Comparison of Elastic Matching Algorithms for Online Tamil Handwritten Character Recognition

Niranjan Joshi, G. Sita, and A. G. Ramakrishnan
Indian Institute of Science, Bangalore, India
{joshi,sita,agr}@ragashri.ee.iisc.ernet.in

Sriganesh Madhvanath
Hewlett-Packard Labs, Bangalore, India
srig@hp.com

Abstract

We present a comparison of elastic matching schemes for writer dependent on-line handwriting recognition of isolated Tamil characters. Three different features are considered namely, preprocessed x-y co-ordinates, quantized slope values, and dominant point co-ordinates. Seven schemes based on these three features and dynamic time warping distance measure are compared with respect to recognition accuracy, recognition speed, and number of training templates. Along with these results, possible grouping strategies and error analysis is also presented in brief.

Figure 1. Basic Tamil character set

1. Introduction

On-line handwriting recognition means machine recognition of the writing as user writes on a sensitive screen. Here the writing is stored as a sequence of points in time as against an image in the case of off-line handwriting recognition. Utilization of this temporal information along with spatial information can lead to better recognition rates. On-line handwriting recognition is a desirable attribute for hand held systems, where resources are limited and the devices are too small to have full sized keyboards. Cursive English script recognition is already an inbuilt feature in pocket sized Personal Digital Assistants and *tablet PC*s with very high recognition accuracies. Another useful aspect of on-line recognition is that it can be easily customized to an individual's handwriting. In the present work, we are considering only this writer dependent recognition. For a good review of on-line handwriting recognition, see [1, 2].

On-line handwriting recognition is especially relevant in the Indian scenario, because it eliminates the need to learn complex key stroke sequences and may be faster than other text input mechanisms for Indian languages. Given the complexity of entering the Indian scripts using a keyboard, handwriting recognition has the potential to simplify and thereby revolutionize data entry for Indian languages.

The challenges posed by Indian languages are different from English as not only the script size is very large but also the closeness between some of the characters call for sophisticated algorithms. In addition, unlike English and other oriental scripts, Indian scripts have compound symbols resulting from vowel-consonant combinations and in many cases, consonant-consonant combinations. Hence, the recognition strategy must take into account the size of the basic character set and the permitted compound symbols to choose between a stroke based analysis and a character based analysis. As a preliminary attempt, we use character based recognition for on-line handwriting recognition of Tamil which is a very popular south Indian language. It is also one of the official languages in countries such as Singapore, Malaysia, and Sri Lanka apart from India. Recently, the Indian Government recognized it as a classical language. There are 156 commonly used, distinct symbols in Tamil. Of these, 12 are pure vowels, 23 are pure consonants, and the remaining are vowel-consonant combinations. The pure vowels and consonants are shown in Fig. 1. Some of the similar looking pairs of characters in Tamil script are shown in Fig. 2. To the best of our knowledge, few attempts have been reported for Tamil on-line handwriting recognition. Keerthi et. el. [3] consider the problem of representation of Tamil characters. Deepu [4] considers subspace based on-line recognition of Indian scripts which

(a) (b) (c) (d) (e)

Figure 2. A few similar looking character pairs in Tamil script

includes Tamil.

This work on on-line Tamil handwriting recognition is based on template based elastic matching algorithms. Advantage of elastic matching algorithms is that they do not require a relatively large amount of training data, making them suitable for writer dependent recognition. Many such methods of elastic matching have been reported [5, 6, 7]. However these algorithms have the disadvantage that their classification time increases linearly with the number of training samples. This may prove problematic particularly for real time applications. In this paper, we present the results of our experiments on three different features and seven different recognition schemes based on dynamic time warping (DTW). The choice of these schemes and their comparison is motivated by the following factors: number of training samples used, recognition accuracy, and recognition speed. Rest of the paper is organized as follows. Section 2 gives the details of the database that we have collected. Section 3 describes preprocessing methods. Section 4 describes three different features that we have used. Dynamic time warping is introduced in brief in Section 5 along with seven different recognition schemes based on it. Section 6 gives experimental results and our discussion on it. Finally, Section 7 summarizes few important observations and concludes the paper.

2. Database Collection

The iPAQ pocket PC (ARM processor running on Win CE operating system) is used for data collection. Sampling rate for this device is 90 points per second. A database of 20 writers, each writing all 156 characters ten times, is collected. Each character is written in a separate box, which obviates the need for character segmentation. In this work, we are attempting writer dependent recognition. Therefore, out of the ten data samples of a particular class for a given writer, up to seven samples are used as training set to model a class and the rest are used for testing.

3. Preprocessing and Normalization

Pen-up and pen-down information is recorded as an integral part of the data-acquisition. This facilitates us to track the number of strokes and their order in each character. The initial number of pen-down points varies roughly between 50 and 200 points for different characters.

The input data, which is uniformly spaced in time, is resampled to obtain a constant number of points uniformly sampled in space. The total length of trajectory is divided by the number of intervals required after resampling. The resampled characters are represented as a sequence of points [x_n,y_n], regularly spaced in arc length. Uniform resampling results in 60 points for all characters. Special care has been taken for re-sampling characters with multiple strokes in order to preserve the proportion of the strokes. In such cases, the re-sampling for any given stroke is done as a fraction of the length of the stroke to the total length of the character. Two methods are tried for finding the number of re-sampled points in each stroke. One is to find the length of each stroke and re-sample each stroke according to the ratio of stroke length to total length of the character. Second method takes the average length of that stroke in the full database among the characters having same number of strokes. It is found to give better performance because it is observed that some sort of uniformity exists across all characters which have the same number of strokes.

To reduce the effect of noise, the X and Y coordinate sequences are separately smoothed, using a 5-tap low pass Gaussian filter. Each stroke is smoothed independently.

$$x_n^{filt} = w_n * x_n^{orig}$$
$$y_n^{filt} = w_n * y_n^{orig}$$
$$w_n = \frac{e^{-\frac{n^2}{2\sigma^2}}}{\sum_{n=-(N-1)/2}^{(N-1)/2} e^{-\frac{n^2}{2\sigma^2}}}$$

where, w_n's are filter coefficients. The characters are normalized by centering and rescaling. Features obtained from the thus pre-processed characters are used for building the recognizer.

4. Feature extraction

We experiment on three different features.

- **Preprocessed x-y co-ordinates:**

The preprocessed x-y co-ordinates are used as features. Let P be the preprocessed character with 60 points.

$$P = \{p_i\}, \quad \text{where, } i = 1, \ldots, 60 \quad \text{and,}$$
$$p_i = (x_i, y_i)$$

Note that feature dimension is 120 for this case.

- **Quantized slopes:**

This feature is also referred to as direction primitives. In order to get these features, we first find the slope angle of the segment between two consecutive points of P as,

$$\theta_i = \tan^{-1}\left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i}\right)$$

The slope angle obtained is quantized uniformly into 8 levels (see Fig. 3), to make this feature invariant to noise. Let Q be the set of quantized slope values corresponding to given P. Then,

$$Q \quad = \quad \{q_i\}, \quad \text{where, } i = 1, \ldots, 60 \text{ and } q_i \in \{0, \ldots, 7\}$$

The dimension of this feature is 60.

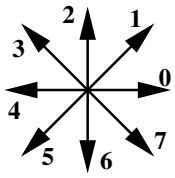


Figure 3. Quantization of slope values

- **Coordinates of Dominant points:**

Dominant points of a character P are those points where the slope values q_i change noticeably. A point p_i of P is said to be a dominant point, if the following two conditions are both satisfied:

$$(q_{i+1} - q_i + 8) \,\%\, 8 \quad \geq \quad CT \quad \text{and} \qquad (1)$$

$$(q_i - q_{i+1} + 8) \,\%\, 8 \quad \geq \quad CT \qquad (2)$$

where, CT is *Curvature Threshold*, $2 \leq i \leq 59$, and $\%$ is $modulo$ operator. CT is the threshold for retaining any point as a dominant point; it can take any value from the set $\{0, \ldots, 4\}$. By default, the first and last points of P are considered dominant points. Fig. 4 illustrates this concept. As CT increases, the structure of the character becomes coarser and vice versa. The dimension of this feature varies from one template to another, according to the inherent complexity of the character and how it has been written. This derived feature implicitly uses slope information along with the explicit use of *x-y* co-ordinate information.

Figure 4. Dominant points of a character for different FI values. (a) $CT = 0$, (b) $CT = 1$, (c) $CT = 2$.

5. Character Recognition Schemes

This section describes seven distinct recognition schemes based on DTW. Being an elastic matching technique, DTW allows to compare two sequences of different lengths. This is especially useful to compare patterns in which rate of progression varies non-linearly, which makes similarity measures such as Euclidean distance and cross-correlation unusable. Here, time alignment is carried out using dynamic programming concepts. To align two sequences A and B of lengths m and n, respectively, using DTW, we construct an $n \times m$ matrix, whose (i, j)th element contains the *cost measure* of dissimilarity between the two points a_i and b_j. Each matrix element (i, j) corresponds to the alignment between the points a_i and b_j; once this matrix is created, an optimal warping path W is selected which is a contiguous set of matrix elements that defines a mapping between A and B. The kth element of W is defined as, $w_k = (i, j)_k$, where $W = w_1, w_2, \ldots, w_k, \ldots, w_K$ and K is the length of warping path. See Fig. 5 for more details. The warping path is subjected to several constraints such as, boundary conditions, continuity, monotonicity, and windowing [8].

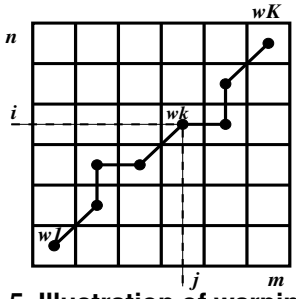


Figure 5. Illustration of warping path

The DTW dissimilarity measure (or *distance*) between two sequences A and B is given by,

$$D(A, B) \quad = \quad \min_{\mathcal{W}} \left[\sum_{k=1}^{K} w_k \Big/ K \right] \qquad (3)$$

where, $\mathcal{W}$ is a set of all possible warping paths under given constraints. The minimization in Eqn. 3 is carried out using dynamic programming techniques. The following recurrence relation is used to find the DTW distance between two sequences,

$$\gamma(i, j) \quad = \quad d(i, j) + \min\{\gamma(i-1, j), \gamma(i, j-1), \gamma(i-1, j-1)\} \qquad (4)$$

where, $\gamma(i, j)$ is cumulative distance up to the current element, $d(i, j)$ is *cost measure* of dissimilarity between ith and jth point of the two sequences.

5.1. Basic schemes

The following four schemes employ single stage procedures for recognition and hence we refer to them as *basic schemes*. The choice of these schemes is motivated by their possible use in real time applications.

- **Scheme 1:**

This scheme uses preprocessed *x*-y co-ordinates as features.

Euclidean distance is used as *cost measure* of dissimilarity between two points of feature vector. DTW distance between test pattern and templates is used in a nearest neighbor classifier. Computational complexity of this method is $O(M^2N)$ where M is the length of the sequence and N is the total number of training templates across all the classes.

- **Scheme 2:**

This scheme uses quantized slope values as features. A fixed cost matrix, given by Table 1, is used to find out *cost measure* of dissimilarity between two quantized slopes. DTW distance measure and nearest neighbor classifier are used for classification. Even though the computational complexity is the same as that for *scheme 1*, Euclidean distance calculation is replaced by the simple table look-up operation and hence overall speed is expected to improve.

Table 1. Cost measure matrix for quantized slope values

	0	1	2	3	4	5	6	7
0	0	0.4	0.7	1	1	1	0.7	0.4
1	0.4	0	0.4	0.7	1	1	1	0.7
2	0.7	0.4	0	0.4	0.7	1	1	1
3	1	0.7	0.4	0	0.4	0.7	1	1
4	1	1	0.7	0.4	0	0.4	0.7	1
5	1	1	1	0.7	0.4	0	0.4	0.7
6	0.7	1	1	1	0.7	0.4	0	0.4
7	0.4	0.7	1	1	1	0.7	0.4	0

- **Scheme 3:**

This scheme uses dominant point *x-y* co-ordinates as features. CT is set to 1. The rest of the procedure is same as that for *scheme 1*. Computational complexity for this scheme is $O(\hat{M}^2N)$ where $\hat{M} < M$. As a result, considerable improvement in the recognition speed is expected. This scheme is completely equivalent to *scheme 1* if we set CT to 0.

- **Scheme 4:**

This scheme uses preprocessed *x-y* co-ordinates as features and Euclidean distance as *cost measure*. The difference between this scheme and *scheme 1* is that here warping path is forced to follow diagonal path of the warping matrix. Hence this is a one to one or *rigid* matching scheme. Nearest neighbor classifier is used for classification purposes. Computational complexity of this scheme is $O(MN)$. Hence this is the fastest scheme among four basic schemes.

5.2. Hybrid schemes

The other three schemes are combinations of the above four basic schemes. Each of the following three *hybrid schemes* accomplishes the recognition task in two stages. The first stage is a *pre-classification* stage with low computational complexity and selects top 5 choices as its output. Second stage selects the output from these 5 choices and provides *post-classification*.

- **Scheme 5:**

This scheme uses quantized slope based classifier described in *scheme 2* at the pre-classification stage and preprocessed *x-y* co-ordinate based classifier described in *scheme 1* at the post-classification stage.

- **Scheme 6:**

This scheme uses dominant point co-ordinates based method described in *scheme 3* at both of its stages. In the pre-classification stage, CT is set to 2 and in the post-classification stage, it is reduced to 1. We refer to this scheme as *hierarchical* dominant point based scheme, where dominant points are selected by gradually reducing CT value. At high values of CT, computational complexity is low but the structure of the character is also coarse, which suits the first stage of classification.

- **Scheme 7:**

This scheme uses rigid matching scheme based on preprocessed *x-y* co-ordinates (*scheme 4*) at its pre-classification stage. Post-classification uses elastic matching scheme based on preprocessed *x-y* co-ordinates (*scheme 1*). Higher computational complexity of DTW is due to its elastic matching capability. The idea here is therefore to gradually switch from rigid matching to elastic matching.

6. Experimental Results and Discussion

In this section, we present the results of our experiments. We study the performance of these schemes with respect to three criteria: average recognition accuracy, average recognition speed, and number of training templates used per class. Average recognition accuracy is found out by dividing the number of correctly recognized test patterns by the total number of test patterns. Average recognition speed is calculated by dividing the number of test patterns recognized by the total time taken and its unit is characters per second. While the first criterion evaluates effective recognition capability of a scheme under consideration, the remaining two are important for studying effectiveness of that scheme in real time applications. All the results are obtained on a machine with Intel P4 processor (1.7 GHz) and 256 MB RAM.

First we present the results for the basic schemes (*schemes 1-4*). Our main intention to study these methods is to find their suitability for hybrid schemes. Therefore, average recognition accuracy is given up to top 5 choices. In this experiment, we use 7 training templates per class. Table 2 gives the results for the basic schemes. We notice that the recognition accuracy for *scheme 1* is the highest. However, recognition speed is the lowest. This fact is attributed to the inherent computational complexity of DTW distance based methods. Low accuracy of *scheme 2* implies that quantized slope values by themselves do not work as a feature for the problem at hand, though they provide

Table 2. Recognition results for basic schemes (20 writers; 7 training and 3 test patterns /writer)

Scheme	top 1 (%)	top 2 (%)	top 3 (%)	top 4 (%)	top 5 (%)	Speed (chs/s)
1	96.30	99.22	99.50	99.64	99.69	1.69
2	88.22	96.21	97.85	98.51	98.87	3.31
3	94.89	98.51	98.93	99.09	99.15	5.83
4	90.65	96.37	97.69	98.20	98.49	67.39

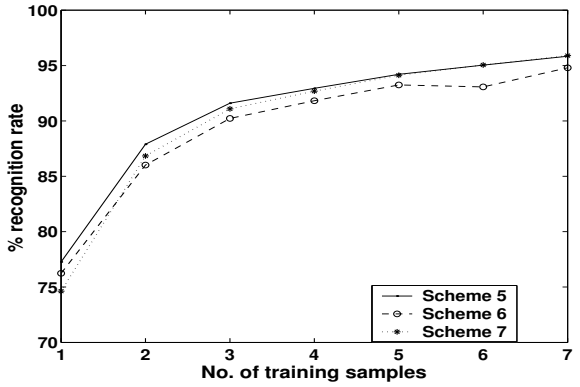


Figure 6. Recognition accuracy vs. number of training samples for the hybrid schemes

small gain in recognition speed. However, high accuracy and improved recognition speed of *scheme 3* suggests that implicit use of quantized slope values is useful for recognition. *Scheme 4* exhibits fairly low accuracy, which may be due to the fact that Tamil characters are more curved in nature and only rigid matching is not sufficient for their recognition. However, top 5 choice accuracy is reasonably high for all of them. Hence *schemes 2-4* can be used as the first stage of a two stage classification strategy.

Next, we present the results for the hybrid schemes (*schemes 5-7*). Figures 6 and 7 show the performance of hybrid schemes in terms of accuracy and recognition speed, respectively, against number of training templates used. We notice that *schemes 5* and *7* perform equally well in the region where training templates are more, with the maximum accuracy of 95.89%. However recognition speed for *scheme 5* is much lower than that of *scheme 7*. On the other hand, *scheme 6* shows lower recognition accuracies than the other two methods. This is possibly because the process of extraction of dominant points may result in removal of some useful information, which in turn leads to increased error rate. But it has better recognition speed than *scheme 5* and hence it is more useful for real time application purposes. Moreover, recognition accuracies can be improved to some extent by increasing the number of top choices selected at

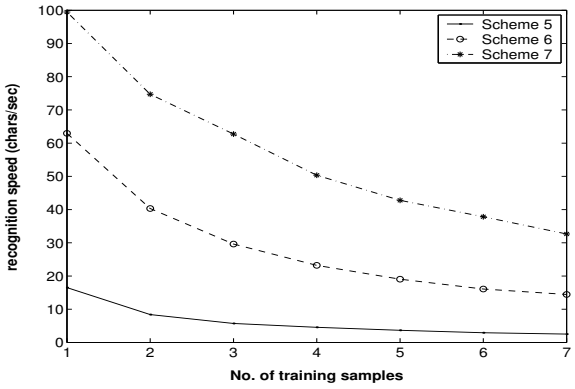


Figure 7. Recognition speed vs. number of training samples for the hybrid schemes

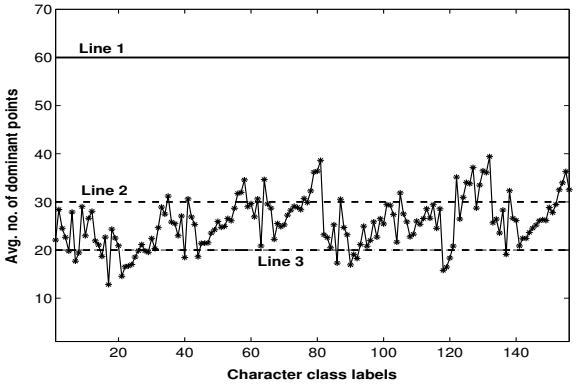
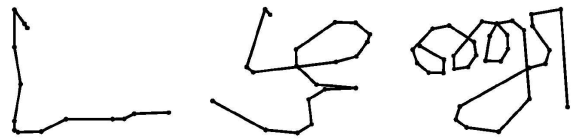


Figure 8. Average number of dominant points vs. character class

pre-classification level, at the cost of reduced recognition speed. Secondly, we notice that all the schemes show the same general behavior with respect to recognition accuracy. Graphs in Fig. 6 do not show saturation at high values of the number of training templates. And hence it is expected that, in general, recognition accuracy will improve with the increase in number of training templates, though at the cost of recognition speed. The graphs also show remarkable decrease in recognition accuracy below 3 training templates per class. A possible reason for this may be that most of the writers adopt two or more writing styles for some characters. This variability is completely disregarded when less number of training templates are used.

Fig. 8 shows how the average number of dominant points varies with character classes. We note that preprocessed characters contain 60 points (represented by *line 1*) where as the number of extracted dominant points from each character is much smaller. Specifically, a given character may contain as high as 40 and as less as 10 dominant points. Average number of dominant points that a character contains is around 25. Since dominant points are the points with "high

information" content, Fig. 8 illustrates the amount of inherent redundancy present in a character. The fairly high accuracy achieved by *scheme 3* confirms this statement. We also notice that the graph shows considerable variation along *y*-axis. And hence the number of dominant points present in a character can serve as a feature for a grouping strategy. Grouping strategy can further help to improve the recognition speed. One such possible strategy is depicted in Fig. 8. *Lines 2* and *3* divide the graph into 3 parts. Character classes from each part can be grouped together to form 3 non-overlapping groups. Fig. 9 shows one example character belonging to each such group. Another possibility is that we form 3 overlapping groups instead of non-overlapping groups.



(a) (b) (c)

Figure 9. Grouping of character classes based on dominant points. Number of dominant points are 14 in (a), 21 in (b) & 35 in (c).

We conclude this section by listing some of the prominent confusion pairs (or triplets). They include (ன, ள, ண), (ஜ், ஜீ), (வ, ல), (ழ, ழ, ழ) and (ஏ, ர). These errors are observed irrespective of the scheme used. While some of them look visually similar such as, (ன, ள, ண) and (வ, ல), others occur due to elastic matching capability of DTW viz. (ழ, ழ, ழ) and (ஏ, ர). A possible solution to overcome this problem could be to extract structural features such as loops and cusps. Another possible solution could be to use these schemes in combination with other classifiers producing non-overlapping errors.

7. Conclusions

We have described implementation details of an on-line Tamil handwriting recognition system. A comparison of seven different schemes based on three different features and dynamic time warping based distance measure has been presented. Our results show that dominant points based two-stage scheme (*scheme 6*), and combination of rigid and elastic matching schemes (*scheme 7*) perform better than rest of the schemes, especially from the point of view of implementing them in a real time application. *Scheme 6* gives 94.8% recognition accuracy with recognition speed of 14.5 chars/sec where as *scheme 7* gives 95.9% recognition accuracy with recognition speed of 32.6 chars/sec. Efforts are underway to devise character grouping schemes for hierarchical classification, and classifier combination schemes so as to obtain a computationally more efficient recognition scheme with improved accuracy. Some of our observations in this regard and possible solutions have also been presented briefly.

References

[1] Charles C. Tappert, Ching Y. Suen, and Tory Wakahara, "The state of the art in on-line handwriting recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, Aug. 1990, pp. 787-808.

[2] R. Plamondon and S. N. Srihari, "On-line and off-line handwriting recognition: A comprehensive survey," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, Jan. 2000, pp. 63-74.

[3] C. S. Sundaresan and S. S. Keerthi, "A study of representations for pen based handwriting recognition of Tamil characters," *Fifth Intern. Conf Doc. Anal. Recognition*, Sep. 1999, pp. 422-425.

[4] Deepu V., "On-line writer dependent handwriting character recognition," Master of Engineering project report, *Indian Institute of Science*, India, Jan. 2003.

[5] S. D. Connell and A. K. Jain, "Template-based on-line character recognition," *Pattern Recognition*, 34(2001), pp. 1-14.

[6] X. Li and D. Y. Yeung, "On-line handwritten alphanumeric character recognition using dominant points in strokes,"*Pattern Recog.*, vol. 30, no. 1, pp. 31-44, 1997.

[7] S. Masaki, M. Kobayashi, O. Miyamoto, Y. Nakagawa, and T. Matsumoto, "An on-line handwriting character recognition algorithm RAV (Reparamentrized Angle Variations)," *IPSJ journal*, vol. 41, no. 9, 1997, pp. 919-925.

[8] E. Keogh and M. Pazzani, "Derivative dynamic time warping," *First SIAM International Conference on Data Mining (SDM'2001)*, Chicago, USA, 2001.