

Asian Resonance

Special Approach for Segmenting and Recognizing Connected Handwritten MODI Script Characters

Abstract

In this paper, Special approach is proposed for the segmentation and recognition of handwritten character strings of MODI script. In the method, a gradient descent mechanism is used to weigh the distance measure in applying KNN for segmenting/ recognizing connected characters (numerals and MODI characters) in the left-to-right scanning direction. In recognizing connected characters, a high quality segmentation technique is essential. Conventional approaches attempt to separate the string into individual characters without recognition and apply a recognition algorithm onto each isolated character, resulting improper segmentation and poor recognition results in many situations. Our proposed algorithm simulates the human beings' process in recognizing connected character strings where segmentation and recognition is mingled with each other. Experimental results on 1000 character strings from the newly created database of different persons show that the algorithm works robustly and efficiently.

Keywords: MODI script, segmentation, recognition, projection profile, gradient descent KNN.

Introduction

Previous studies on recognition of handwritten numerals of other languages revealed that about 15% of such characters involved connected numerals [1]. For recognition of connected characters, an efficient fragment technique is essential, while the conventional approaches attempted to separate the string into individual characters by splitting the string slantly and apply distinguishing analysis to retrieve the characters by shapes [1]. Some researchers considered ways to separate the characters completely by selecting favorable plausible areas to the segmentation from the background [2]. Some others tried to separate the lexicon string by a continuous density hidden Markov model and used dynamic programming approach to match the character with word images and strings [3]. These approaches might not be able to achieve a satisfactory result. While the uncertainty occurs due to the lack of knowledge on a string with connected characters, the previous algorithms had to make the separation of touched characters by applying an previously assigned confidence value for the segments of words to reflect the ambiguity among character classes, and such separation algorithm could only successfully separate the strings with two-touched or three-touched together characters [4]. In consideration of human-beings ability to deal with the recognition of strings with connected characters, the mixed process of bottom-up (image content based analysis) and top-down (knowledge based recognition) and the combination of segmentation and recognition processes had been proposed by Chi [3], [5] and other scientists for a more efficient algorithm. In their proposed solutions, segmentation boundaries in the forms of vertical or slant lines and in many cases curves [9] or piecewise lines are in utilization for separation of the connected characters. Many researchers have also proposed models of fuzzy logic rules [6], neural networks, linear discriminant functions, template matching, and binary comparisons for evaluation and retrieval of features from the strings with single and double-touched handwritten numerals [7]. Some other good models have been proposed in this area. T.M Ha et al. [14] suggested a system for off-line recognition of handwritten numeral strings, which can recognize a previously divided string fragment of numbers by calculating of weighted sum of confidences over all classes. A very effective algorithm was proposed by Shi and Govindaraju [15], that segmenting connected handwritten digit strings. They have very good results to recognize samples



D. N. Besekar

Associate Professor,
Dept. of Computer Science,
Shri Shivaji College of
Science, Akola, Maharashtra

Asian Resonance

from popular handwritten character databases. The previous researches could not analyze the features completely upon the recognition of characters. In this paper, a novel segmentation-recognition algorithm is proposed where the segmentation mechanism in KNN (K-nearest neighbor) algorithm and gradient descent mechanism is introduced into the procedure of string segmentation for the purpose to enable a computer to recognize the character in a way as similar as a human being does. The recognition procedure is accomplished by K-means clustering algorithm [8]. In this paper, a detailed methodology is described and some experimental results are provided for discussion.

About MODI script:

A script, which is generally neglected to be mentioned in the discussion on the Indian scripts, is MODI, shown in the figure 1. The use of MODI in official Marathi documents and administration was common in Maharashtra till the end of 19th century. The British Government of Bombay Presidency in the beginning of 20th century for the sake of convenience and uniformity with the other areas of the presidency decided that the Devnagari (Balbodh as it is called in Maharashtra) should be used as a primary writing system in administration. Thus the Devnagari became the predominant script although MODI continued to be taught in schools and was used as an alternate script in Marathi writing. The script was widely used even in 1940s by the people of older generation for personal and financial documentation. With the time however the use of MODI diminished and now it has become almost extinct.

Traditionally it is believed that the MODI script was developed by Hemadpant, a well-known administrator in the court of Ramdevrao, the last king of Yadav dynasty (1187-1318) at Devgiri. Hemadpant is also credited with a specific temple architecture called "Hemadpant temples". The general use of MODI in administration is however seems to be introduced by Balaji Avaji Chitnis, a minister in Chhatrapati Shivaji's court. It is said that Balaji while attending Durbar (Mogul Court) at Delhi observed that for the fast transcription of Persian proceedings in Mogul court were written in Shikasta (broken) script as against Nastaliq script, a clear but slower Persian handwriting. Balaji recognized the importance of speed of writing in administrative affairs and thus introduced the MODI script in Maratha administration. The term MODI seems to be literal translation of Persian term Shikasta.

Strandberg in her work on the MODI documents from Tanjore in Danish collection has given interesting information on MODI writing along with complete series of twelve letters of Marathi alphabets i.e. Barakhadi. The work also contains various theories regarding the origin of MODI writing including some fanciful suggestions such as "Paishachi" was written in MODI. We, however, know that the legend of Gunadhyay and Kanubhuti Vetal attached to Brihtkatha belongs to 2nd century. MODI is strictly written below the line unlike any other scripts of 2nd century such as Brahmi or Kharoshti. Moreover

many letters in MODI are the same as in Devnagari. On the basis of known documents it is safe to assume that the MODI was not developed before 12th century.

The MODI script was used to write the Marathi language spoken in the Indian state of Maharashtra. It originated as a cursive variant of the script during the 17th century CE. MODI was used until the 1950's when Devanagari replaced it as the written medium of the Marathi language.

The following is the basic MODI script.



Fig. 1. MODI characters

Methodology:

Several modules are involved in newly proposed segmentation-recognition algorithm, which is shown in Fig.2. The main processes include the followings:

- A) Data preparation processing of the string with connected characters;
 - normalization,
 - noise removal, and
 - Small connected components elimination.
- B) Construction of training sample database by manually segmenting connected character strings and the recognition modules training using the samples;
- C) Segmentation of the string from left most pixel to right most one and recognition of connected numeral involved by comparing with the samples from the trained recognition modules in Step 2.

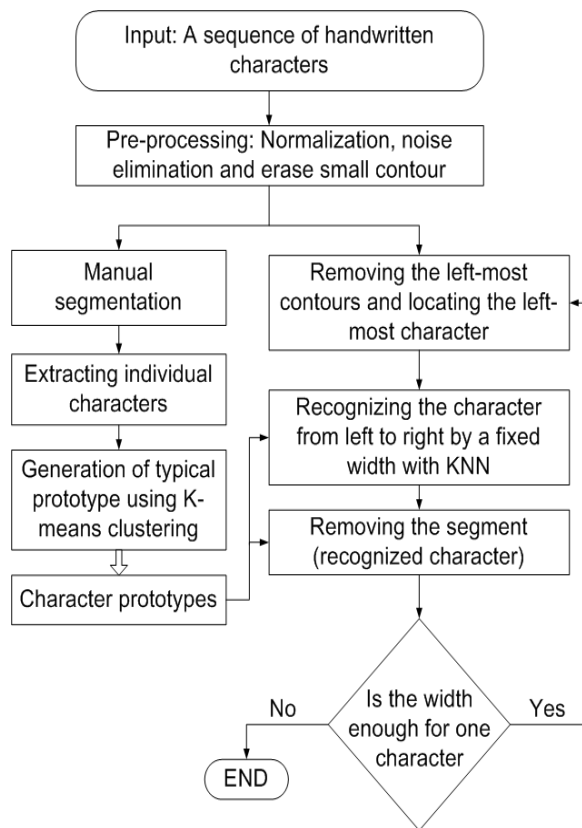


Fig. 2. Flowchart of the algorithm

A. Data Preparation

1) Input character normalization:

As the size of an input string image could be different, the image needs go through the normalization process to become standardized. The length of the string depends on length of the original sentence, but the height of the image should be standardized to be the same for comparison with the sample characters in training database. In the experiments, the height of the input characters in a candidate string image is normalized to 32 pixels.

2) Small contours elimination:

In the original candidate string image or after removal of the recognized character, some noise like spots might be on the image. These small black dots are called small contours, which always affect the localization of a character [13]. The contours between two characters could be connecting area of two characters, while the small contours on the top or low areas of the image are more possibly to be noise, so before segmentation and recognition of a character, the small up- or down-contours should be detected and deleted properly. Two sample images with small contours are shown in upper part of Fig. 3, while 3(a) has an extension stroke over top of the left most character, and 3(b) has a trace after removal of a character in the upper line. The lower part of Figure 3 shows the results of deleting small contours from the original images.

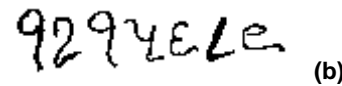
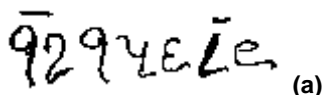


Fig. 3. Example strings before removal of small contours in sample candidate string (a) and after removal (b).

3) Contour detection and deletion:

Several algorithms have been proposed by previous studies about detecting the connected components algorithms. A connected component is defined as a set of pixels that share the following characteristics [8], which therefore to be recognized as area of a small contour:

- Pre-define the height of a normalized character to 32-pixel.
- If there exists one path entirely composed of elements in the set between any two pixels, and the height of all connected pixels is less or equal to 8-pixel, the path is regarded as a small contour.

Small contour deception: the process to be completed in segmentation step:

- To eliminate the noise in the image of candidate string, the connected contours with very small areas (height is less than 8-pixel) or the locations where the numbers of black pixels are to limited are removed.
- To extract character blocks, the average width of connected components and the average aspect ratio (after noise elimination) are estimated. If these two values exceed a certain limit, the adhesion would be identified for subsequent segmentation.

B. Construction of training template database

1) Training template collection:

Training templates are selected from several strings with connected characters, which are manually segmented with three steps:

- eliminating small contours and noise components,
- carefully drawing curve to separate the connected characters manually, and
- Labeling the segmented character while several similar numeral templates are recorded in the database.

Fig. 4 shows an example of manual segmentation of a string with connected characters. The red curves are the boundary for separating the characters from each other, and, obviously, very little straight cutting line can be found.

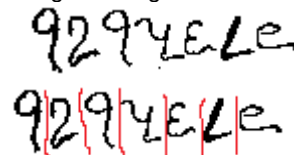


Fig. 4. Manual segmentation of a string with connected characters

2) Extraction of individual characters:

Individual characters are extracted and left aligned for training templates. After converting the gray regions of each segmented character block to the color of background, the clean segmented

character templates as shown in Figure-5 can be obtained and are stored in the database for future string comparisons.

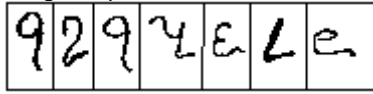


Fig. 5. Individual characters extracted and aligned

3) Clustering:

In statistics and machine learning, k-means clustering is a method of cluster analysis which aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. It is similar to the expectation-maximization algorithm for mixtures of Gaussians in that they both attempt to find the centers of natural clusters in the data. This method is applied to construct a database of samples of possible characters/numerals which should be set as comparison targets in our recognition algorithm.

Given a set of observations (x_1, x_2, \dots, x_n) , where each observation is a d-dimensional real vector, then k-means clustering aims to partition the n observations into k sets $(k < n)$

$S = S_1, S_2, \dots, S_k$ so as to minimize the within-cluster sum of squares (WCSS) as Equation-1.

$$\arg \max_s \sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - \mu_i\|^2 \dots (1)$$

Where μ_i is the mean of points in S_i .

4) Training recognition modules:

An improved KNN algorithm is used to train the recognition modules. Several factors are considered in training. Given the width of the image of candidate string is W, and for a particular pixel A in the image, its ordinates are (x, y) , the pixel next to it should be calculated as Equation-2.

$$A^i(x, y) = \frac{W-x}{W} \times A(x, y) \dots (2)$$

Where $\frac{W-x}{W}$ could be considered as a gradient decremented weight to the next pixel to the right.

The foremost problem is that the training samples we used are noise-free ones (i.e., samples which are segmented completely). However, in actual execution, the width of samples (i.e., the width of characters for recognition) could not be identified, which could be solved by two means:

- Extract only half width of characters to build training samples. For example, a 32 pixels - width X 32 --pixels height sample could be replaced by a 16 —pixels width X 32 pixels height one. In this way, the right-half noise could then be effectively ignored and the half width left is enough for character recognition (with considerably high recognition rate)
- Apply our proposed gradient descent KNN algorithm to calculate the weights of each pixel from left to right according to Equation 2 and thus make the left weight stand out.

C. Left to right segmentation-recognition Algorithm:

Find the leftmost location where exists a useful pixel and use a fixed width window to collect a sample from the candidate string. The algorithm ends

when the width of the string image is too small or the continuous space is not sufficient. The height and width of a sampling window are fixed to 32, respectively, in pixel units.

Since the width of sampling window is fixed, the sample collected might have more than one character in it. The very left useful pixel should be part of the first character, as the noisy small contours are removed from the image. Comparing the sample with the possible characters (target) in the training database by K-means, the distance d between the sample and target in database is calculated with Equation-3. If no match is found, the gradient descent KNN is applied to decrease the viewable weights for the right pixels in the sampling window, until a matched character could be found.

$$d(S, T) = \sqrt{(\sum_{x=1}^W \sum_{y=1}^W (S(x, y) - T(x, y))^2)} \dots (3)$$

Where, W is the width of the string image, $S(x, y)$ is the sample location with ordinates, and $T(x, y)$ is the target location with ordinates.

When the distance between the sample and target is within the predefined confident criteria, a character is found to be matched with the target in the database, and it can be removed from the candidate string image. Usually, the removed area has a curved boundary instead of a straight line one.

If the eliminated part of the string image fails to be removed, the iteration of the similar processes goes back to step of finding left most useful pixel, until the computer can recognize gradually the whole string.

Experimental Result and Discussion:

A. Numeral Strings

Totally 1000 connected numeral strings have been used for training and 955 samples with connected strokes for testing. Manual segmentation was performed on the 1000 training samples and the recognition modules were trained. The improved gradient descent KNN algorithm was used to perform recognition.

If the numerals of a full line are correctly segmented recognized, we count one correct segmentation and recognition; otherwise, it is incorrectly segmented and recognized. An accuracy of 89.35% has been achieved on the test samples. Fig.6 shows an example of correctly segmented and recognized connect numeral string.

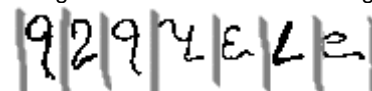


Fig. 6. Segmentation-recognition result of a numeral string

B. MODI script Character Strings

As to MODI characters samples, our algorithms can deal with a small character set (decades of characters). The accuracy can reach 57 were used to train the recognition modules. Figure 6 shows an example of correctly segmented and recognized MODI character string. Fig.7. Segmentation recognition result of a MODI script character string.

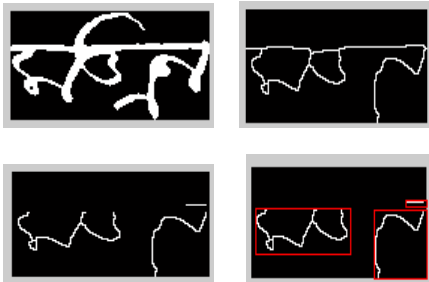


Fig. 7. Segmentation-recognition result of a MODI script character string

There are a few reasons for a high error rate in the segmentation and recognition of handwritten MODI character strings, which are discussed as follows:

- Horizontal lines often intersect with MODI characters, so they are difficult to be eliminated by analyzing connected components. Horizontal lines may result in false localization of characters;
- The set of MODI characters is very large and the differences between some characters are small, resulting in an error in either segmentation or recognition or both.

Conclusion:

We have presented an improved segmentation-recognition algorithm for dealing with connected handwritten characters in this paper. The proposed algorithm has several novel features:

- The number of connected characters that can be handled can be more than three.
- The proposed gradient descent KNN algorithm works effectively to segment characters from a string by gradually decreasing the effect from the right side of the character.
- The technique for small contour elimination assists the algorithm in locating the left-most useful pixel by removing noise after peeling the leftmost character off the string; And
- Matching the segment with the typical templates trained can peel off a character with an unusual curving boundary.

Experimental results on 1000 character strings from the self created database that the algorithm works robustly and efficiently. As it is expected, the algorithm works more effectively on numerals than on MODI characters because the set of MODI characters is much larger than the set of numerals. Some amelioration should be carried on in the future to improve the performance of the algorithm in recognizing connected handwritten MODI characters.

References:

1. Kimura F. and Sridhar M., "Segmentation-recognition algorithm for Zip code field recognition", pp. 199-210, *vis. Appl.* 5, 1992.
2. Cheriet M, Huang Y. S. and Suen C. Y., "Background region-based algorithm for the segmentation of connected digits." *Pattern Recognition*, pp. 619-622, 1992.
3. Magdi A. Mohamed, Paul D. Gader: "Handwritten Word Recognition Using Segmentation-Free Hidden Markov Modeling and Segmentation-Based Dynamic Programming Techniques". *IEEE*

Trans. Pattern Anal. Mach. Intell. 18(5): 548-554, 1996.

4. Strathy N. W., Suen C. Y., and Krzyza A., "Segmentation of Handwritten Digits Using Contour Features", *Document Analysis and Recognition*, pp. 577-580, 1993.
5. YILU and Sridhar M., "Character segmentation in handwritten words - An overview", *J. Pattern Recognition* Vol. 29, No. 1, pp.77-96, 1996.
6. Shaw B., Parui S. K. And Sridhar M., "Offline Handwritten Devanagari Word Recognition: A Segmentation based Approach", the 19th International Conference on Pattern Recognition (ICPR'08), Tampa, Florida, USA, December, 8-11, 2008.
7. Chi Z., Suters M., and Yan H., "Separation of single- and double-touching numeral strings," *Optical Engineering*, Vol. 34, No. 4, pp. 1159-65, 1995.
8. Chi Z. and Yan H., "Feature evaluation and selection based on an entropy measurement with data clustering," *Optical Engineering*, Vol. 34, No. 12, pp. 3514-9, 1995.
9. Chi Z. and Yan H., "Handwritten numeral recognition using a small number of fuzzy rules with optimized defuzzification parameters," *Neural Networks*, Vol. 8, No. 5, pp. 821-7, 1995.
10. Zhao S., Chi Z., Shi P., and Yan H., "Two-stage segmentation of unconstrained handwritten Chinese characters," *Pattern Recognition*, U.S.A , Vol. 36, No. 1, pp.145-156, January 2003.
11. Lu Z., Chi Z., and Siu W.C., "Extraction and optimization of B-spline PBD templates for recognition of connected handwritten digit strings," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, U.S.A , Vol. 24, no. 1, pp.132-139, January, 2002.
12. Lu Z., Chi Z., Siu W.C., and Shi P., "A background-thinning based approach for separating and recognizing connected handwritten digit strings," *Pattern Recognition*, Vol. 32, No. 6, pp. 921-933, June, 1999.
13. Chang F., Chen C. J. and Lu C.J., "A Linear-Time Component-Labeling Algorithm Using Contour Tracing Technique", *Computer Vision and Image Understanding*, vol. 93, no. 2, pp. 206-220, 2004.
14. Ha TM, Zimmermann M. and Bunke H., "Off-line handwritten numeral string recognition by combining segmentation-based and segmentation free methods", *Pattern Recognition*, vol. 31, no. 3, pp 257-272, 1998.
15. Shi Z. and Govindaraju V., "Segmentation and recognition of connected handwritten numeral strings", *Pattern Recognition*, vol. 30, no. 9, pp. 1501-1504, 1997.