

In this paper we attempt to evolve an online recognition system for Tamil characters. Tamil is a popular South Indian language spoken by a significant population in countries such as Singapore, Malaysia and Sri Lanka besides India. There are totally 247 letters (consonants, vowels and consonant vowel combinations) in the Tamil alphabet. Each letter is represented either as a separate symbol or as a combination of discrete symbols, which we refer to as 'characters' in this work. Only 156 distinct symbols or 'characters' are needed to recognize all the 247 letters in the Tamil alphabet. Samples of each of these characters constitute a separate class. So far as the work on online handwriting recognition for Tamil is concerned, Aparna et al. [4] have used string matching schemes. Dimensionality reduction techniques like Principal Component Analysis have also been employed for online character recognition.

We propose a new hierarchical approach to classify Tamil characters based on the prior knowledge of the rules for writing the language. The structure at the start of each character is used for building the first level of the classifier while the number of strokes of the preprocessed character is employed to perform the grouping at the second level. This hierarchical classification strategy reduces the search space for classifying a given test character at each succeeding level.

The outline of this paper is as follows: Section 2 highlights the preprocessing steps performed on the raw character. The proposed hierarchical classification technique is outlined in Section 3. Section 4 discusses two different methods to detect the structure at the start of a Tamil character for grouping characters at the first level of the classifier. Section 5 illustrates the performance of the above methods and its overall effect on the final classification accuracy. Section 6 summarizes the proposed approach and possible avenues for further research.

2. Preprocessing

Prior to recognition, the input data is smoothed to reduce noise. Dehooking algorithms [5] are applied to

remove hooks if any that may appear at the beginning and end of the character. The character is then resampled to obtain a constant number of points uniformly sampled in space following which it is normalized by centering and rescaling.

We adopt a stroke concatenation technique based on a distance criterion. We define two strokes to be in proximity [4] of each other if the distance between a pair of points of the respective strokes is less than a pre defined threshold. Let N denote the number of strokes in the raw character. We briefly outline our stroke concatenation strategy below:

1. For a character with N=1 goto Step 4.
2. If N=2, concatenate the strokes if they are the proximity of each other. Goto Step 4.
3. If (N>2)
   (a) Concatenate the first N-1 strokes if stroke i is in the proximity of stroke i+1 for all i ∈ {1, 2… N-2}; otherwise goto Step (c).
   (b) Concatenate the $(N-1)^{th}$ stroke with the N^{th} stroke if they are in the proximity of each other. Goto step 4.
   (c) If the length of the last 2 strokes are negligible compared to the total length of the character, do not concatenate the character; otherwise reject the character.
4. End

If the character at the end of the stroke concatenation algorithm has 2 strokes, we assume the last stroke to be that of the vowel modifier. This type of stroke concatenation regards certain consonant vowel combination characters (such as பி, பீ, நூ) as being made up of a single stroke provided the proximity criterion is satisfied between all consecutive strokes in the character. Pure vowels ஈ and ஃ satisfy the 'If' condition in Step 3 (c) and so are regarded as characters with 3 strokes.

3. Proposed Hierarchical Classification

Let (x_i, y_i) denote the pen coordinate at location i of the preprocessed character P. Before outlining our hierarchical classification technique, a few definitions need to be put in perspective:

- A point (x_{i+1}, y_{i+1}) is said to be an 'ascender' with respect to its previous point (x_i, y_i) if we have $y_{i+1} > y_i$. Likewise we define (x_{i+1}, y_{i+1}) to be a 'descender' with respect to (x_i, y_i) if we have $y_{i+1} < y_i$.
- A point (x_i, y_i) is said to be a 'critical point' (CP) if any one of the two conditions (i) and (ii) are satisfied. (i) $y_i > y_{i+1}$ and $y_i > y_{i-1}$
  (ii) $y_i < y_{i+1}$ and $y_i < y_{i-1}$.

The aforementioned definitions hold good irrespective of the relation between x_i and x_{i+1}.

A careful examination of the Tamil characters reveals that the locus traced by the stylus around the starting point of a character is either a line, semi loop or a loop. We make use of this important structural observation to classify the 156 characters into two groups at the first level – Group 1 comprising of characters such as ங, ச, ப for which the locus is a line and Group 2 comprising of characters for which the locus is either a semi loop or a loop like அ, ல, ஞ. Characters under Group 1 can be further subdivided depending on whether the second sample (x_2, y_2) is an ascender or descender with respect to the first sample (x_1, y_1). For simplicity, we assign a variable B_1 to take one of the 3 values for a given preprocessed character P.

$$B_1 = \begin{cases} 0 & \text{if P is assigned to Group 1 and } (x_2, y_2) \text{ is an ascender with respect to } (x_1, y_1). \\ 1 & \text{if P is assigned to Group 1 and } (x_2, y_2) \text{ is a descender with respect to } (x_1, y_1). \\ 2 & \text{if P is assigned to Group 2} \end{cases} \quad (1)$$

The number of strokes of the preprocessed character P (identified by counting the number of pen down and pen up events) is considered for grouping at the second level and a variable B_2 is assigned as below

$$B_2 = \begin{cases} 0 & \text{if P has 1 stroke.} \\ 1 & \text{if P has 2 strokes.} \\ 2 & \text{if P has 3 strokes} \end{cases} \quad (2)$$

Table 1 shows the grouping of Tamil characters for valid combinations of B_1 and B_2 .Given a test character, it is assigned to one of the groups in Table 1 for final classification at the third level.

4. Structural Recognition of Tamil characters

In this section, we explore two different methods to detect whether the Tamil character starts with a semi loop, loop or a line and thereby to accordingly assign a value to B_1. In effect, we are building the first level of the classifier. The first method outlined in Section 4.1 groups the characters on the basis of the quantized slope information which we refer to it as the 'Chain code method' in this work. The other method discussed in Section 4.2 uses the pen motion dynamics as features and so is named as 'Dynamics of Pen Motion method' for sake of simplicity.

Table 1. Grouping of the 156 Tamil characters

B_2	B_1	
0	0	க ங ச த ந ர ற க்ஷ ஙி ஙீ கு ஙு சு து நு ரு று கூ ஙூ சூ தூ நூ ரூ றூ ா
0	1	ட ப ம ய ழ டி பி யி டீ பீ யீ டு பு மு யு ழு டூ பூ மூ யூ ழூ
0	2	அ ஆ இ உ எ ஏ ஐ ஒ ஓ ஔ ண ல வ ள ன ஸ ஷ ஜ ஹ லி வி ஸி லீ வீ ஸீ ஞு ணு லு வு ளு னு ஞூ ணூ லூ வூ ளூ ணூ ெ ே ை
1	0	கி சி தி நி ரி றி க்ஷி கீ சீ தீ நீ ரீ றீ க்ஷீ க்ஷு க்ஷூ க் ங் ச் த் ந் ர் ற் க்ஷ்
1	1	மி ழி மீ ழீ ட் ப் ம் ய் ழ் ஸ்ரீ
1	2	ஊ ஞி ணி ளி னி ஷி ஜி ஹி ஞீ ணீ ளீ னீ ஷீ ஜீ ஹீ ஸு ஷு ஜு ஹு ஸூ ஷூ ஜூ ஹூ ஞ் ண் ல் வ் ள் ன் ஸ் ஷ் ஜ் ஹ் ஔ
2	0	ஃ
2	2	ஃஃ

4.1 Chain code method

Let N_p be the number of points in the preprocessed character P.

1 Quantize the slope angle of the segment between two consecutive points of P into 8 levels [1]. Let Q denotes the set of quantized slope values for a given P. Then we have:

$$Q = \{q_i\} \quad \text{where } i = 1,2,3,\ldots..,\ N_p$$
$$q_i \in \{0,1,2,\ldots..,7\} \qquad (3)$$

2 Examine a subset S of quantized slope values in Q and calculate the frequency of each quantized level or direction code in this subset. Subset S is formed with first *Th*% (*Th* being a user defined threshold) of the number of samples in P (N_p).

3 If the absolute difference between any consecutive quantized slope values in S is greater than a threshold *T1*, goto Step 6.

4 If (x_2, y_2) is an ascender with respect to (x_1, y_1) and the frequency of any one of the direction codes in S is more than *Th1* % (*Th1* being an empirically defined threshold), set B_1 =0. Go to Step 7.

5 If (x_2, y_2) is a descender with respect to (x_1, y_1) and the frequency of one of the direction codes in S is more than *Th1* %, set B_1 =1. Go to Step 7.

6 Set B_1 =2.

7 End

We found that this technique is very sensitive to the length of the character and may sometimes fail to correctly group a character having a longer trajectory. To illustrate this point, consider the characters ட and ழூ shown in Figure 1. We see that the character ழூ has a longer length than ட. However the number of preprocessed points in these 2 characters N_p is the same. Accordingly, there is larger spacing between successive points in ழூ than in ட. By considering a subset S of set Q, there is high probability that the character ழூ could get misclassified in Group 2 when the condition in Step 3 of the algorithm is satisfied by any 2 consecutive quantized slope values in S.

To overcome this problem, we propose the 'Dynamics of Pen Motion method' in Section 4.2 that performs grouping of characters at the first level while being robust to variations in the length of the character. This algorithm extensively makes use of the pen motion dynamics including critical points as its features and does not rely on any thresholds.

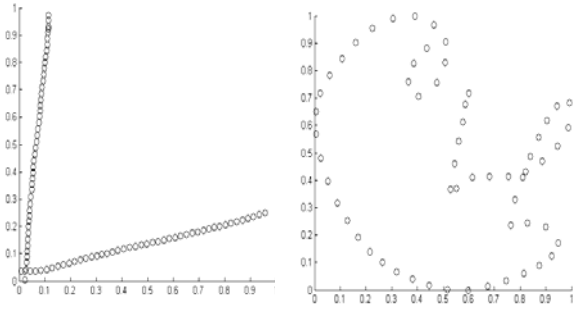

Figure 1. Character ட is longer than ழூ. ழூ has a higher chance of getting into Group 2 instead of Group 1 if any 2 consecutive quantized slope values in S are greater than a threshold T1.

4.2 Dynamics of Pen Motion method

Let N_p denote the number of sample points in the preprocessed character P and CP_i be the i^{th} encountered critical point as the loci of the pen is traced. Let the location of CP_i in P be ind(i). Thus we may write:
$CP_i = (x_{ind\,(i)}, y_{ind\,(i)})$.

1. Find the first 2 encountered critical points CP_1 and CP_2. If P has only one critical point, its last sample point is taken as CP_2.
2. Compare the x coordinate values of CP_1 and CP_2 and assign the minimum value and its

corresponding pen position in P to *xmin* and *indmin* respectively.

3. If the second sample (x_2, y_2) of P is an ascender with respect to the first sample (x_1, y_1), perform steps 4 to 6. Otherwise perform steps 8 and 9.

Steps 4 to 6 assign P to either one of the following

(i) Group 1 and sets $B_1 = 0$

(ii) Group 2 and sets $B_1 = 2$

4. Consider the segment from the first sample point to the location of the first critical point ind(1). In this segment, search for a location *loc* ($1 <= loc < ind(1)$) whose x coordinate x_{loc} is greater than $x_{ind(1)}$. If $loc \neq \varnothing$, set $B_1 = 2$ and goto step 10; else goto Step 5.
5. Starting from *indmin*, move along the trajectory to find that position *gre* whose x coordinate value x_{gre} is just greater than the x coordinate value of $\mathbf{CP_1}$, $x_{ind(1)}$. If $x_{gre} = \varnothing$, set $B_1 = 0$ and goto Step 10; otherwise goto Step 6.
6. Compare the y coordinate value at *gre*, y_{gre} with $y_{ind(1)}$, the y coordinate value of $\mathbf{CP_1}$. If $y_{gre} > y_{ind(1)}$ set $B_1 = 2$; else set $B_1 = 0$.
7. Goto step 10.

Steps 8 and 9 assign P to either one of the following

(i) Group 1 and sets $B_1 = 1$

(ii) Group 2 and sets $B_1 = 2$

8. Starting from *indmin*, move along the trajectory to find that position *gre* whose x coordinate value x_{gre} is just greater than the x coordinate value of the **first sample** x_1.
9. Compare the y coordinate value at *gre*, y_{gre} with y_1, the y coordinate value of **first sample**. If $y_{gre} > y_1$, set $B_1 = 2$; otherwise set $B_1 = 1$.
10. End

To illustrate this method, consider the preprocessed character ம shown in Figure 2.

- ம has 2 critical points $CP_1 = (0.012, 0.109)$ and $CP_2 = (0.895, 0.817)$.
- The minimum x coordinate value (*xmin*) by comparing 0.012 and 0.895 is 0.012 and its location (*indmin*) in ம is 27.
- The second sample $(x_2, y_2) = (0.101, 0.933)$ is a descender with respect to the first sample $(x_1, y_1) = (0.101, 0.961)$ since $0.961 > 0.933$. So from step 3 of the algorithm, we branch to step 8.
- We have $x_1 = 0.101$. The value of x just greater than x_1 (x_{gre}) is 0.111 and its position *gre* in the character ம is 31.
- We have $y_{gre} = 0.133$. Since $y_{gre} < y_1$, we assign the character ம to Group 1 and set $B_1 = 1$ (by Step 9).
- Stop.

Similarly for the character அ shown in Figure 3, (x_2, y_2) is an ascender to (x_1, y_1) since $y_2 > y_1$. In step 4 of the algorithm, we have $loc = \varnothing$; so we branch to Step 5 and find the location *gre* at which $x_{gre} > x_{ind(1)}$. Since $y_{gre} > y_{ind(1)}$, we conclude from step 6, that அ belongs to Group 2 and accordingly assign $B_1 = 2$.

Unlike the Chain code method, this method does not require any predefined threshold and is therefore robust to the variations in the length of the character.

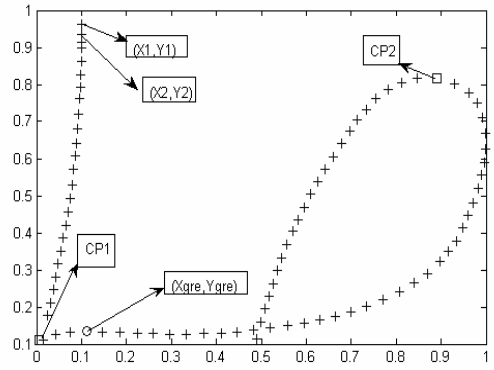

Figure 2. Character ம is assigned to Group 1 with $B_1 = 1$ since $y_{gre} < y_1$.

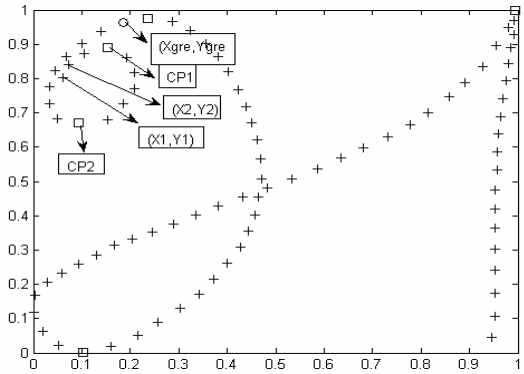

Figure 3. Character அ is assigned to Group 2 with $B_1 = 2$ since $y_{gre} > y_{ind(1)}$.

5. Experimental Results and Discussion

The data base of Tamil characters was collected from 15 native Tamil writers using a custom application running on a tablet PC. Each writer was made to input 10 samples of each of the 156 distinct characters. To avoid the problem of segmentation of the characters, users were asked to write each character in a bounding box. Samples of each of the 156 characters form a distinct class. In this section, we present our results for the writer dependent scenario. The characters are resampled to 90 points.

Given a preprocessed test character, depending on the values B_1 and B_2, it is matched against the groups in Table 1. To finally recognize this character from the subset of characters in the matched group, k Nearest

Neighbor algorithm is used. The distance from the centroid to each pen coordinate of the character is used as the feature for classification. Euclidean distance is employed as a measure of similarity for the k- NN classifier.

In the first experiment, we design the hierarchical classifier by incorporating the Chain code method to group characters at the first level. For implementing the Chain code method, we set the thresholds: Th= 10, T1=3 and Th1=80. We have used 6 training templates per class. Table 2 (a) shows the average classification accuracy across the 15 writers at each level of the hierarchical classifier. The possible factors for the lower grouping accuracy of 91.2% at the first level are:

1. A character gets wrongly grouped when the difference between consecutive slope values in the predefined subset S exceeds the threshold T1.
2. A character is assigned to an incorrect group if there is no unique directional code in S whose frequency is greater than the threshold Th1 % of the total number of quantized slope values in S. This scenario occurs when the structure at the start of the character is a steep rising or falling curve but nearly approaching a line.

The number of strokes in the preprocessed character P is used for grouping characters at the second level and there is no deviation in the grouping accuracy at this level compared to the first. The k -NN classifier is used at the final level with two different values of k (k =1 and k =5).

In the second experiment, we build the same hierarchical structure but employ the Dynamics of Pen Motion method for the first level of the classifier. This method is automated and does not require any pre defined parameters. By using the prior knowledge of the writing rules of a character, this method gives greater grouping accuracy (above 99%) at the first level thereby improving the final recognition accuracy. This is suggested by the results in Table 2 (b). Any misgrouping at the first level can be attributed to unconventional handwriting that produces non-standard structures at the start of the preprocessed character.

Some of the confusion pairs and triplets are listed below (ஒ, ஓ) (எ, ஏ) (க, ச, த) (ள, ண, ன) (ழ, ழ, ழ). Most of the ambiguity arises because these characters look visually similar and a simple k Nearest Neighbor may not be powerful enough to track minor variations that distinguish these characters.

6. Conclusions

We have proposed a novel hierarchical scheme for online Tamil characters by utilizing the a-priori structural information at the start of each character and the number of strokes involved. We adopted two methods for building the first level of the classifier – 'Chain code' method and 'Dynamics of pen motion' method and established the superiority of the latter method. In conclusion, a hierarchical structure employing the pen motion dynamics to construct the first level classifier drastically improves the final classification accuracy. Further potential areas of research are to develop a grouping strategy based on number of loops and cusps in the character and extend the idea of this work to other Indian languages.

Table 2 (a). Average classification accuracy across 15 writers with Chain code method employed at first level.

	1st level	2 nd level	3 rd level
k=1	91.2 %	91.2 %	82.3 %
k=5	91.2 %	91.2 %	86.4 %

Table 2 (b). Average classification accuracy across 15 writers with Dynamics of pen motion method employed at first level.

	1st level	2 nd level	3 rd level
k =1	99.2%	99.2%	92.2 %
k =5	99.2%	99.2%	95.3 %

References

[1] C.C.Tappert, C.Y.Suen and T. Wakahara. The state of online handwriting recognition. *IEEE Trans. on Pattern. Analysis and Machine Intelligence*, 12 (8), pp.787-807, August 1990.

[2] Cheng-Lin Liu, Stefan Jaeger and Masaki Nakagawa. Recognition of Chinese Characters: The State-of-the-Art. *IEEE Trans.on Pattern Analysis and Machine Intelligence*, 26 (2), pp.198-213, 2004.

[3] S. Jaeger, C.-L. Liu and M. Nakagawa . The state of the art in Japanese online handwriting recognition compared to techniques in western handwriting recognition . *Intl Journal on Document Analysis and Recognition*, Springer Berlin 6 (2): pp. 75-88, October 2003.

[4] K.H. Aparna, Vidhya Subramanian, M. Kasirajan, G. Vijay Prakash, V.S. Chakravarthy, Sriganesh Madhavanath. Online Handwriting Recognition for Tamil. *Proceedings of the 9th Intl Workshop on Frontiers in Handwriting Recognition* (IWFHR -9), pp 438- 443 October 2004.

[5] W. Guerfali and R. Plamondon. Normalizing and Restoring On-Line Handwriting, *Pattern Recognition*, vol. 16, no. 5, 1993.