Choice of classifiers in hierarchical recognition of online handwritten Kannada and Tamil aksharas

Venkatesh N
(Innovation lab, Tata Consultancy Services Limited, Bangalore, India
venk.n@tcs.com)

A G Ramakrishnan
(MILE lab, Electrical Engineering Department, Indian Institute of Science, Bangalore, India
ramkiag@ee.iisc.ernet.in)

Abstract: In this paper, we propose a novel dexterous technique for fast and accurate recognition of online handwritten Kannada and Tamil characters. Based on the primary classifier output and prior knowledge, the best classifier is chosen from set of three classifiers for second stage classification. Prior knowledge is obtained through analysis of the confusion matrix of primary classifier which helped in identifying the multiple sets of confused characters. Further, studies were carried out to check the performance of secondary classifiers in disambiguating among the confusion sets. Using this technique we have achieved an average accuracy of 92.6% for Kannada characters on the MILE lab dataset and 90.2% for Tamil characters on the HP Labs dataset.

Keywords: Handwritten character recognition, DTW, PCA, Hierarchical Classification, Kannada, Tamil.

1 Introduction

For large character set languages like Chinese and Indian languages, handwriting is a preferred alternative and a natural form of communication from humans to computers. The online handwriting recognition has a lot of potential in India, which has 18 official languages.

On-line handwriting recognition requires the user to write on a digitizing tablet using a special stylus. The user's strokes are captured as they are being formed by sampling the pen's x-y coordinates at evenly spaced time intervals. The use of a pressure sensitive switch on the pen tip indicates pen up or pen down status and aids stroke segmentation.

Very little research has been reported on online Kannada handwriting recognition. For online handwritten recognition of Tamil characters, Deepu [Deepu, 04] has used class specific subspaces, while Niranjan [Niranjan, 04] has employed elastic matching schemes. Dinesh [Dinesh, 07] has recently proposed 'star -based' features for the same. Hidden Markov models for recognition have also been reported in [Alejandro, 07] [Bharath, 07].

It has been proved that the recognition rate of the system significantly increases as more number of classifiers are used in the system [Schapire, 99]. A number of

techniques have been proposed for classifier combination [Pudil, 92] [Alpaydin, 98] [Last, 02].

Classifier fusion can be performed in two ways: serial and parallel combination. In parallel combination [Kittler, 98], each classifier performs individually on test patterns and finally the decision is made based on voting or combination rules. In contrast, in serial combination [Rahman, 99] [Last, 02], each classifier plays a significant role in reducing the number of classes and the secondary classifiers perform classification on reduced set of classes resulting in increased recognition accuracy.

In this paper, we propose a two level scheme for recognizing online handwritten symbols. The presented results indicate that by combining only primitive non-parametric classifiers, we can achieve results which are comparable to the results achieved using sophisticated classifier combination; this greatly helps in reducing the computational complexity.

Our proposed system consists of a primary classifier followed by multiple classifier-feature combinations at second stage, out of which only the best secondary classifier is selected based on prior knowledge for obtaining final output.

Prior knowledge arises from analyzing the confusion matrix built from primary classifier‘s output. Appropriate weights are assigned based on the performance of the secondary classifiers on each of the confusion sets. These weights help us in dynamically choosing the best secondary classifier based on the output of the first stage to yield the final classification output. The classification architectures for the training and testing phases are shown in Figures 3 and 4, respectively.

To the best of our knowledge, this work presents the first attempt in dynamically choosing the best secondary classifier from a group of secondary classifiers based on prior knowledge in order to disambiguate the primary classifier confusion set.

2 Online handwritten Data

Similar classification experiments have been carried out independently on distinct Kannada and Tamil online handwritten databases. The details of the two databases used for training and testing our classifier architectures are given below.

2.1 Kannada character set

Tablet PC was used to collect the data with a sampling frequency of 120 Hz. The user was asked to write each Kannada character in a separate box displayed on the tablet PC’s screen using the stylus. The captured information contains the sampled x-y coordinate values along with the writer information.

Kannada is the official language of the south Indian state of Karnataka. The language has its own script with a base set of 52 characters. This comprises 16 vowels, divided into two categories, namely, *swaragalu* (14 characters) and *yogavaahakagalu* (two characters) and 36 consonants (also called as *vyanjanagalu)*.

Further, there are base character modifiers, called consonant conjuncts (*vottaksharas)* and vowel modifiers. The numbers of these modifiers are the same as that of the base characters. The script has its own numerals. In addition to the base set of characters and numbers, the script includes special symbols, which are used in

poetry, grammar and to represent special intonation patterns for reciting Sanskrit *mantras* written in Kannada. The number of possible consonant–vowel combinations = 36 x 16 = 576. The number of possible consonant–consonant–vowel combinations = 36x36x16 = 20736. The set of all such combinations, together with the base character set are known as aksharas.

If we consider each *akshara* as a separate category to be recognized, the classifier needs to handle a huge number of classes. Also, such an approach does not exploit the basic structure of the script, that the *aksharas* are formed through well-defined geometric combinations of individual symbols. Many of the *aksharas* are very similar and differ only in having an additional stroke or an attachment. Therefore, it is good to break the *aksharas* into their constituents and recognize these components independently [Mahadeva, 09]. So, after a detailed analysis of the script, we have reduced the number of classes to about one-third the whole combinations. From this reduced subset of symbols, we can generate any akshara in the main Kannada character set. Rather than collecting all the aksharas for the on-line Kannada character data base, we have only collected the reduced symbol set of 241 consonants, 54 conjuncts including Kannada numbers from 0-9.

A database has been collected from 20 writers, each writing all the 295 symbols. Each of them wrote different number of samples per symbol, ranging from 2 to 5. In this work, we have attempted writer independent recognition which means that, the test input from a user is classified by verifying it against the training patterns comprising of handwritten data from different users.

2.2 Tamil character set

The Online handwritten Tamil Data "*hpl-tamiliwfhr06-train-online* and *"hpl-tamil-iwfhr06-testonline*" [HP, 06], that covers 156 Tamil symbols, has been independently used for testing the effectiveness of our approach for recognizing Tamil characters. The training and testing sets contain 50683 and 26926 samples respectively.

3 Preprocessing and feature extraction

3.1 Preprocessing and normalization

The captured digital ink data is preprocessed in order to remove unnecessary points in the data. The preprocessing involves smoothing, resampling and normalization. The raw data is smoothed using a moving average filter of window size 3 to remove repeated points. It is then resampled in space in order to have same number of points with equidistant spacing along the arc length for all characters. The character is resampled stroke by stroke by assigning sampling points proportional to the length of each stroke. The number of sampling points is 60 per character. The sampled character is size normalized, as shown below.

$$x_i^n = (x_i - x_{\min}) / (x_{\max} - x_{\min}) \quad (1)$$

$$y_i^n = (y_i - y_{\min}) / (y_{\max} - y_{\min}) \quad (2)$$

3.2 Feature Extraction

3.2.1 Pre-processed x-y coordinates (P_{xy})

The pre-processed x-y co-ordinates are used as features. Let P_{xy} be the pre-processed character with 60 points. $P_{xy}=\{p_i\}$, where, $i=1....60$ and $p_i=(x_i,y_i)$.

3.2.2 Quantized slope

The angle of the straight line joining any two consecutive points is given by

$$\theta_i = \tan^{-1}(y_{i+1}^n - y_i^n)/(x_{i+1}^n - x_i^n) \tag{3}$$

This angle is quantized uniformly into 8 levels, to make this feature invariant to noise [Niranjan, 04]. Let Θ be the set of quantized slope values corresponding to a given character. Then $\Theta = \{q_i\}$, where i = 1, … , 60 and $q_i \in \{0,\ldots 7\}$. The dimension of this feature vector is 60.

3.2.3 Coordinates of accurate dominant points (DP)

Dominant points are those points where the values of q_i change noticeably. Niranjan et al [Niranjan, 04] define a point p_i of P to be a dominant point d_i, if the following two conditions are satisfied:

$$\begin{aligned} &(q_{i+1} - q_i + 8)\,\%\ 8 \geq CT \quad \text{and} \\ &(q_i - q_{i+1} + 8)\,\%\ 8 \geq CT \end{aligned} \tag{4}$$

where, CT is a curvature threshold for retaining any point as a dominant point, % is the modulo operator and 8 corresponds to the number of levels of quantization of the angle. CT can take any value from the set {0,…4}. By default, the first and the last points of P are considered dominant points.

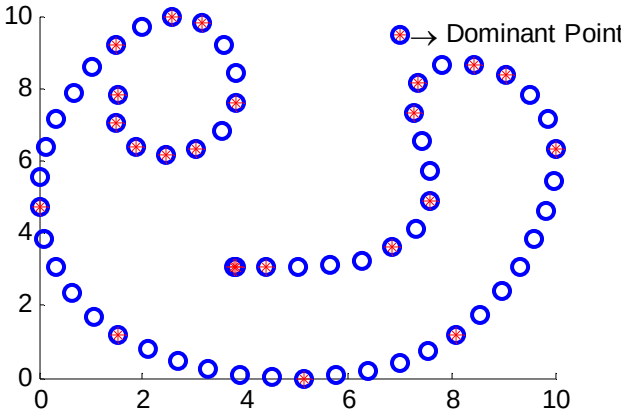
In our case, we mark the dominant point accurately based on one of the following two criteria being satisfied:

a) If the local change of slope angle (difference between the angles at the current and the previous point) is greater than experimentally found threshold of 22.5 degree or

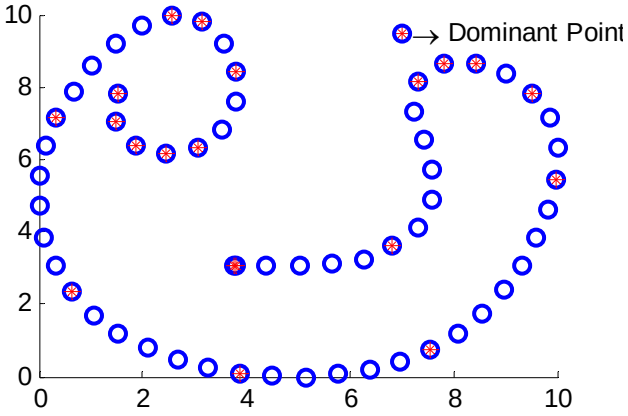
b) If the angle difference between the current point with respect to the previous dominant point exceeds the experimentally found threshold of 45 degree.

$$\begin{aligned} &|\theta_i - \theta_{i-1}| \geq 22.5^\circ \quad \text{or} \\ &|\theta_i - \theta_{DP}| \geq 45^\circ \end{aligned} \tag{5}$$

where, θ_i is the angle of the straight line joining i^{th} and $(i+1)^{th}$ pre-processed xy coordinate points and θ_{DP} is the angle calculated at previous dominant point coordinate.



(a)



(b)

Figure 1: (a) Dominant points using Equation 4. (b) Accurate dominant points using Equation 5.

3.2.4 Quartile Feature

The arc length of the character is divided into four equal segments and experimentally it was found that four segments yielded better results. The centroid is obtained for each segment. The distance of each point in a segment from its own centroid are combined into a feature vector of that particular segment. The combined features of all the four segments are called as quartile features.

Let $Q_i=\{p_{i1}\ldots\ldots p_{in}\}$ be the points in the i^{th} segment. and m_1, m_2, m_3, m_4 be the centroids of Q_1, Q_2, Q_3, and Q_4 segments, respectively. The features for the first segment are calculated as $E_1=\{e_{11}\ldots\ldots e_{1n}\}$, where e_{11} is the Euclidean distance

between p_{11} and m_1. Similarly, E_2, E_3 and E_4 are computed. The quartile feature vector is given by $\boldsymbol{E}=\{E_1, E_2, E_3, E_4\}$. The feature dimension remains constant and is equal to the dimension of the pre-processed x-y coordinates.

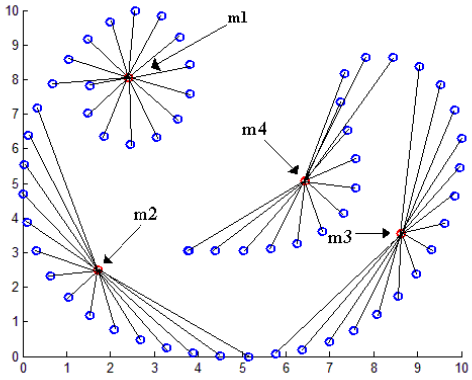


Figure 2: Computation of quartile features. (m_1, m_2, m_3, m_4 are the centroids of the 4 segments).

4 Classifiers

4.1 Dynamic Time Warping

The dynamic time warping (DTW) classifier [Niranjan, 04] employed in this work is a prototype-based classifier. This means that an unknown sample, or query, offered to the system is compared with all the prototypes in the system. The query is given the label of the nearest prototype. DTW is known to be well suited for the task of handwriting recognition.

Suppose we have two time series F = (f_1……f_n) and G = (g_1……..g_m), of length n and m, respectively. To align the two sequences using DTW, we construct an n-by-m matrix whose (i, j)th element is the Euclidean distance d(i, j) between the two points f_i and g_j. The (i, j)th matrix element corresponds to the alignment between the points f_i and g_j. A warping path, R is a contiguous set of S matrix elements that defines a mapping between F and G and is written as R = {r_1, …, r_S} where, $\max(m,n) \le S < (m+n-1)$. The warping path is typically subject to several constraints such as, boundary conditions, continuity, monotonicity, and windowing. The DTW algorithm finds the point-to-point correspondence between the curves, which satisfies the above constraints and yields the minimum sum of the costs associated with the matching of the data points. There are exponentially many warping paths that satisfy the above conditions. The path that minimizes the warping cost is,

$$D(F, G) = \min_{W} \left(\sum_{s=0}^{S} r_s \right) \tag{6}$$

where, W is a set of all possible warping paths under given constraints.
The warping path can be found efficiently using dynamic programming to evaluate a recurrence relation, which defines the cumulative distance $\gamma((i, j)$ up to the element (i,j) as the sum of *d(i,j),* the cost of dissimilarity between the i[th] and the j[th] points of the two sequences and the minimum of the cumulative distances up to the adjacent elements:

$$\gamma(i, j) = d(i, j) + \min \{\gamma(i-1, j), \gamma(i, j-1), \gamma(i-1, j-1)\} \quad (7)$$

4.2 Principal Component analysis

The principal component analysis (PCA) [Deepu, 04], a linear transformation used mainly for dimensionality reduction, maps the original N dimensional feature space to an M-dimensional space such that $M < N$. Let there be K samples of dimension N for each of C classes, given by $\{U_{11}, U_{12}, .. U_{1K}, U_{21}, U_{22}, .., U_{2K}, .. , U_{CK}\}$. The linear transformation for class 'l' leads to a new feature vector V_{il} given by,

$$V_{il} = W_l^T U_{il} \quad (8)$$

where W_l^T is the matrix with orthonormal eigen vectors corresponding to class 'l'.
The scatter matrix Z_l is given by,

$$Z_l = \sum_{i=1}^{K} (U_{il} - \mu_l)(U_{il} - \mu_l)^T \quad (9)$$

where μ_l is the mean of the K samples of class 'l'. After applying the linear transformation W_l^T, the scatter of the transformed feature vector $\{y_{i1l}, y_{il2}, .. y_{ilM}\}$ is given by $W_l^T Z_l W_l$. The projection matrix W_l is chosen to maximize the determinant of the total scatter matrix of the projected samples. That is, $W_{opt} = arg\ max| W_l^T Z_l W_l|$ and $W_l = [w_1, w_2, w_3, . w_M]$, where W_{opt} is a matrix in which each column is an N-dimensional eigenvector corresponding to one of the M largest eigen values of the total scatter matrix Z_l obtained from the demeaned training data. Then each sample is projected onto the W_{opt} matrix to get its principal components. Given a test data, it is projected onto W_{opt} of each class and the nearest neighbour rule is used for classification.

5 Classification Experiments

Initially experiments are carried out to evaluate the performance of our proposed algorithm on Kannada data sets. Similar experiments are also carried out on publicly available Tamil database to give comparative results. Kannada data set consists of data collected from 20 writers, who wrote 2 to 5 times each. Altogether there were 58 samples, among which 18 were used for testing and remaining 40 samples for training. Tamil data set is composed of 50683 training and 26926 testing samples.

5.1 Single stage classification

Pre-processed x-y coordinates feature and PCA with nearest neighbour classifier (NNC) are used in single stage classification for writer independent results. Single stage classification is performed for both Tamil and Kannada character databases in order to check its suitability for using it as a primary classifier in our multistage classification system. The reason for selecting PCA with NNC classifier is that it is a simple, non-parametric classifier and requires less computational time. Based on the first stage output, we compute the confusion matrix and carry out the error analysis in order to choose the best classifier at second stage to disambiguate the confusion and result in better recognition accuracy. This is explained in detail in the following section.

5.2 Dexterous classification

This system consists of two stage classification in which the single stage classifier explained in section 5.1 is employed as a primary classifier. For the second stage, based on a priori knowledge, the best feature-classifier combination is selected from a set of three classifier-feature combinations, to yield the final result.

Prior knowledge is gained by analysing the confusion matrix generated from the primary classifier on the training dataset. Using this prior knowledge, we test distinct secondary classifiers that work efficiently to disambiguate each confusion group.

Three different features are separately used with a DTW classifier, namely dominant points, quantized slope and quartile features, for the second stage classification. One of them is chosen dynamically as explained in the following section.

5.2.1 Study for dynamic selection of secondary classifier

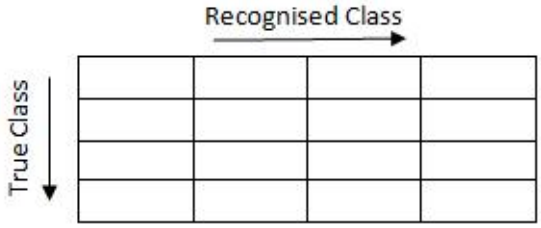


Figure 2: Confusion matrix skeleton

In the case of multi level recognition engines [Niranjan, 04], generally the top-N choices are passed on from a previous classifier to the next level. This approach, however, may lead to redundancy, especially when at any given level, for a particular recognized class, there is a high probability that it will be identified correctly without getting misclassified with the remaining classes.

Thus in our work, we exploit prior knowledge of this probability from the confusion matrix to overcome the drawback of the top-N choice approach. The confusion matrix for the primary classifier is shown in Figure 2. Diagonal elements

indicate the number of times the particular class has been identified correctly. By scanning the row and column of the confusion matrix entry corresponding to the decision of the first classifier, we obtain prior information on the classes that may get misrecognized. If a particular column or row has only one non-zero entry corresponding to the diagonal element, then it implies that there is no misclassification or confusion with other classes. In such a case, there is no need to perform second stage classification.

Error analysis for the first stage confusion matrix is as follows:

From the confusion matrix A_M , let us assume that samples of i^{th} class get misclassified to k^{th} or j^{th} class, thus forming the confusion group made up of i, j and k classes.

Let K be the number of test samples for each class and T be the number of feature-classifier combinations used at second stage to discriminate among any confused set of primary classifier.

$$h_1^i = k_1, h_2^i = k_2, h_3^i = k_3 \ldots h_T^i = k_T \tag{10}$$

where $k_1, \ k_2, k_3, \ \ldots k_T$, indicate the number of times i^{th} class was correctly recognised by the different secondary classifiers.

Depending on the performance of the secondary classifiers, appropriate weights are assoigned to them. Weight (w) is calculated as the ratio of the number of correctly recognized samples to the total number of test samples.

Then, for the i^{th} class, the secondary classifier (SC) with maximum weight is selected for yielding final recognition label.

$$SC_i = \max\left(w_1^i, w_2^i, w_3^i \ldots w_T^i\right) \tag{11}$$

Finally, depending on the first stage classification output, appropriate secondary classifier is selected. Please note that the primary classifier output is declared as the final recognition label, if that class does not have any confusion group. One such example is shown in the last row of Table 1. Table 1 provides some examples of confusion set and associated weights for choosing secondary classifiers. Figures 3 and 4 present the classification architecture for the training and testing phases, respectively.

Confusion sets			**Secondary Classifier 1 weights**	**Secondary Classifier 2 weights**	**Secondary Classifier 3 weights**
ಬೊ	ವೊ		1	0.8	0.7
ಮೆ	ನೆ		0.9	1	0.7
ವ್	ದ್	ಬ್	0.6	0.8	1
ಶಾ	ಖಾ		1	0.8	0.9
ಈ			0	0	0

Table 1: Example of Confusion sets and their corresponding secondary classifier weight

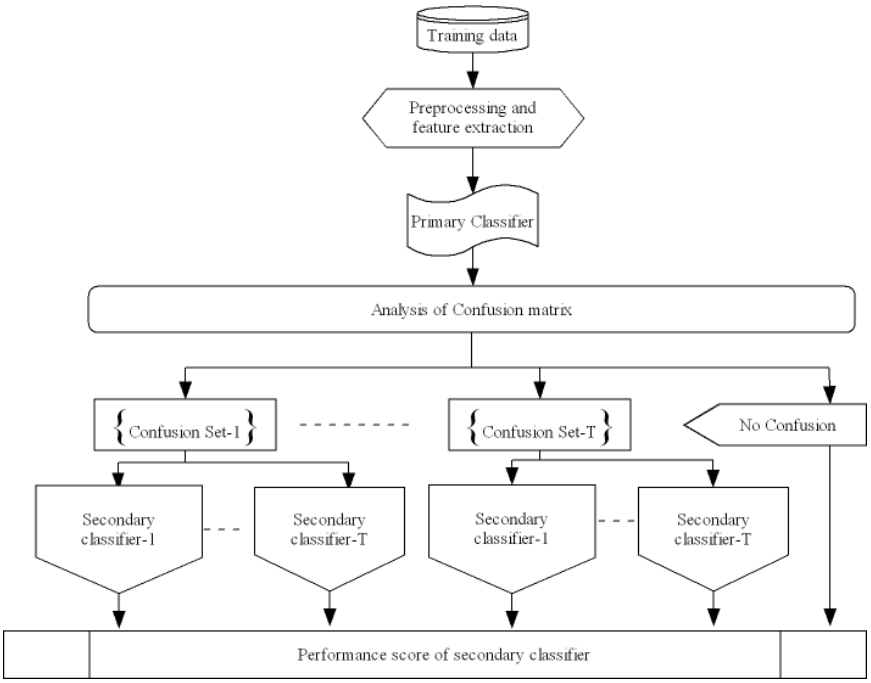


Figure 3: Classification architecture for the training phase

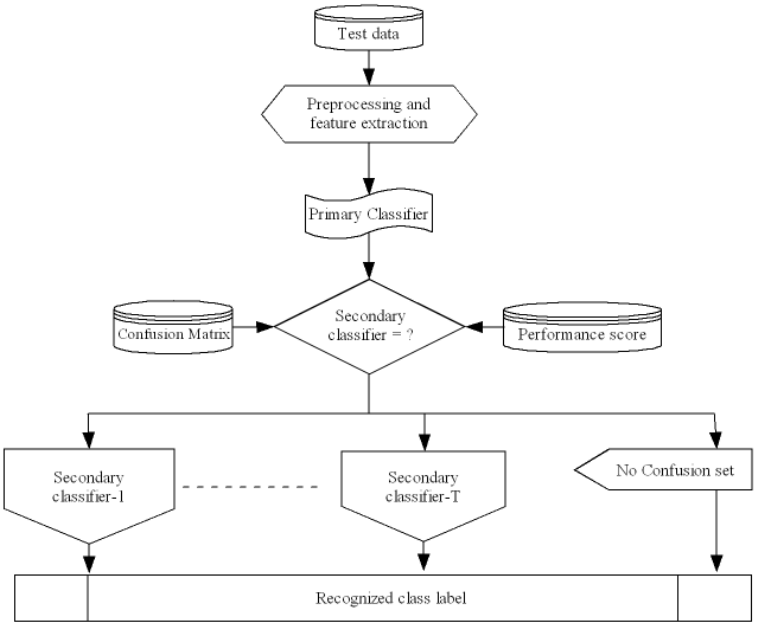


Figure 4: Classification architecture for the testing phase

6 Results and Discussion

In this section, we present the results of our experiments for single stage and dexterous classification for both Kannada and Tamil character sets. We compare the performance of the algorithms with respect to recognition accuracy.

Average recognition accuracy is the ratio of the number of correct recognitions to the total number of test patterns. All the results have been obtained on a machine with Intel Centrino Duo processor (1.66 GHz) and 1 GB RAM. Algorithms have been implemented in Matlab.

6.1 Results for Kannada character set

During training phase, we have built a confusion matrix using PCA with XY coordinate points as a feature for 295 classes. We have used only the training set consisting of 40 samples for generating the confusion matrix. From this set, 20 samples are chosen for training the primary classifier and the remaining 20 samples arr tested to generate the confusion matrix.The remaining 18 samples are used for the testing phase.

For writer independent case, it is seen that a dexterous classification with multiple classifiers at second stage yields the best recognition accuracy of 92 % for base characters and 93% for conjuncts when compared to single stage classification accuracies of 76.5 and 77.8%, respectively. Single stage and dexterous classification accuracy for Kannada characters in writer independent scenario is provided in Table 2.

From Table 2, we observe that dexterous classification has significantly increased the recognition accuracy compared to single stage classification. In multistage classification, we have overcome the problem of blindly sending top five or top ten primary classifier's output classes to the secondary classifier. Instead, we have performed a comprehensive error analysis on the output of the primary classifier by generating a confusion matrix and choosing the appropriate classifier-feature combination from multiple classifiers at second stage for improving the recognition accuracy by getting rid of the confusion sets of primary classifier.

Single stage classifier		**Dexterous classifier**	
Base characters + special symbols (241)	**Conjuncts (54)**	**Base characters + special symbols (241)**	**Conjuncts (54)**
76.5%	77.8%	92.2%	93.1%

Table 2: Comparison of recognition performances of single stage and multistage classifiers on Kannada character set. (40 samples/class in the training set and 18 samples/class for testing)

6.2 Results for Tamil character set

During training phase as a first step, a confusion matrix is built using PCA with preprocessed XY coordinate as a feature for 156 classes. Out of HP training dataset,

first 100 samples per class are chosen, among which 50 are used for training and remaining 50 samples are used for testing.

Disambiguation among each of the confused classes is attempted with each of the three classifier-feature combinations. Among these three classifiers, whichever best distinguishes among the confusion set for that particular class, is chosen for the second stage.

For actual testing, the PCA classifier is retrained with 162 samples each from the training set. The entire 26926 test samples from the IWFHR test dataset are classified by the above classifier. Now, the appropriate second stage DTW classifier checks the test sample against only the classes confused with the output of the PCA classifier. For this, all the available training samples of these confused classes are used.

Single stage and dexterous classification accuracies for Tamil characters in writer independent scenario are provided in Table 3. The overall recognition accuracy (considering both the stages) is 90.2% on the test set. The best accuracy (using both online and off-line features) on the same test set reported in the international competition on handwritten Tamil character (IWFHR 10) was 93.2%. Our recognition accuracy is comparable to the third best of the competition results. Most importantly, we have used online features and novel two-stage classification system incorporating primitive non-parametric classifiers and are able to achieve the third best result which also has far less computational complexity. The proposed system does not suffer from overfitting problem during training phase like hidden markov model (HMM), neural networks and support vector machine (SVM).

From Table 3, we can clearly infer that dexterous classification has significantly improved the recognition accuracy compared to single stage classification.

Single stage classifier	Dexterous classifier
81.1%	90.2%

Table 3: Comparison of recognition performances of single stage and multistage classifiers on Tamil character set (26926 test samples from 156 classes)

7 Conclusion and Future work

In this work, we have presented a novel dexterous classification technique which involves PCA with NN classifier adopted at the first stage followed by DTW classifier with different feature combinations at second stage.

Thorough analysis of the first stage confusion matrix yielded a clue of confusion sets which need to be appropriately handled by the secondary classifiers in order to improve the recognition accuracy. Considering this prior knowledge in choosing the appropriate classifier at the second stage has significantly improved the online handwritten recognition accuracy.

Studies are being carried out for developing a robust segmentation algorithm for obtaining symbols as a separate pattern from handwritten Kannada characters and word data so that, segmented symbols will be present in our elementary symbol set and by combining the segmented symbols based on rules, we will be able to recognize all the Kannada characters, even when the data is collected at the word level.

Acknowledgements

The authors acknowledge the funding for the collection of the data used in this study by Technology Development for Indian Languages (TDIL), DIT, MCIT, Government of India.

References

[Niranjan, 04] 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.

[Deepu, 04] Deepu V, Sriganesh Madhavanath and A G Ramakrishnan.: Principal Component Analysis for Online Handwritten Character Recognition, Proc. Intl Conf .Pattern Recog. 2: pp 327 330, 2004.

[Dinesh, 07] 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.

[Alejandro, 07] 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.

[Bharath, 07] 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.

[HP, 06] HP Labs Isolated Handwritten Tamil Character Dataset, IWFHR competition, 2006 http://www.hpl.hp.com/india/research/penhw-interfaces-1linguistics.html

[Mahadeva, 09] M Mahadeva Prasad, M Sukumar and A G Ramakrishnan.: Divide and Conquer Technique in Online Handwritten Kannada Character Recognition. Proc. Intl. Workshop on Multilingual OCR, Barcelona, Spain, july 25, 2009.

[Schapire, 99] Schapire RE. A Brief Introduction to Boosting. Proc 16th International Joint Conf on Artificial Intelligence, 1999, pp. 1-6.

[Kittler, 98] Kittler J, Hatef M, Duin RPW, and Matas J. On Combining Classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence 1998; 20 (3): 226-239.

[Alpaydin, 98] E. Alpaydin and C. Kaynak. Cascading classifiers. Kybernetika, 34:369– 374, 1998.

[Pudil, 92] P. Pudil, J. Novovicova, S. Blaha, and J. Kittler. Multistage pattern recognition with reject option. In Procs of the 11th International Conference on Pattern Recognition, pages 92–95, 1992.

[Last, 02] M. Last, H. Bunke, and A. Kandel. A feature-based serial approach to classifier combination. Pattern Analysis and Applications, 5:385–398, 2002.

[Rahman, 99] Rahman AFR, and Fairhurst MC. Serial Combination of Multiple Experts: A Unified Evaluation. Pattern Analysis and Applications 1999; 2: 292-311.