Language Models for Online Handwritten Tamil Word Recognition

Suresh Sundaram
HP Research Labs
Bangalore, India
sureshsundaram1980@gmail.com

Bhargava Urala K*
Indian Institute of Science
Bangalore, India
bhargav@mile.ee.iisc.ernet.in

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

ABSTRACT

N-gram language models and lexicon-based word-recognition are popular methods in the literature to improve recognition accuracies of online and offline handwritten data. However, there are very few works that deal with application of these techniques on online Tamil handwritten data. In this paper, we explore methods of developing symbol-level language models and a lexicon from a large Tamil text corpus and their application to improving symbol and word recognition accuracies. On a test database of around 2000 words, we find that bigram language models improve symbol (3%) and word recognition (8%) accuracies and while lexicon methods offer much greater improvements (30%) in terms of word recognition, there is a large dependency on choosing the right lexicon. For comparison to lexicon and language model based methods, we have also explored re-evaluation techniques which involve the use of expert classifiers to improve symbol and word recognition accuracies.

Keywords

Online Handwriting Recognition, Tamil Script, Statistical Language Models, Lexicon-driven recognition, Attention feedback Segmentation

1. INTRODUCTION

Work done on Tamil handwriting recognition so far has been mostly on recognition of isolated characters, focussing on extraction of different features viz. Fourier coefficients, wavelet, angular features [20, 22], PCA features [5], offline features [21]; or investigating different classifiers such as neural networks [20], HMM classifiers [22], DTW [15] and SVM [21].

The challenge of handwritten Tamil word recognition is addressed by Rituraj et. al in [13], Bharath and Madhvanath in [3] and Sundaram and Ramakrishnan in [19]. In [13], the authors reported a heuristics based segmentation of Tamil symbols but did not report the accuracy of segmentation or recognition. The work in [3] is stroke-based and employs independently built HMM-based models for symbols and inter-symbol strokes which are then concatenated to develop word models that are tested against a lexicon. In contrast, the work in [19] details a segmentation based recognition scheme for Tamil handwritten words where a word is segmented into groups of one or more strokes - which may or may not fully constitute a valid recognition primitive. The problems of over-segmentation i.e. a stroke group being part of a valid symbol and under-segmentation i.e. a stroke group being a merger of two or more valid symbols are addressed via attention-feedback segmentation (AFS) methodology where SVM classifier likelihood and features from stroke groups are used to resolve errors in segmentation.

Language models exploit linguistic regularities and characteristics derived from a corpus of text and have been used to improve word recognition accuracies in both offline and online handwriting text recognition [16, 2]. The idea of using language models to improve recognition accuracy is inspired from its extensive use in speech recognition [10]. Language models use N-gram statistics, typically generated from a corpus of text data. A study of unigram, bigram and trigram statistics for offline handwritten text [23] has found that bigram models outperform unigram models while trigram models provide marginal improvement over bigram models in terms of word recognition accuracy. As far as we know, there are no works that explore application of language models to Tamil handwritten word recognition.

Lexicon-based methods of improving recognition accuracy generally involve developing a word model for an unknown word and comparing it against a lexicon of known word models. A word model from the lexicon which is closest to the given word, measured by a suitable distance measure, is then assigned to the unknown word. In [7], context-dependent HMM models developed for sub-word sequences are used as building blocks to find a match between the unknown word model and the closest lexicon entry. The authors in [12] explore an offline handwritten word recognition system which uses a combination of N-gram language models and lexicon of varying sizes. Lexicon-driven and lexicon-free methods of word recognition for Indic scripts - Devanagari and Tamil, are explored in [4]. In this work, each word in the lexicon is represented by a sequence of HMMs, according to the standard writing order derived from the phonetic representation. Further a 'bag of symbols' representation is used to remove dependency on the writing order and rapidly prune the number of lexicon entries.

*Corresponding author

In this paper, we give a brief overview of the nature of Tamil script, our training and test datasets and the recognition process which implements attention-feedback segmentation. We then describe the development of bigram language models from a large corpus of text and their application to improve symbol and word level recognition accuracies of the attention feedback segmentation methodology to recognise Tamil handwritten words. We also discuss the improvements in symbol and word level recognition accuracies obtained by employing lexicon-based method for word recognition. Further, the results of a re-evaluation scheme to disambiguate between frequently confused symbol pairs are tabulated.

2. BACKGROUND

This section is divided into two parts:
1) Section 2.1 explains the nature and composition of Tamil script
2) Section 2.2 provides details about the isolated Tamil character database which is used for training and the word database used for testing our methods.

2.1 Description of Tamil script

Tamil script consists of 12 vowels, 18 pure consonants, 4 additional consonants and a consonant cluster (/ksha/) derived from Grantha script. The consonant-vowel (CV) combinations of these consonants along with the letter /shri/ also borrowed from Grantha script result in a total of 313 characters. However, these 313 characters can be represented by a minimal set of 155 unique symbols as demonstrated in [14].

2.2 Description of training and test datasets

IWFHR Tamil dataset is a set of 50,385 training samples and 26,926 test samples across the 155 unique symbols (described in Section 2.1) and is publicly available [8] for research. Preprocessed (refer Section 3.1) data from the IWFHR dataset is used as training data for our classifier. The test data comprises about 2000 unique words selected from a total of more than 10000 words written by high school students from across 6 educational institutions in the state of Tamil Nadu, India. The test data is partitioned into 8 distinct sets, each of size 250 words.

3. RECOGNITION SCHEME

The first task in our recognition scheme is to segment a given word $W = \{s_i\},\ i\epsilon[1, N]$ into a set of symbols $\{Si\},\ i\epsilon[1, M]$ where N is the number of strokes and M is the number of symbols in the word W. Since a symbol consists of one or more strokes, we use the words 'stroke group' (SG) to define a candidate symbol. The method of segmentation involves a combination of dominant overlap criterion segmentation (Section 3.2) and attention-feedback segmentation (Section 3.4).

Secondly, each stroke group S_i, obtained after segmentation is preprocessed (Section 3.1), recognised by the primary classifier (Section 3.3) and assigned a class label w_i. Thus, the input word is expressed as a sequence of class labels $\{w_i\}$.

The individual steps are explained in the following sections:

3.1 Preprocessing

The data acquired from the tablet PC is in the form of (x, y) coordinates along with pen-down and pen-up events. Pen-down and pen-up information is used to separate individual strokes. This data undergoes smoothing, normalising and resampling to compensate for variations in time, scale and velocity [11, 5].

- Smoothing reduces the amount of high frequency noise in the input resulting from the capturing device or jitters in writing. Each stroke is smoothed independently using a Gaussian low-pass filter.
- Normalising eliminates variability due to size differences by separately mapping both x and y coordinates to the $[0, 1]$ range by a linear transformation.
- Resampling is performed to obtain a constant number of points $n_P(= 60)$ and is done by uniform sampling along the arc length of a stroke. In case of a stroke group containing more than one stroke, the number of points alloted to each stroke is proportional to its arc length and the total number of points in that stroke group is made 60.

The final preprocessed vector of 60 pairs of (x, y) coordinates is used as the feature vector for the primary classifier.

3.2 Dominant overlap criterion segmentation

In the case of multi-stroke Tamil symbols, strokes of the same symbol may significantly overlap in the horizontal direction. This prior knowledge is utilised for initial segmentation of the input word. Horizontal overlap is mathematically quantified as follows:

$$O_k^c = \max\left(\frac{x_{max}^{S_k} - x_{min}^{s_c}}{x_{max}^{S_k} - x_{min}^{S_k}}, \frac{x_{max}^{S_k} - x_{min}^{s_c}}{x_{max}^{s_c} - x_{min}^{s_c}}\right) \quad (1)$$

where,

- s_c indicates current stroke and S_k indicates current stroke group.
- x_{max} and x_{min} indicate the bounding box extrema in the horizontal direction.

If O_k^c is greater than T_0 (a threshold set to 0.2) we merge the current stroke s_c with stroke group S_k; otherwise s_c is made to be the first stroke of a new stroke group S_{k+1}. This is repeated for all strokes in the word. The choice of threshold $T_0 = 0.2$ is made after measuring segmentation accuracies across a range of thresholds (0.1 to 0.9). The results are shown in [19]. A stroke group obtained by DOCS may or may not be a valid symbol. It could be a part of a valid symbol (oversegmentation) or a merger of two or more valid symbols (undersegmentation).

3.3 Primary classifier

The primary classifier is a support vector machine (SVM) which is trained on preprocessed (x, y) coordinates from the IWFHR dataset. A radial basis function ($c = 10, \gamma = 0.3$) is used as kernel for the SVM. The primary classifier returns recognition class labels and corresponding probability values which are used to improve the segmentation of words into symbols and hence, symbol and word recognition accuracies.

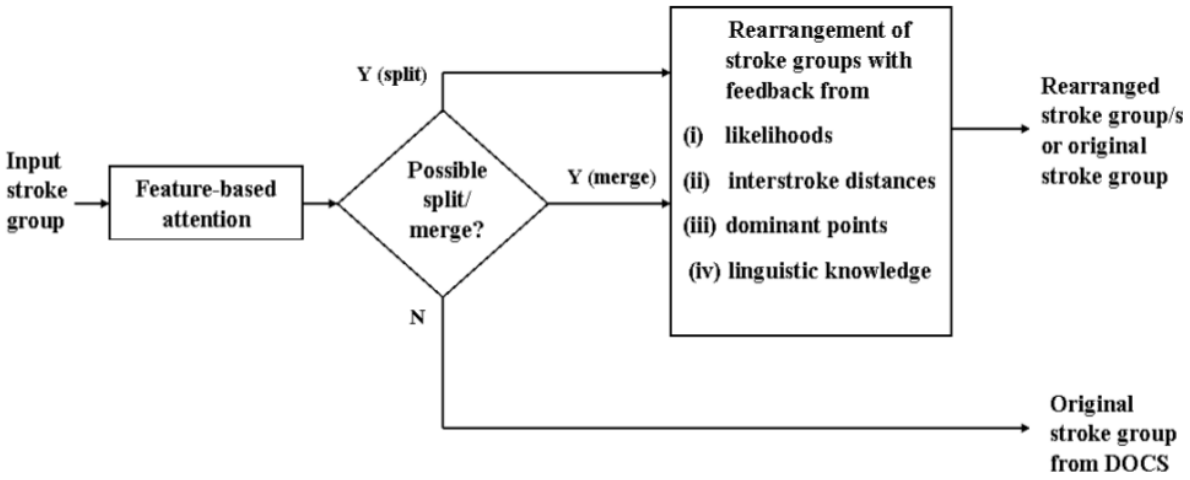


Figure 1: Overview of the attention feedback segmentation scheme

SVM classifier was implemented using the LIBSVM library [1].

SVM is chosen as the classifier, based on a comprehensive evaluation of different state-of-the-art classifiers on the test data in the IWFHR dataset reported in [19].

The term primary classifier is used to distinguish it from the secondary or expert classifiers discussed in Section 6.

3.4 Attention feedback

The problems of oversegmentation and undersegmentation are addressed by using the recognition likelihoods and certain statistical features of the stroke groups, e.g bounding box to stroke displacement in vertical and horizontal directions, horizontal displacements between strokes of a multistroke SG, obtained from the IWFHR dataset. Figure 1 shows an overview of AFS. The exact schema to detect and resolve under and over segmentation errors and features used are described in [19].

The results of attention feedback segmentation on a set of about 2000 words is shown in Table 1.

4. LANGUAGE MODELS

In this section, we describe 1) the procedure for generating bigram statistics from a large corpus of sentence data and 2) a method of using the bigram models to improve upon the recognition accuracy of the method detailed in Section 3.

4.1 Generation of bigram statistics

The data used to generate language model statistics is obtained from EMILLE corpus and our Tamil text corpus which is a collection of sentences where each word is a sequence of Tamil characters. The unicode sequence of every word is mapped to a class label sequence corresponding to the symbols used as recognition primitives. To generate bigram models we first compute the following statistics:

- N_w - total number of words in the text corpus
- $N_s(w_i)$ - total number of occurrences of symbol w_i
- $N_{ss}(w_i, w_j)$ - total number of occurrences of symbol pair (w_i, w_j)
- $N_b(w_i)$ - total number of occurrences of symbol w_i at the starting position
- $N_e(w_i)$ - total number of occurrences of symbol w_i at the last position

A specific word W can be expressed as a series of p symbols $W = w_i, i\epsilon[1, p]$. These symbols are obtained by employing the attention feedback segmentation method described in Section 3. In the bigram model, we assume that the probability of occurrence of a symbol depends only on the previous symbol. Thus, probability of the word, using a first-order Markov dependency can be written as

$$P(W) = P_b(w_1)P(w_2|w_1)...P(w_i|w_{i-1})...P(w_p|w_{p-1})P_e(w_p) \quad (2)$$

where,

$$P(w_i|w_{i-1}) = \frac{N_{ss}(w_{i-1}, w_i)}{N_s(w_{i-1})} \quad (3)$$

In order to incorporate non-zero probability values we use a smoothing technique by assuming each symbol pair has one occurrence more than the actual number [9]. The smoothing is achieved as follows,

$$P(w_j|w_i) = \frac{1 + Nss(w_i, w_j)}{155 + N_s(w_i)} \quad (4)$$

Probability of class w_i being present at the beginning and at the end of a word are computed using Equations 5 and 6 respectively.

$$P_b(w_i) = \frac{N_b(w_i)}{N_w} \quad (5)$$

$$P_e(w_i) = \frac{N_e(w_i)}{N_w} \quad (6)$$

4.2 Using language models for word recognition

Let X represent an unknown online handwritten word consisting of p symbols $S_i, i\epsilon[1, p]$ obtained by the method detailed in Section 3. Our aim now is to find a W_0 which is the most plausible sequence of symbols to represent the

Table 1: Symbol and word recognition accuracies with AFS on the eight distinct test datasets

Test Set	No. of correct symbols	Total No. of symbols	% Accuracy symbol level	No. of correct words	Total No. of words	% Accuracy word level
1	1053	1341	78.52	92	249	36.95
2	1117	1339	83.42	115	250	46.00
3	895	1470	60.88	44	250	17.60
4	1028	1437	71.54	76	250	30.40
5	1006	1330	75.64	87	250	34.80
6	1049	1443	72.70	79	249	31.73
7	1149	1330	86.39	125	250	50.00
8	1103	1455	75.81	79	250	31.60
total	**8400**	**11145**	**75.37**	**697**	**1998**	**34.88**

Table 2: Symbol and word recognition accuracies with AFS + bigram language models

Test Set	No of correct symbols	Total No. of symbols	% Accuracy symbol level	No. of correct words	Total No. of words	% Accuracy word level
1	1096	1341	81.73	104	249	41.77
2	1144	1339	85.44	124	250	49.60
3	926	1470	62.99	59	250	23.60
4	1052	1437	73.21	94	250	37.60
5	1117	1330	83.98	124	250	49.60
6	1111	1443	76.99	94	249	37.75
7	1184	1330	89.02	153	250	61.20
8	1153	1455	79.24	97	250	38.80
total	**8783**	**11145**	**78.81**	**849**	**1998**	**42.49**

word. Therefore,

$$W_0 = \arg\max_W P(W|X) \tag{7}$$

where W is the set of all likely symbol sequences that represent X. W is constructed by considering the top 3 symbols according to their likelihood as computed by the SVM.
From Bayes rule,

$$W_0 = \arg\max_W \left(\frac{P(X|W)P(W)}{P(X)}\right) \tag{8}$$

The denominator $P(X)$ is independent of W and can be ignored. Here, $P(X|W)$ represents the likelihood of the handwritten word as estimated by the SVM classifier and can be written as,

$$P(X|W) = \prod_{i=1}^{p} P(x^{S_i}|w_i) \tag{9}$$

where, x^{S_i} is the feature vector corresponding to i^{th} symbol of input word X (S_i), and w_i is the most likely class label of S_i.
$P(W)$ can be estimated by using the bigram models and start and end probability models described in Equations 4, 5 and 6. Equation 7 can be rewritten using decimal logarithms as

$$W_0 = \arg\max_W (log_{10}(P(X|W)) + log_{10}(P(W))) \tag{10}$$

The optimal sequence of symbols to represent X i.e. W_0 can be traced using the popular Viterbi algorithm [17].
Table 2 shows symbol and word level recognition accuracies after incorporating bigram language models into the AFS scheme.

5. LEXICON-BASED RECOGNITION

A lexicon of about 2.4 lakh unique words was prepared from the EMILLE and our Tamil text corpus which consist of sentences and paragraphs taken from varied sources such as newspaper articles, books and translated works. The unicode sequence of every word in the lexicon was then mapped to a class label sequence corresponding to the 155 unique symbols used as recognition primitives for the SVM. The lexicon was then divided into smaller lexicons based on the number of symbols present in each word. The algorithm for word-recognition is as follows:

- Obtain class label sequence of input word X using the method described in Section 3.
- Note down the No. of symbols p in the input word X, reported by the AFS module.
- Load lexicon L_p onto memory, where L_p indicates the lexicon of words of length p symbols.
- For every word l_i in the lexicon L_p, using appropriate distance measures e.g direct symbol matching or Levenshtein distance[6], find distance d_i between X and l_i.
- If an exact match is found i.e. if $d_i = 0$, stop the search and allot symbol sequence of i^{th} word in L_p to X
- Else find $d_{min} = \min(d_i), i\epsilon[1, N]$, where N = number of words in the lexicon L_p, and allot symbol sequence of word l_{min} to X. l_{min} is that word of lexicon L_p which has distance d_{min} from X.

Table 3: Symbol and word recognition accuracies with AFS + lexicon 1

Test Set	No of correct symbols	Total no of symbols	% Accuracy symbol level	No of correct words	Total no of words	% Accuracy word level
1	1052	1341	78.45	133	249	53.41
2	1087	1339	81.18	134	250	53.60
3	908	1470	61.77	80	250	32.00
4	971	1437	67.57	92	250	36.80
5	1014	1330	76.24	122	250	48.80
6	1029	1443	71.31	103	249	41.37
7	1115	1330	83.83	148	250	59.20
8	1082	1455	74.36	119	250	47.60
total	**8258**	**11145**	**74.10**	**931**	**1998**	**46.60**

Table 4: Symbol and word recognition accuracies with AFS + lexicon 2

Test Set	No of correct symbols	Total no of symbols	% Accuracy symbol level	No of correct words	Total no of words	% Accuracy word level
1	1153	1341	85.98	172	249	69.08
2	1205	1339	89.99	187	250	74.80
3	984	1470	66.94	114	250	45.60
4	1063	1437	73.97	130	250	52.00
5	1095	1330	82.33	158	250	63.20
6	1155	1443	80.04	148	249	59.44
7	1248	1330	93.83	202	250	80.80
8	1180	1455	81.10	161	250	64.40
total	**9083**	**11145**	**81.50**	**1272**	**1998**	**63.66**

Table 5: Symbol and word recognition accuracies for different lexicon choices, where L_p is the lexicon of words of length p symbols, where p is the number of symbols in the current test word, as reported by the AFS module

Test Set	% Symbol accuracy with Lexicon L_p	%Word Accuracy with Lexicon L_p	%Symbol Accuracy with combined lexicon L_{p-1}, L_p and L_{p+1}	%Word Accuracy with combined lexicon L_{p-1}, L_p and L_{p+1}
3	61.77	32.00	60.07	31.60
5	81.18	53.60	80.21	53.20

Table 6: Symbol and word recognition accuracies with AFS + Re-evaluation

Test Set	No of correct symbols	Total no of symbols	% Accuracy symbol level	No of correct words	Total no of words	% Accuracy word level
1	1079	1341	80.46	102	249	40.96
2	1145	1339	85.51	134	250	53.60
3	920	1470	62.59	56	250	22.40
4	1053	1437	73.28	94	250	37.60
5	1043	1330	78.42	97	250	38.80
6	1061	1443	73.53	119	249	47.79
7	1157	1330	86.99	166	250	66.40
8	1116	1455	76.70	132	250	52.80
total	**8574**	**11145**	**76.93**	**900**	**1998**	**45.05**

The results of using lexicon based word recognition methods are shown in Table 3. We found that out of the 2000 word dataset that we used for testing, only 1430 were already present in the corpus of unique words from which the lexicon was constructed. The morphologically rich nature of Tamil language results in every root verb or noun giving rise to typically hundreds of words by assimilating number, tense, gender and other language markers [18]. Therefore, the absence of about 600 words from the collection of 2.4 lakh unique words is not surprising. As an additional exercise, the remaining words of the ground truth were added to the lexicon as well and the results are tabulated in Table 4. The two lexicons are named 'lexicon 1' and 'lexicon 2' respectively. In order to verify our lexicon pruning method, an

experiment was conducted with 500 unique words, where, instead of lexicon L_p, a combined lexicon comprising L_{p-1}, L_p and L_{p+1} was chosen. It was found that there were marginal differences between the two choices as shown in Table 5.

6. RE-EVALUATION

In Tamil script there are certain symbols which are similar in nature and are therefore most likely to be confused by the SVM classifer. We have identified six such commonly occurring confusion pairs (see Table 7) and built expert classifiers which are trained on (x, y) coordinate features in the discriminating regions between the two confused symbols. An illustration of one of the confused pairs is shown in Figure 2.

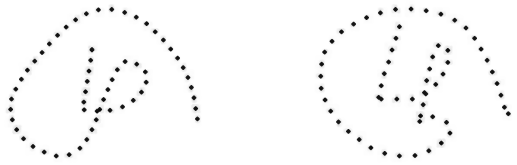
After the attention feedback segmentation, we consider the most likely class label and corresponding likelihood as obtained from the SVM for each stroke group. If any one of the symbols belong to the list of possible confused pairs, we use the expert classifier to disambiguate between the two similar symbols.

The results of re-evaluation are tabulated in Table 6.

Table 7: Commonly confused symbol pairs

Symbol pair	Total no of symbols	No of confusions	Classifier Accuracy
/mu/ and /zhu/	349	26	92.6%
/Na/ and VM of /ai/	351	32	90.9%
/Ni/ and /Li/	364	32	91.2%
/La and /Na/	353	23	93.5%
/ki/ and /ci/	355	23	93.5%
/la/ and /va/	359	14	96.1%

Figure 2: Illustration of one of the commonly confused pairs: /mu/ and /zhu/



(a) Symbol for /mu/ (b) Symbol for /zhu/

7. DISCUSSION

The class label sequence of each of the words, obtained by employing bigram language models, lexicon-based recognition or re-evaluation, is compared to the class label sequence of the ground truth of that word. The number of errors at the symbol level is the Levenshtein distance [6] between the obtained class label sequence and the ground truth sequence.

Re-evaluation techniques offer an improvement of about 2% from the AFS method by disambiguating some commonly confused pairs by making use of expert classifiers. However, it is interesting to note that the improvement in word recognition accuracy is comparable to that obtained by the use of lexicon 1 and exceeds that realized using bigram language models.

As seen from Tables 1 and 2, language models offer an improvement of about 3% at the symbol level and about 8% at the word-level. Using lexicon 1 (Table 3) resulted in an overall decrease in symbol level accuracy by about 1% whereas the word level accuracy was increased by almost 12%. Though the number of correctly detected words has increased, this suggests that some symbols correctly detected by the AFS method are being alloted incorrect labels by the lexicon method. Usage of lexicon 2 (Table 4), which contained all the words of the ground truth, shows a significant improvement at both symbol level (6%) and word level (32%) from the AFS method. This illustrates the dependency of this method on the composition of lexicon. The large improvement especially in the word recognition accuracy makes lexicon-based method a good choice for limited vocabulary word recognition problems whereas bigram language models along with re-evaluation techniques may be better suited for problems in the online handwritten Tamil word recognition domain which do not have a restricted vocabulary.

In this work, we have independently explored lexicon based and bigram statistics based approaches to improve symbol and word recognition accuracies of online handwritten Tamil words. The possibility of combining the two methods in a meaningful way could be an avenue for exploration. Further, better lexicon pruning methods to reduce the search space and time complexity for matching words could also be explored. A detailed study of symbol and word recognition accuracies of the top N choices as obtained by any of the methods or a combination thereof would be useful from a user application point of view. Bigram models in our experiment were developed from symbol level statistics, however a character in Tamil may span multiple symbols. Therefore, developing and exploring N-gram language models at the character level may also be a feasible line of work for the future.

8. ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their constructive suggestions and their valuable insights, which helped us enhance the presentation of our paper. We thank all research students and project staff at MILE lab, IISc for their support, direct or indirect, to our work. We would like to specially thank Technology Development for Indian Languages (TDIL), Department of Information Technology, Government of India for funding this research work.

9. REFERENCES

[1] *LIBSVM - A Library for Support Vector Machines.* http://www.csie.ntu.edu.tw/cjlin/libsvm/.

[2] A. Bharath and S. Madhvanath. Online handwriting recognition for Indic scripts. *Guide to OCR for Indic scripts, V Govindaraju and S Setlur*, Edn.London:209–234.

[3] A. Bharath and S. Madhvanath. Hidden Markov Models for online handwritten Tamil word recognition. In *Proc. IWFHR*, pages 506–510, 2007.

[4] A. Bharath and S. Madhvanath. HMM-based lexicon-driven and lexicon-free word recognition for online handwritten Indic scripts. *IEEE Transactions*

on Pattern Analysis and Machine Intelligence, 34(4):670–682, 2012.

[5] V. Deepu, A. G. Ramakrishnan, and S. Madhvanath. Principal Component Analysis for online Tamil handwritten character recognition. In *Proc. ICPR*, pages 327–330, 2004.

[6] Duda, Hart, and Stork. *Pattern Classification*. Springer Wiley, 1995.

[7] G. N. Fink and T. Pløtz. On the use of context-dependent modeling units for HMM-based offline handwriting recognition. In *Proc. ICDAR*, pages 729–733, 2007.

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

[9] H. Jeffreys. *Theory of Probability, Second Edition*. Clarendon Press, Oxford, 1948.

[10] F. Jelinek. Statistical methods for speech recognition. *MIT Press*, 1998.

[11] N. Joshi, G. Sita, A. G. Ramakrishnan, and S. Madhvanath. Comparison of elastic matching algorithms for online Tamil character handwriting recognition. In *Proc. IWFHR*, pages 444–449, 2004.

[12] A. L. Koerich, R. Sabourin, and C. Y. Suen. Lexicon-driven HMM decoding for large vocabulary handwriting recognition with multiple character models. *International Journal on Document Analysis and Recognition (IJDAR)*, 6(2):126–144, 2003.

[13] R. Kunwar, S. Kiran, S. Sundaram, and A. G. Ramakrishnan. A HMM based online Tamil word recognizer. In *Proc. Tamil Internet Conference*, pages 165–168, 2009.

[14] B. Nethravathi, C. P. Archana, K. Shashikiran, A. G. Ramakrishnan, and V. Kumar. Creation of a huge annotated database for Tamil and Kannada OHR. In *Proc. ICFHR*, pages 415–420, 2010.

[15] L. Prasanth, J. Babu, R. Sharma, P. Rao, and M. Dinesh. Elastic matching algorithms of online handwritten Tamil and Telugu scripts using local features. In *Proc. ICDAR*, pages 501–505, 2007.

[16] S. Quiniou, E. Anquetil, and S. Carbonnel. Statistical language models for online handwritten sentence recognition. In *Proc. ICDAR*, pages 516–520, 2005.

[17] L. Rabiner and B. Juang. An introduction to Hidden Markov Models. *IEEE ASSP Magazine*, 3(1):4–16, 1986.

[18] S. Rajendran, S. Arulmozhi, and S. Viswanathan. Computational morphology of Tamil verbal complex. *Language in India*, 3(4), 2003.

[19] S. Sundaram and A. G. Ramakrishnan. Attention-feedback based robust segmentation of online handwritten isolated Tamil words. *ACM Transactions on Asian Language Information Processing*, In press, March 2013.

[20] C. S. Sundaresan and S. S. Keerthi. A study of representations for pen based handwriting recognition of Tamil characters. In *Proc. ICDAR*, pages 422–425, 1999.

[21] H. Swethalakshmi, C. C. Sekhar, and V. S. Chakravarthy. Spatiostructural features for recognition of online handwritten characters in Devanagari and Tamil scripts. In *Proc. ICANN*, pages 230–239, 2007.

[22] A. H. Toselli, M. Pastor, and E. Vidal. On-line handwriting recognition system for Tamil characters. In *Proc. PRIA*, pages 370–377, 2007.

[23] A. Vinciarelli, S. Bengio, and H. Bunke. Offline recognition of unconstrained handwritten texts using HMMs and statistical language models. *IEEE Trans. PAMI*, 26(6):709–720, 2004.