Multi-lingual City Name Recognition for Indian Postal Automation

Umapada Pal
Computer Vision and Pattern Recognition Unit; Indian Statistical Institute, Kolkata-108, India
Email: umapada@isical.ac.in

Ramit Kumar Roy
St. Xavier's College
30 Park Street
Kolkata-16, India
Email: ramitkumar.roy@gmail.com

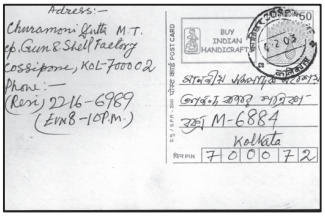
Fumitaka Kimura
Graduate School of Engg., Mie University; 1577 Kurimamachiya-cho, TSU, 514-8507, Japan
Email: kimura@hi.info.mie-u.ac.jp

***Abstract*—Under three-language formula, the destination address block of postal document of an Indian state is generally written in three languages: English, Hindi and the State official language. From the statistical analysis we found that 12.37%, 76.32% and 10.21% postal documents are written in Bangla, English and Devanagari script, respectively. Because of inter-mixing of these scripts in postal address writings, it is very difficult to identify the script by which a city name is written. To avoid such script identification difficulties, in this paper we proposed a lexicon-driven method for multi-lingual (English, Hindi and Bangla) city name recognition for Indian postal automation. In the proposed scheme, at first, to take care of slanted handwriting of different individuals a slant correction technique is performed. Next, a water reservoir concept is applied to pre-segment the slant corrected city names into possible primitive components (characters or its parts). Pre-segmented components of a city name are then merged into possible characters to get the best city name using the lexicon information. In order to merge these primitive components into characters and to find optimum character segmentation, dynamic programming (DP) is applied using total likelihood of the characters of a city name as an objective function. We tested our system on 16132 Indian trilingual city names and 92.25% overall recognition accuracy was obtained.**

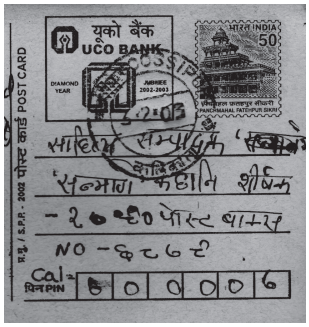
Keywords- Handwritten character recognition, Indian Script, city name recognition, Indian postal automation.

I. INTRODUCTION

Postal automation is a topic of research interest for more than two decades and many pieces of published article are available towards postal automation of non-Indian language documents [1,2, 11-14]. At present postal sorting machines are available in several countries like USA, UK, Canada, Japan, France, Germany etc. There are only a few works on Indian postal system [3,4,8] and at present no postal automation machine is available for India. System development towards Indian postal automation is more difficult and challenging than that of other country because of its multi-lingual and multi-script behavior. In India there are 22 official languages and 11 scripts are used to write these languages. Although there are many languages in India, the destination address block of a postal document in an Indian state is generally written in three languages: English, Hindi and the State official language. Thus Indian postal documents are tri-lingual in nature. To take care of such tri-lingual documents, in this paper we proposed a tri-lingual (English, Hindi and Bangla) city name recognition. Bangla is the official language of West Bengal State of India and Hindi is the Indian national language. We computed different statistics from a database of 7500 postal documents collected from West Bengal state of India. For detail statistics see [4]. From the statistical analysis we found that 12.37%, 76.32% and 10.21% postal documents are written in Bangla, English and Devanagari script, respectively (Hindi is written in Devanagari script). Thus, development of multi-script postal documents is very useful. Earlier we proposed a bilingual city name recognition [8] and this work is the extension for city name recognition of Indian Postal documents written in English, Hindi and Bangla.



(a)



(b)

Figure1. Examples of Indian postalDocuments written in Bangla/Hindi and English.

There are two approaches for the OCR of multi-script nature: (1) identify the script and then use appropriate OCR based on the script (2) develop a system capable to recognize samples of all the scripts. There are many pieces of published work on script identification. The script identification from the address portion of a postal document is very complicated due to inter-mixing of scripts while writing postal address. It is found that some people write

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.238

the destination address part of a postal document in two or more scripts. See Fig.1, where the destination address is written partly in Bangla, Hindi and English. Also, a single line of an Indian postal document may contain two or more scripts. So identification of script by which city names are written is very difficult task because of the inter-mixing of scripts. A lexicon-driven segmentation-recognition scheme is proposed here for tri-lingual city name recognition and it is as follows. At first, binarization of a city name is done and the slant correction of the individual city name is performed. Next city names are segmented into primitives (individual characters or its parts) using water reservoir principle [9]. Each primitive ideally consists of a single character or a sub-image of a single character. In order to merge these primitive components into characters and to find optimum character segmentation of a city name, dynamic programming (DP) is applied using the total likelihood of characters as the objective function. To compute the likelihood of a character, a MQDF based on the directional features of the contour points of the components is used. Block diagram of the proposed system is shown in Fig.2.

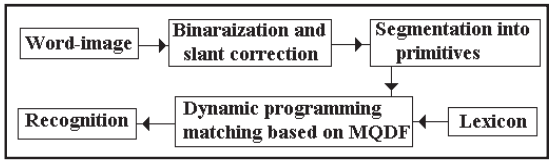


Figure 2. Block diagram of the proposed system.

II. SLANT CORRECTION AND CHARACTER PRE-SEGMENTATION

A. Slant correction

Slant correction is an important preprocessing step for handwriting recognition. In this paper the gray scale image of an input city name is binarized by Otsu algorithm [5] and the binary image is then slant estimated and corrected using the method proposed by Kimura et al. [7]. The slant estimation is done based on the chain code information of the contour of the city name image.

B. Character pre-segmentation

To find optimal character segmentation by the segmentation-recognition scheme using dynamic programming, we pre-segmented the city names into primitives. When two or more characters sit side by side to form a word the characters touch and generate big cavity region. We find this cavity region based on the profile information and water reservoir principle and the deepest point of each cavity region is considered for pre-segmentation. Please note that not all the cavities are considered for pre-segmentation. We compute the median of the heights of different cavities obtained in a city name and those cavities having height greater than 0.8*median are considered for segmentation. Because of the structural shape of some characters, some small cavities may be obtained where segmentation should not be done and we use this threshold to ignore the segmentations of such small cavities. In character pre-segmentation our aim was to segment a city name into individual characters as much possible avoiding much over segmentation. The lower cavity (reservoir) portions of a Bangla city name shown in Fig. 3(a) are marked by grey in Fig.3(b). After detection of pre-segmentation columns from the cavities of an input city name image, the image is split vertically at each pre-segmentation column and is separated into non-overlapping zones. A connected component analysis is applied to the split image to detect the boxes enclosing each connected component. These boxes are usually disjoint and do not include parts of other connected components. Connected components in the split city name image and their enclosing boxes are shown in Fig.3(c). These boxes are numbered (from left to right) and these numbers are shown at the upper side of Fig.3(c). These connected components are regarded as primitive segments and each of which corresponds to a full character or a part of a character.

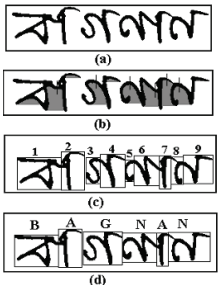


Figure 3. Examples of city namepre-segmentation.

III. FEATURE EXTRACTION

Histograms of direction chain code of the contour points of the components are used as features for recognition [6]. For recognition we use 64 dimensional features and the feature extraction procedure is described below.

At first the bounding box is divided into 7 × 7 blocks (as shown in Fig.4(c). In each of these blocks the direction chain code for each contour point is noted and frequency of direction codes is computed. Here we use chain code of four directions only [horizontal, 45 degree slanted, vertical and 135 degree slanted]. Thus, in each block, we get an array of four integer values representing the frequencies of chain code in these four directions. These frequencies are used as feature. Histogram of the values of these four direction codes in each block of a Bangla character is shown in Fig.4(d). Thus, for 7 × 7 blocks we get 7 × 7 × 4 =196 features. To reduce the feature dimension, after the histogram calculation in 7 × 7 blocks, the blocks are down sampled into 4 × 4 blocks using a Gaussian filter. As a result we have 64 (4 × 4 × 4) dimensional features for recognition. Histogram of these values of all the four directions obtained after down sampling is shown in

Fig.4(e). The feature vector is normalized by dividing each component by the character height to make it size independent.

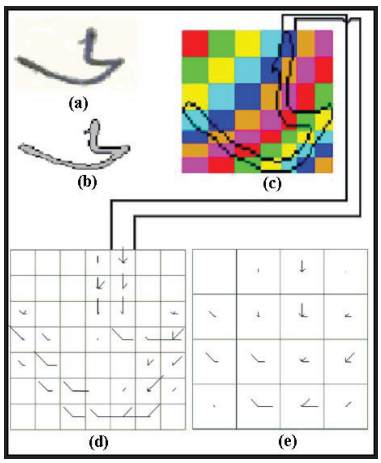


Figure 4. Example of feature extraction (a) Bangla character. (b) its contour (c) 7 x 7 segmented blocks shown in the zoomed version of 4(b). (d) Block-wise chain code histogram of contour points. (e) Chain code histogram after down sampling into 4 x 4 blocks from 7 x 7 blocks.

One critical point in segmentation-recognition techniques using dynamic programming is the speed of feature extraction, because the correct segmentation points have to be determined in optimization process with respect to the total likelihood of the resultant characters. The use of the cumulative orientation histogram enables one to realize high-speed feature extraction. Border following for feature extraction and orientation labeling are performed only once to an input city name image, and the orientation feature vector of a rectangular region including one or more boxes is extracted by a small number of arithmetic operations for high-speed feature extraction [7].

IV. SEGMENTATION RECOGNITION USING DYNAMIC PROGRAMMING

The core of a dynamic programming algorithm is the module that takes a city name image, a string to incorporate contextual information, and a list of the primitives from the city name image and returns a value that indicates the confidence that the city name image represents the string. Given a lexicon city name, the primitive segments of the city name image are merged and matched against the characters in the lexicon city name so that the average character likelihood is maximized using dynamic programming. If an input city name has Z primitive segments and when such city name is matched against a lexicon word of P characters then a character can contain at most (Z-P+1) primitive segments.

A. Markov chain representation

The number of the boxes (primitives) of an input city name image is usually 1.2 to 2 times as many as the number of characters in the city name image. In order to merge these primitive components into characters and find the optimum character segmentation, dynamic programming (DP) is applied using the total likelihood of characters as the objective function [7]. The ASCII lexicon of possible city names of Bangla, Hindi and English is utilized in the process of dynamic programming to incorporate contextual information. The likelihood of a component segment is calculated using the modified quadratic discriminant function. To apply the DP, the boxes are sorted from left to right according to the location of their centroids. If two or more boxes have the same x coordinates of their centroids, they are sorted from top to bottom. Numbers at the top of the boxes in Fig.3(c) show the order of the sorted boxes. It is worth observing that the disjoint box segmentation and the box sorting process reduce the segmentation problem to a simple Markov process, in most cases. For example, the box 1 corresponds to character “B” of the Bangla city name *BAGNAN,* box 2 corresponds to character “A”, boxes 3 and 4 correspond to character “G”, boxes 5 and 6 correspond to character “N”, box 7 corresponds to character “A”, and boxes 8 and 9 correspond to character “N”. See Fig.3(d) where characters’ codes of this city name image are given. These assignments of boxes to character are represented, for example, by B A G - N - A N -
where "-" is used to denote the continuation. The assignment is also represented, for example, by

	B	A	G	N	A	N
i -->	1	2	3	4	5	6
$j(i)$ -->	1	2	4	6	7	9

where i denotes the letter number, $j(i)$ denotes the number of the last box corresponding to the i-th letter. Note that the number of the first box corresponding to the i-th letter is $j(i-1)+1$. Given $[j(i), i=1,2,..,n]$ the total likelihood of characters is represented by

$$L = \sum_{i=1}^{n} l(i, j(i-1)+1, j(i)) \qquad \text{----(1)}$$

where $l(i, j(i-1)+1, j(i))$ is the likelihood for i-th letter. The optimal assignment (the optimal segmentation) that maximizes the total likelihood is found in terms of the dynamic programming as follows. The optimal assignment $j(n)^*$ for n-th letter is the one such that:

$$L^* = L(n,j(n)^*) = \text{Max } L(n,j(n)) \qquad \text{----(2)}$$

where $L(k,j(k))$ is the maximum likelihood of partial solutions given $j(k)$ for the k-th letter, which is defined and calculated recursively by

$$L(k,j(k)) = \underset{j(1),j(2).....j(k-1)}{Max} \{\sum_{i=1}^{k} l(i, j(i-1)+1, j(i))\}$$

$$= \underset{j(k-1)}{Max}[l(k, j(k-1)+1, j(k)) + L(k-1, j(k-1))] \quad \text{----(3)}$$

and $L(0,j(0)) = 0$ for $j(0) = 1,2,\ldots m$ ----(4)

Starting from (4), all $L(k, j(k))$'s are calculated for $k = 1,2,\cdots\cdots,n$ using (3) to find $j(n)^*$ using (2). The rest of $j(k)^*$'s (k=n-1,n-2,...,1) are found by back tracking a pointer array representing the optimal $j(k\text{-}1)^*$'s which maximizes $L(k,j(k))$ in (3).

B. MQDF for Character likelihood

Modified Quadratic Discriminant Function (MQDF) is a powerful, easy to train classifier and because of this character likelihood is calculated by the following modified quadratic discriminant function [6].

$$g(X)=\{|X-\hat{M}|^2 - \sum_{i=1}^{k} \frac{\lambda_i}{\lambda_i + h^2} [\phi_i^T (X-\hat{M})]^2\}/h^2$$
$$+ \ln[h^{2(n-k)} \prod_{i=1}^{k} (\lambda_i + h^2)] \quad \text{----(5)}$$

where X denotes the input feature vector, $\hat{M}$ denotes the sample mean vector for each character class, and λ_i and ϕ_i denote the eigenvalues and eigenvectors of the sample covariance matrix. Values of constants h^2 and k are selected experimentally to achieve the best performance. In the following experiments, k is set to 20 and h^2 to $3/8*\sigma^2$, where σ^2 is the mean of eigenvalues $\lambda_{i's}$ over i and character classes.

Given a feature vector, $g(X)$ is calculated for a character class specified by a city name lexicon.

V. RESULTS AND DISCUSSION

A. Data details

For the experiment of the city-name recognition scheme, we collected a total of 16132 handwritten tri-lingual city-name samples of which 4257 samples are from Hindi, 8625 in Bangla and 3250 in English. There were 290 city-name classes in total (117 Hindi, 84 Bangla and 89 English). Minimum number of samples in a class was 20. Also average lengths of Hindi, Bangla and English city names were 6.86, 5.85 and 8.46 characters. These city name samples are collected from handwritten address block of Indian postal documents as well as from some individuals using some specially designed forms. We have used 5-fold cross validation scheme for recognition result computation. Here database is divided into 5 subsets and testing is done on each subset using other four subsets for learning. The recognition rates for all the test subsets are averaged to calculate recognition accuracy. The learning is conducted to estimate the parameters of MQDF.

B. Measures of result computation

For recognition result computation we used different measures and they are defined as follows: Recognition rate = (N_C*100) / N_T, Error rate = (N_E*100) / N_T, Rejection rate = (N_R*100) / N_T, Reliability = (N_C*100) / (N_E+N_C), Where N_C is the number of correctly classified city names, N_E is the number of misclassified city names, N_R is the number of rejected city names and N_T is the total number of city names tested by the classifier. Here N_T= (N_C +N_E +N_R).

C. Global tri-lingual city name recognition results

From the experiment we noted that the overall city name recognition accuracy of the proposed multi-script scheme was 92.25%, when no rejection was considered. Also, from the experiment we noted that overall 95.55% (96.36%) accuracy was obtained when first three (five) top choices of the recognition results were considered. Detail results of 3 scripts of our multi-script city name recognition system with different top choices are shown in Table 1. From the table it can be noted that Hindi city name samples show lower results than Bangla and English city name samples in our tri-lingual system.

TABLE 1: TRI-LINGUAL CITY NAME RECOGNITION RESULTS BASED ON DIFFERENT TOP CHOICES OF BANGLA, HINDI AND ENGLISH CITY NAMES (WHEN REJECTION IS 0.0%)

Number of top choices	Recognition rate (%)			
	Bangla	Hindi	English	Overall
Top 1 choice	93.68	90.09	91.26	92.25
Top 2 choices	96.38	92.34	93.35	94.71
Top 3 choices	97.25	93.56	93.63	95.55
Top 4 choices	97.80	94.03	93.91	96.02
Top 5 choices	98.16	94.43	94.09	96.36

D. Mono-lingual city name recognition results

To get idea of the city name recognition results of mono-lingual city name, we also computed their recognition results. We obtained 94.08%, 90.16% and 91.63% accuracy when city names samples of individual scripts are considered, separately. Mono-lingual city name recognition results based on different top choices of Bangla, Hindi and English city names are shown in Table 2. From the tables 1 and 2 it can be noted that tri-lingual system works as good as mono-lingual system.

E. Rejection versus Reliability results

From the system we also computed rejection vs. reliability of our tri-lingual city name recognition system. We noted that system provides 99.55% reliability when error and rejection rates are 0.20% and 28.11%, respectively. City name recognition reliability with different rejection rates is

given in Table 3. Rejection is done based on: (i) optimal likelihood value of the best recognized city name, and (ii) difference of the optimal likelihood values of the best and the second-best recognized city names.

TABLE 2. MONO-LINGUAL CITY NAME RECOGNITION RESULTS BASED ON DIFFERENT TOP CHOICES OF BANGLA, HINDI AND ENGLISH CITY NAMES (WHEN REJECTION IS 0.0%)

Number of top choices	Recognition rate (%)		
	Bangla	Hindi	English
Top 1 choice	94.08	90.16	91.63
Top 2 choices	96.74	92.44	93.54
Top 3 choices	97.55	93.66	93.94
Top 4 choices	98.12	94.08	94.09
Top 5 choices	98.38	94.57	94.28

TABLE 3.ERROR AND RELIABILITY RESULTS OF THE PROPOSED SYSTEM WITH RESPECT TO DIFFERENT REJECTION RATES.

Reliability (%)	Error rate (%)	Rejection rate (%)
95.04	4.62	4.44
98.00	1.65	11.69
99.04	0.67	18.01
99.55	0.20	28.11

F. Comparison of results

To the best of our knowledge this is the first work on tri-lingual city name recognition where we considered Bangla, Hindi and English city name strings in our system. Since there is no tri-lingual city name recognition result in the literature, we cannot compare our results. To the best of our knowledge there is only work on Hindi word recognition [10] and HMM has been used in the work. Their scheme was tested only on a small class of Hindi word data. They consider only 50 word classes of Hindi and obtained 82.89% accuracy from a test set of 3000 words. With compare to this we obtained 90.16% results from Hindi city name recognition when number of class was 117. We hope mono-lingual as well as tri-lingual city name recognition results reported in this paper will be helpful to the researchers for future wok.

G. Error analysis

From the experiment we noted that some errors occurred when the set of character pre-segmentation points obtained from our pre-segmentation module did not contain actual character segmentation points. Another reason of miss-recognition was the shape similarity of some of the city names. As mentioned earlier, we obtained lowest results in Hindi city names. From the experiment we noticed that one of the main reasons for not getting higher recognition rates in Hindi was the complex structure of some of the city names due to presence of compound characters in Hindi. Since there are more than 200 compound characters in Hindi, it was difficult to get the real samples of this large number of compound characters to train our systems to compute their likelihood. In future we plan to collect more samples of compound characters to get higher results.

CONCLUSION

In this paper we proposed a system for Indian tri-lingual handwritten city name recognition and the city name recognition problem is treated as lexicon based word recognition. There is no work in the literature dealing with tri-lingual city name recognition for Indian scripts and this is the first report of tri-lingual city name recognition. We obtained 99.55% reliability from our proposed system when error and rejection rates are 0.20% and 28.11%, respectively.

REFERENCES

[1] L. Liu and M. Koga and H. Fujisawa, "Lexicon driven segmentation and recognition of handwritten character strings for Japanese address reading", IEEE Trans on PAMI, vol. 24, pp. 1425-1437, 2002.

[2] R. Plamondon and S. N. Srihari, "On-Line and off-line handwritten recognition: A comprehensive survey", IEEE Trans on PAMI, Vol.22, pp.62-84, 2000.

[3] U. Pal, K. Roy and F. Kimura, "A Lexicon Driven Handwritten City Name Recognition Scheme for Indian Postal Automation", IEICI Trans. on Information and Systems, Vol.E92-D, No.5, pp. 1146-1158, 2009.

[4] S. Vajda, K. Roy, U. Pal, B. B. Chaudhuri and A. Belaid, "Automation of Indian Postal Documents written in Bangla and English", Int. Journal of PRAI, Vol. 23, No. 8, pp. 1599–1632, 2009.

[5] N. Otsu, "A threshold selection method from gray-level histograms", IEEE Trans. SMC, vol.9, pp.62-66, 1979.

[6] F. Kimura, Y. Miyake and M. Sridhar, "Handwritten ZIP code recognition using Lexicon free word recognition algorithm", In Proc. 3rd ICDAR, pp. 906-910, 1995.

[7] F. Kimura, S. Tsuruoka, Y. Miyke and M. Shridhar, "A Lexicon Directed Algorithm for Recognition of Unconstrained Handwritten Words", IEICE Trans. Inf. and System, Vol.E7-D, No.7, pp.785-793, 1994

[8] U. Pal, R. K. Roy and F. Kimura, "Bangla and English City Name Recognition for Indian Postal Automation", In Proc ICPR-2010, pp. 1985-1988, 2010.

[9] U. Pal, A. Belaid and Ch. Choisy,"Touching numeral segmentation using water reservoir concept", Pattern Recognition Letters, vol.24, no. (1-3), pp. 261-272, 2003.

[10] S. K. Parui and B. Shaw, "Offline Handwritten Devanagari Word Recognition: An HMM Based Approach", In Proc. PReMI 2007, pp. 528-535. 2007.

[11] U. Mahadevan, and S. N. Srihari, "Parsing and Recognition of City, State, and ZIP Codes in Handwritten Addresses", In Proc. of 5th ICDAR, pp. 325-328, 1999.

[12] X. Wang, and T. Tsutsumida, "A New Method of Character Line Extraction from Mixed-unformatted Document Image for Japanese Mail Address Recognition", In Proc. of 5th ICDAR, pp. 769-772, 1999.

[13] S. N. Srihari, and E .J. Keubert, "Integration of Hand-Written Address Interpretation Technology into the United States Postal Service Remote Computer Reader System", In Proc. of 4th ICDAR, pp. 892-896. 1997.

[14] A. Kornai, "An Experimental HMM-Based Postal OCR System", In Proc. of ICASSP'97, pp. 3177-3180, 1997.