Offline Handwritten Devanagari Word Recognition: An HMM Based Approach

Swapan Kumar Parui and Bikash Shaw

Computer Vision & Pattern Recognition Unit, Indian Statistical Institute,
203, B.T. Road, Kolkata, India, 700108
swapan@isical.ac.in, bikash_t@isical.ac.in

Abstract. A hidden Markov model (HMM) for recognition of handwritten Devanagari words is proposed. The HMM has the property that its states are not defined a priori, but are determined automatically based on a database of handwritten word images. A handwritten word is assumed to be a string of several stroke primitives. These are in fact the states of the proposed HMM and are found using certain mixture distributions. One HMM is constructed for each word. To classify an unknown word image, its class conditional probability for each HMM is computed. The classification scheme has been tested on a small handwritten Devanagari word database developed recently. The classification accuracy is 87.71% and 82.89% for training and test sets respectively.

Keywords: Hidden Markov Model(HMM), Devanagari Word Recognition, Stroke Primitives.

1 Introduction

Handwriting recognition is one of the challenging problems in Pattern Recognition. The problem has been studied for several decades and many reports on handwriting recognition in the scripts of developed nations are available in the literature. However, only a few works on handwriting recognition in Indian scripts have been reported ([1]-[3]). The present paper deals with recognition of offline handwritten Devanagari words. Works on recognition of handwritten Devanagari characters/numerals exist ([4],[5]). However, no work on handwritten Devanagari word recognition has been reported.

According to literature review there are two approaches for handwritten word recognition: local or analytical approach held at the character level [6] and global approach held at the word level [7]. The first approach deals with the segmentation problem i.e., the words are first segmented into characters or pseudo-characters, then the character model is used for recognition. Since word segmentation is itself a challenging problem, the success of recognition module depends much on segmentation performance. The second approach treats the word itself as a single entity and it goes for recognition without doing segmentation explicitly. However this approach is restricted to applications with small lexicon.

A. Ghosh, R.K. De, and S.K. Pal (Eds.): PReMI 2007, LNCS 4815, pp. 528–535, 2007.

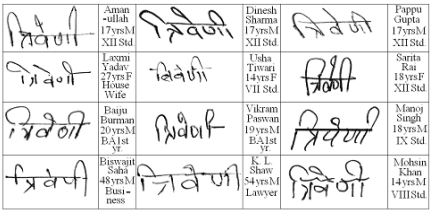
In our present work for word recognition we have applied the second approach because of two reasons: (a) to avoid the overhead of segmentation and (b) due to lack of standard benchmark database for training the classifier. Since a standard benchmark database was not availabe for Indian script so we created a word database for Devanagari to test the performance of our system. In the present report, training and test results of the proposed approach are presented on the basis of this database.

We have used a hidden Markov model (HMM) in the proposed scheme for recognition of handwritten Devanagari words. An HMM is capable of making use of both the statistical and structural information present in handwritten images. This is why HMMs have been used in several handwritten character recognition tasks in recent years [8]. In such HMMs, the states are usually defined as predetermined entities. However, in the present HMM a data-driven or adaptive approach is taken to define the states. The proposed method is robust in the sense that it is independent of several aspects of input such as thickness, size etc.

The next section describes the Devanagari word database followed by Preprocessing and feature extraction in Section 3. Section 4 proposes a HMM classifier. Experimental results are illustrated in Section 5 with conclusions drawn in the last section.

0	आसनसोल	10	हुगली	20	इटावा	30	फरिदाबाद	40	लुधियाना
1	औरंगाबाद	11	मैसूर	21	राणाघाट	31	डेहरीओनसोन	41	पोरबंदर
2	कांकीनाड़ा	12	छपरा	22	साहिबगंज	32	गिरिडीह	42	तिसतातोरसा
3	कपूरथला	13	मेरठ	23	अंडमान	33	एटा	43	विराटि
4	खजूराहो	14	ऊटी	24	भरतपुर	34	उलवेड़िया	44	काकूरगाछी
5	ऋषिकेश	15	झरिया	25	हावड़ा	35	डानकुनी	45	देवघर
6	नैनीताल	16	अहमदाबाद	26	जोधपुर	36	सेवड़ाफुलि	46	चित्रकूट
7	चौरंगी	17	महेषतला	27	पानागढ़	37	थाने	47	कोथीन
8	त्रिवेणी	18	एलौरा	28	विजयवाड़ा	38	वैशाली	48	चंदौसी
9	वाराणसी	19	लक्षमनपुर	29	क्षत्रपतीनगर	39	देहरादून	49	तंजौर

(a)



(b)

Fig. 1. (a) Class number and the Devanagari words forming the 50 classes in our database, (b) Several handwritten samples of the same town name "Tribeni", printed form shown in (a) for class number 8, having lots of variation in writing style

2 Handwritten Devanagari Word Database

Handwritten English benchmark word database exist for the research community and CEDAR word database [9] is one of them. But, there does not exist any benchmark word database for any Indic script. Here, we have attempted to create such a database for handwritten Devanagari words. For data collection, we have designed a special kind of form to collect the data. The form contains 50 boxes within which a writer is to write.

The writers were from different classes of society. They were school/college students, business men, housewives, professionals etc. Each writer was asked to fill a form where the word corresponding to each town name is written once.

No restrictions were imposed on the writing style and no handwritten models were provided in order to obtain a heterogeneous database. These handwritten documents were then scanned at 300 dots-per-inch resolution, in 256 levels of gray. For our experiments, we have considered 50 different names of Indian towns, i.e., the number of word classes is 50. Then the whole database of handwritten words is randomly split into two data sets: a training set with 7000 word images, i.e. 140 word images per class and a test set with 3000 words, i.e. 60 words per class.

A few samples from the same town name of this database written by different classes of people are shown in Fig. 1(b), illustrating variation in handwriting style.

3 Pre-processing and Feature Extraction

Variation in handwriting style makes the handwriting recognition problem quite difficult. So to minimize the effect of writing variability related to different writing styles, we take some preprocessing steps. Feature extraction part exploits the global approach for extracting features without explicitly going for word segmentation.

3.1 Preprocessing

Generally for handwriting recognition, the preprocessing stage includes image smoothing, skew and slant correction, image height and pen stroke width correction. For smoothing, the input gray level image is first median filtered and then binarized by Otsu's [10] thresholding method. The binarized image is then smoothed using median filtering. No skew and slant correction is done here. However, our feature extraction method is insensitive to skew/slant within +/-5 degrees. No image height and pen stroke width correction is done since the extracted features are invariant under image height and the extracted strokes are always one-pixel thick irrespective of the stroke width.

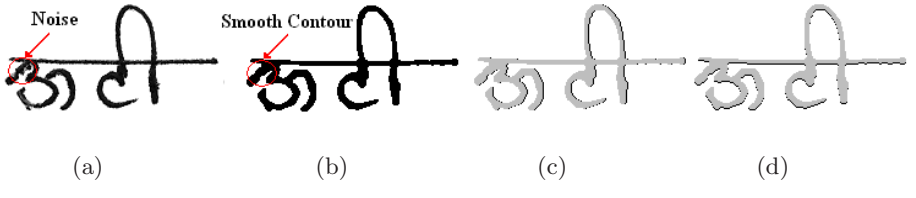


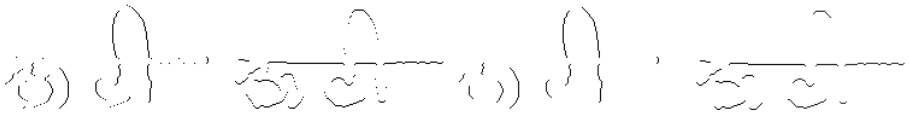
Fig. 2. (a) An input word image for the word Ooty, (b) Image obtained after thresholding and smoothing, Dark and gray pixels indicate (c) E and A images respectively, (d) S and A images respectively

A sample image from the present database and the same after thresholding and smoothing are shown respectively in Figs. 2(a) and 2(b).

3.2 Extraction of Strokes

Let A be the binarized image. We now describe the process of extraction of vertical and horizontal strokes that are present in A. Let E be a binary image consisting of object pixels in A whose right or east neighbour is in the background. That is, the object pixels of A that are visible from the east Fig. 2(c) form E. Similarly, S is defined as the binary image consisting of object pixels in A whose bottom or south neighbour is in the background Fig. 2(d).

The connected components in E represent strokes that are vertical while the connected components in S represent strokes that are horizontal. Each horizontal or vertical stroke is a digital curve. Shapes of these strokes are analyzed for extraction of features. Very short curves are not considered.



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

Fig. 3. (a) E image consisting of vertical strokes obtained from the image in Fig. 2(b), (b) S image consisting of horizontal strokes obtained from the image in Fig. 2(b), (c) Final E image after removal of smaller vertical strokes from the image in (a), (d) Final S image after removal of smaller horizontal strokes from the image in (b)

3.3 Extraction of Features

One of the major factors for the success of any handwritten recognition module is its feature extraction part. The feature should be selected in such a way that it should reduce the intra-class variability and increase the inter-class discriminability in the feature space. From each stroke in E and S, 8 scalar features are extracted. These features indicate the shape, size and position of a digital curve with respect to the word image. A curve C in E is traced from bottom upward. Suppose the bottom most and the top most pixel positions in C are P_0 and P_5. The four points P_1, ... , P_4 on C are found such that the curve distances between P_{i-1} and P_i *(i=1,..., 5)* are equal [11]. Let α_i, *i =1,..., 5* be the angles that the lines $\overrightarrow{P_{i-1}P_i}$ make with the x-axis. Since the stroke here is vertical, $45^0 \leq \alpha_i \leq 135^0$. α_i's are features that are invariant under scaling and represent only the shape. The position features of C are given by $\overline{X}$,$\overline{Y}$ which are the x and y-coordinates of the centre of gravity of the pixel positions in C. $\overline{X}$ is also useful in arranging the strokes present in an image from left to right. Let L be the length of the stroke C. The 3 features $\overline{X}$, $\overline{Y}$ and L are normalized with respect to the image height. Thus, the feature vector becomes $(\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \overline{X}, \overline{Y}, L)$.

The features extracted from a horizontal stroke C in S are similar. Here C is traced from west to east. The feature vector of a horizontal stroke C is defined

as $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5, \overline{X}, \overline{Y}, L)$ where $-45^0 \leq \beta_i \leq 45^0$, $\overline{X}$, $\overline{Y}$ and L are defined in the same way as before [11].

4 Proposed HMM Classifier

An HMM with the state space $S = s_1, \ldots, s_N$ and observation sequence $Q = q_1, \ldots, q_T$ is defined as $\gamma = (\pi, A, B)$ where the initial state distribution is given by $\pi = \{\pi_i\}$, $\pi_i = \text{Prob}\ (q_1 = s_i)$, the state transition probability distribution by $A = \{a_{ij}(t)\}$ where $a_{ij}(t) = \text{Prob}\ (q_{t+1} = s_j / q_t = s_i)$ and the observation symbol probability distributions by $B = \{b_i\}$ where $b_i(O_t)$ is the distribution for state i and O_t is the observation at instant t. The HMM here is non-homogeneous.

Here the problem is how to efficiently compute $P(O/\gamma)$, the probability of the observation sequence, given an observation sequence $O = O_1, \ldots, O_T$ and a model $\gamma = (\pi, \text{A}, \text{B})$. For a classifier of m classes of patterns, we denote m different HMMs by γ_j, $j = 1, ..., m$. Let an input pattern X of an unknown class have a sequence O. The probability $P(O/\gamma_j)$ is computed for each model γ_j and X is assigned to class c whose model shows the highest probability. That is,

$$c = arg \max_{1 \leq j \leq m} P(O/\gamma_j) \tag{1}$$

For a given γ, $P(O/\gamma)$ is computed using the well known forward and backward algorithms [12] . Note that the observation sequence $O = O_1, \ldots, O_T$ in our problem is the sequence of feature vectors of the strokes (arranged from left to right) that are present in a handwritten word image. T is the number of strokes in the image. The states here are certain feature primitives (or more specifically, individual 8-dimensional Gaussian distributions in the feature space) that are found below using EM algorithm.

4.1 HMM Parameters

A feature vector $\theta = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \overline{X}, \overline{Y}, L)$ can come either from a vertical or a horizontal stroke. It is assumed that the features follow a multivariate Gaussian mixture distribution. In other words, $\theta = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \overline{X}, \overline{Y}, L)$ has a distribution $f(\theta)$ which is a mixture of K 8-dimensional Gaussian distributions, namely,

$$f(\theta) = \sum_{k=1}^{K} P_k f_k(\theta) \tag{2}$$

where

$$f_k(\theta) = exp\{-0.5(\theta - \mu_k)^T \Sigma_k^{-1} (\theta - \mu_k)\} / \{(2\pi)^{8/2} |\Sigma_k|^{1/2}\} \tag{3}$$

and P_k is the prior probability of the k-th component. The unknown parameters of the mixture distribution, namely, P_k, μ_k, Σ_k, $(k = 1, ..., K)$ are estimated using the EM (Expectation Maximization) algorithm ([13],[14]) that maximizes the log

likelihood of the training vectors $\{\theta_i, i = 1, ..., n\}$ coming from the distribution given by $f(\theta)$.

The state space of the proposed HMM consists of states which are characterized by the probability density functions $f_k(\theta)$ $(k = 1, ..., K)$. It is assumed that the vertical and horizontal strokes in the word image database are distributed around K different prototype strokes. These are called stroke primitives corresponding to the mean shape vectors $\mu_1, \mu_2, ..., \mu_k$. These K stroke primitives constitute the state space. Thus, the states here are not defined a priori but are determined adaptively on the basis of the training set of word images.

To determine the optimum values of K, we use the Bayesian information criterion (BIC) which is defined as $BIC(K) = -2LL + mlog(n)$, for a Gaussian mixture model with K components, LL is the log likelihood value, m is the number of independent parameters to be estimated, n is the number of observations. For several K values, the $BIC(K)$ values are computed. The first local minimum indicates the optimum K value.

4.2 Estimation of HMM Parameters

In our implementation, $N = K$ and observation symbol probability distribution $b_i(O_t)$ is, in fact, the Gaussian distribution $f_i(\theta) = N(\mu_i, \Sigma_i)$. Thus

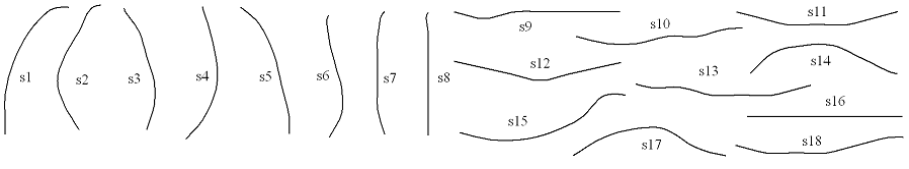
$$b_i(O_t) = exp\{-0.5(O_t - \mu_i)^T \Sigma_i^{-1}(O_t - \mu_i)\}/\{(2\pi)^{8/2}|\Sigma_i|^{1/2}\} \tag{4}$$

The parameters produced by EM algorithm are $P_1, P_2, ..., P_N,\ \mu_1, \mu_2, ..., \mu_N,$ $\Sigma_1, \Sigma_2, ..., \Sigma_N$. Let, in a word image, the strokes be arranged from left to right on the basis of $\overline{X}$ to generate the observation sequence $O_1, O_2, ... , O_T$. For each O_t , compute

$$h_i(O_t) = p_i b_i(O_t)/\{\sum_{j=1}^{N} p_j b_j(O_t)\} \tag{5}$$

and O_t is assigned to state k where

$$k = arg \max_{1 \le i \le N} h_i(O_t). \tag{6}$$



(a) (b)

Fig. 4. Stroke primitives for (a) vertical and (b) horizontal strokes for Devanagari word Ooty

This assignment to respective states is done for all L observation sequences (L is the number of training images). From these L state sequences, the estimates of the initial probabilities are computed as $(1 \leq \ i \leq \ N)$ π_i = (*number of occurrences of* $\{q_1 \in s_i\}$)/(*total number of occurrences of q1*).

The transition probability estimates $a_{i,j}(t)$ are computed as $(1 \leq \ i \leq \ N, 1 \leq j \leq \ N, 1 \leq \ t \leq \ T-1)$ (*number of occurrences of* $\{q_t \in s_i \& q_{t+1} \in s_j\}$) / (*total number of occurrences of* $\{q_t \in s_i\}$). The above HMM parameter estimates are fine-tuned using re-estimation by Baum-Welch forward-backward algorithm.

5 Experimental Results

The proposed scheme has been tested on the recently developed database of handwritten Devanagari word images. The results of our study are reported below. To the best of our knowledge, there does not exist any other standard database of handwritten Devanagari word images. The training and test datasets here consist of 7000 and 3000 handwritten word images respectively. From these word images, 156906 horizontal and 137250 vertical strokes have been extracted from the training set whereas 67245 horizontal and 58821 vertical strokes have been extracted from the test set. The parameters of an HMM for each of the 50 word classes are determined using the method described in Section 4.

For example, for word class **Ooty**, the K value is found to be 18. The curves corresponding to the 18 mean vectors μ_k are shown in Fig. 4. These represent the 18 HMM states for **Ooty**. For the image shown in Figs. 3(c) & 3(d), the strokes are shown in Fig. 5. The strokes arranged in terms of $\overline{X}$ from left to right are e1, e2, e3, ..., e26. The most likely states of these 26 strokes individually are s15, s4, s13, s17, s7, s15, s17, s6, s4, s16, s17, s15, s3, s15, s2, s14, s15, s4, s1, s14, s16, s3, s18, s6, s16 and s3 respectively. The probability $P(O/\gamma_j)$ is computed

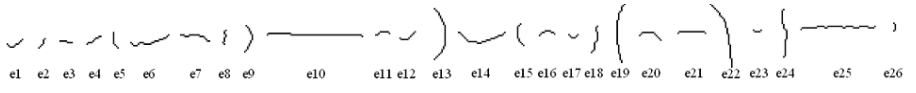


Fig. 5. e1 to e26 represent the strokes arranged from left to right along X-axis

for $j = 1, ..., 50$ and the image is classified as class c where

$$c = arg \max_{1 \leq \ j \leq \ 50} P(O/\gamma_j) \tag{7}$$

We have achieved 82.89% correct recognition rate on the test set and 87.71% on the training set.

6 Conclusion

In this paper we have proposed a HMM based approach to recognition of handwritten Devanagari words. The results of our approach are promising for a small Lexicon size. It indicates that it is possible to scale our system to large vocabularies. Our system is based on Global approach, which extracts global features thus reducing the overhead of segmentation. Our future work will be to combine both these local and global approaches resulting in a hybrid approach for more efficiency.

References

1. Rahman, A.F.R., Rahman, R., Fairhurst, M.C.: Recognition of handwritten Bengali characters: a novel multistage approach. Pattern Recognition 35, 997–1006 (2002)
2. Bhattacharya, U., Das, T.K., Datta, A., Parui, S.K., Chaudhuri, B.B.: A hybrid scheme for handprinted numeral recognition based on a self-organizing network and MLP classifiers. Int. J. Patt. Recog. & Art. Intell. 16(7), 845–864 (2002)
3. Bhattacharya, U., Chaudhuri, B.B.: A majority voting scheme for multiresolution recognition of handprinted numerals. In: Proc. of the 7th ICDAR, Edinburgh, Scotland, vol. I, pp. 16–20 (2003)
4. Ramakrishnan, K.R., Srinivasan, S.H., Bhagavathy, S.: The independent components of characters are 'Strokes'. In: Proc. of the 5th ICDAR, pp. 414–417 (1999)
5. Lehal, G.S., Bhatt, N.: A recognition system for Devnagri and English handwritten numerals. Advances in Multimodal Interfaces. In: Tan, T., Shi, Y., Gao, W. (eds.) ICMI 2001. LNCS, vol. 1948, pp. 442–449. Springer, Heidelberg (2000)
6. Kim, G.: Recognition of offline handwritten words and its extension to phrase recognition, PhD Thesis. University of New York at Buffalo, USA (1996)
7. Guillevic, D.: Unconstrained Handwriting Recognition Applied to the Processing of Bank Cheques, Thesis of Doctor's Degree in the Department of Computer Science at Concordia University, Canada (1995)
8. Park, H., Lee, S.: Off-line recognition of large-set handwritten characters with multiple hidden Markov models. Pattern Recognition 29, 231–244 (1996)
9. Hull, J.J.: A database for handwritten text recognition research. IEEE Trans. Patt. Anal. Mach.Intel. 16, 550–554 (1994)
10. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Trans. on Systems, Man, and Cybernetics 9, 62–66 (1979)
11. Bhattacharya, U., Parui, S.K., Shaw, B., Bhattacharya, K.: Neural Combination of ANN and HMM for Handwritten Devnagari Numeral Recognition. 10th-IWFHR, pp. 613–618 (2006)
12. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE 77(2), 257–286 (1989)
13. Fukunaga, K.: Introduction to Statistical Pattern Recognition, 2nd edn. Academic Press, San Diego (1980)
14. Parui, S. K., Bhattacharya, U., Shaw, B., Poddar, D.: A Novel Hidden Markov Models for Recognition of Bangla Characters. 3rd-WCVGIP, pp. 174–179 (2006)