2012 International Conference on Frontiers in Handwriting Recognition

Stroke Segmentation and Recognition from Bangla Online Handwritten Text

Nilanjana Bhattacharya
Bose Institute
Kolkata, India
nilibht@gmail.com

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

Abstract—**This paper deals with recognition of online handwritten Bangla (Bengali) text. Here, at first, we segment cursive words into strokes. A stroke may represent a character or a part of a character. We selected a set of Bangla words written by different groups of people such that they contain all basic characters, all vowel and consonant modifiers and almost all types of possible joining among them. For segmentation of text into strokes, we discovered some rules analyzing different joining patterns of Bangla characters. Combination of online and offline information was used for segmentation. We achieved correct segmentation rate of 97.89% on the dataset. We manually analyzed different strokes to create a ground truth set of distinct stroke classes for result verification and we obtained 85 stroke classes. Directional features were used in SVM for recognition and we achieved correct stroke recognition rate of 97.68%.**

Keywords-Online character segmentation, online recognition, handwriting recognition, Bangla script, Indian text.

I. INTRODUCTION

Handwriting recognition is a difficult task because of the variability involved in the writing styles of different individuals. Writing two or more characters by a single stroke is another difficulty for online character recognition. Segmentation is one of the important phases of handwriting recognition in which data are represented at character or stroke level so that nature of each character or stroke can be studied individually. A number of studies [1-2] have been done for offline recognition of printed Indian scripts like Bangla, Devanagari, Gurmukhi, Tamil, Telugu, Oriya, etc. Some works are available in segmentation of offline Bangla handwriting [3-5]. In the earliest available work on segmentation of handwritten cursive Bangla words [3], a recursive contour following approach was proposed. In [4], water reservoir principle based technique was used for segmentation of handwritten Bangla word images, where the "water reservoirs" were considered as the cavities between two consecutive characters. A fuzzy feature based segmentation technique for Bangla word images was proposed in [5].

Both segmentation as well as recognition of online Bangla handwriting is yet to get full attention from researchers. Some works are available on online isolated Bangla character/numeral recognition in [6-10]. In [11], handwritten words were segmented estimating the position of headline of the word. Preprocessing operations such as smoothing and re-sampling of points were done before feature extraction. They used 77 features considering 9 chain-code directions. Modified quadratic discriminant function (MQDF) classifier was used for recognition. In [12], the authors used sub-stroke segmentation before recognition. They divided each stroke of the preprocessed word sample into several sub-strokes using the angle incurred while writing. We think, because of vast variability of Bangla writing styles, we can have different sub-strokes from different handwritings for a single stroke.

An approach for stroke segmentation and recognition from Bangla online handwritten text is proposed here. The algorithm is robust against various types of stroke connections as well as shape variations. For segmentation of text into strokes, we discovered some rules analyzing different joining patterns of Bangla characters. Combination of online and offline information was used for segmentation. We manually analyzed different strokes and obtained 85 distinct stroke classes. Directional features of 64 dimensions are extracted to recognize the segmented strokes using SVM classifier.

The rest of our paper is organized as follows. In Section II, we discuss some properties of the Bangla script. Data collection is described in Section III followed by segmentation algorithm in Section IV. Stroke analysis, and feature extraction and classification are described in Section V and VI, respectively. The experimental results are discussed in Section VII. Finally, conclusion is given in Section VIII.

II. PROPERTIES OF BANGLA

Bangla is the second most popular language in India and the fifth most popular language in the world. More than 200 million people speak in Bangla and Bangla script is used in Assamese and Manipuri languages in addition to Bangla language. The set of basic characters of Bangla consists of 11 vowels and 39 consonants. As a result, there are 50 different shapes in the Bangla basic character set. The concept of upper/lower case is absent in this script. Fig. 1 shows ideal (printed) forms of these 50 basic character shapes.

In Bangla, a vowel (except for the first vowel) can take modified form and we call it a vowel modifier. Ideal

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.275

(printed) shapes of these vowel modifiers corresponding to 10 vowels with a basic character KA are shown in fig. 2. Similarly consonants can also take modified form. Fig. 3 shows consonant modifiers with a basic character BA.

Figure 1. Bangla basic characters (vowels are in green, consonants in brown) and their respective codes for future reference.

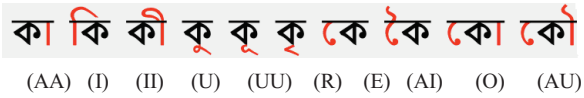


Figure 2. Vowel modifiers of Bangla and their respective codes with basic character KA.

Figure 3. Consonant modifiers of Bangla and their respective codes with basic character BA.

Unconstrained Bangla handwriting is usually cursive. In one stroke (a stroke is a collection of points from pen down to pen up), writer can write a part of a character or one or more characters. In our experiment we found that a single stroke may contain upto 4 characters and 2 modifiers. Also in Bangla, the most of the touchings of characters in a word occur in the region of word's headline or sirorekha portion [1] in contrast to English handwriting where the touchings occur in the lower part of the word shape.

On the other hand, several single characters are written in variety of ways – in a single stroke or in more than one stroke. From statistical analysis it is found that the minimum number of stroke used to write a Bangla character is 1 and maximum number is 6. Hence online recognition of Bangla is a difficult task.

III. Data Collection and Ground Truth Generation

A set of 2000 Bangla words written by 50 writers were collected using Wacom tablet. No restriction was imposed on writing except that they were requested to use all basic characters, all vowel and consonant modifiers in their words. Input data consist of (x, y) coordinates along the trajectory of the pen together with positions of pen-downs (stroke starting points).

We have built a text file with ground truths of segmentation for all input word files. Each row of this file contains input filename, number of ideal segmentation points and their x, y co-ordinates. For each input file, output segmentation points are compared with ground-truth-file and accuracy is automatically calculated without manual intervention. Similarly ground truth file is created for automatic recognition accuracy calculation. Each row of this file contains input filename, stroke ids of segmented strokes.

IV. Proposed Segmentation Approach

For better segmentation, combination of online and offline information has been used here. Except for vowel modifiers U, UU, R and consonant modifier RR, touchings occur mostly in the upper portion of the word. Considering this fact our segmentation steps are as follows.

A. Segmentation steps

1) Make an offline word image from online input data file.
2) Find horizontal histogram of the offline image.
3) Identify approximate busy zone from the horizontal histogram (Busy-zone of a word is the region of the word where most of the character parts lie.). Busy zone is defined by two lines- TOP_LINE and BOTTOM_LINE (fig. 4). Now we define up and down zones. From topmost row of the word to (TOP_LINE + t1) row is up zone and (TOP_LINE + t2) row to down most row of the word is down zone. Here, t1=height of busy zone/3. t2= height of busy zone/2. Height of busy zone= BOTTOM_LINE - TOP_LINE.
4) Describe all points as up, down or don't know points according to their belonging to up zone, down zone or outside these zones. From now, we consider only up and down points.
5) If the pen tip goes from down zone to up zone and then again come to down zone, two characters or modifiers may be touching in the up zone and hence the stroke should be segmented (fig. 5). Because of this, for each stroke, we find patterns like "down->up->down", i.e. "any number of down points followed by any number of up points

followed by any number of down points" within the stroke. For such pattern, we segment at the highest point of up zone of the touching. We call such segmentation point as candidate segmentation point. For each stroke we can get zero, one or more than one such candidate points.

6) For "down->up->down" stroke, from the first "down", find down most point. From second "down" also find the down most point. Find the point with higher row value among these two points. Call it "HIGHER_DOWN".

7) Validate the candidate points. Using positional information and stroke pattern, two levels of validations are performed here. Detailed description of validation of candidate points is provided below.

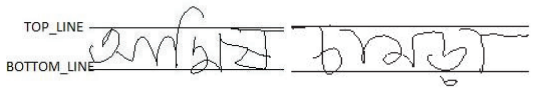


Figure 4. TOP_LINE and BOTTOM_LINE of busy zone for 2 samples.

Figure 5. Touching of AA and MA (up->down->up->down->up form).

B. Validation of candidate points at Level-1

This validation is done to check the position of the candidate point with respect to the position of HIGHER_DOWN, BOTTOM_LINE of busy zone, and also with respect to stroke height to avoid incorrect over-segmentation. The following four conditions are tested:

1) r(HIGHER_DOWN) − r(candidate point) > (height of busy zone*40%)

2) r(HIGHER_DOWN) − r(candidate point) > (height of the stroke*30%)

3) r(BOTTOM_LINE) − r(candidate point) > (height of busy zone*60%)

4) r(down most point of the stroke) − r(candidate point) > (height of the stroke*40%)

where r(x) means row value of x.

If all of these 4 conditions are satisfied by the candidate segmentation point, it is a valid segmentation point.

C. Validation of candidate points at Level-2

We implement some rules which we discovered by analyzing stroke patterns of Bangla writing. Our observations are as follows:

Case A: As Bangla writing goes from left to right, end point of a stroke consisting of more than one character is always at the right side of the start point.

Case B: If the stroke consists of only a character or a part of a character this relationship between start point and end point does not always hold.

As we want to segment the strokes which consists of more than one character, we will consider only case-A for segmentation. So our segmentation rules are as follows:

a) End point of a connected stroke should be at the right side of start point of the stroke, i.e. c(end point) > c(start point), where c(x) means column value of x. Otherwise, candidate segmentation point is cancelled.

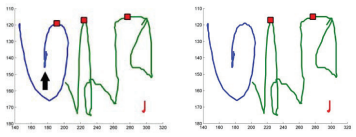
b) End point of a connected stroke should be at the right side of previous validated segmentation point of the stroke, i.e. c(end point) > c(previous segmentation point). Otherwise, candidate segmentation point is cancelled.

c) Any candidate segmentation point (except for the first one) should be at the right side of previous candidate segmentation point of the stroke, i.e. c(candidate segmentation point) > c(previous candidate segmentation point). Otherwise, previous candidate segmentation point is marked to be deleted.

These rules prevent over-segmentation of 15 characters having codes: A, AA, BHA, TA, E, AI, NYA, U, UU, JA, DDA, RRA, NGA, O and AU.

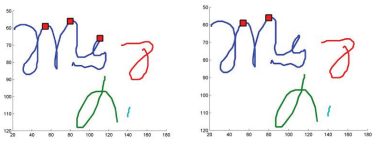
Examples of some of the results obtained before and after Level-2 validation are shown in fig. 6-8. Different strokes of input word are depicted in different colors and the segmentation points are shown in red on the strokes.

Three modifiers II, AU and YA may go from right to left. A stroke containing these may not be segmented because of the above validations. (fig. 9). So we check whether the latest "down" portion of the stroke goes under the start point or previous segmentation point of the stroke. If yes (true for the 15 characters specified above and modifier YA), candidate segmentation point is cancelled. If no (for modifiers II and AU), candidate segmentation point is valid. For modifier YA, another checking is necessary. We check the length L of the stroke from start point or previous segmentation point to the point just before the last "down" portion of the stroke. Since the combined length of character and YA should be greater than the length of part of a character, if L is greater than a threshold, candidate segmentation point is valid. The threshold is found empirically. Fig. 10 shows corrected results after these checking.



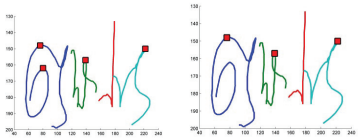
(i) (ii)

Figure 6. (i) Before applying Rule-(a): A is over-segmented. Start point is indicated by an arrow. (ii) After applying Rule-(a).



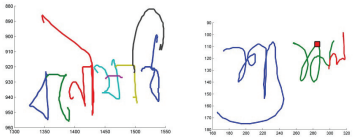
(i) (ii)

Figure 7. (i) Before applying Rule-(b): NGA is over-segmented. (ii) After applying Rule-(b).



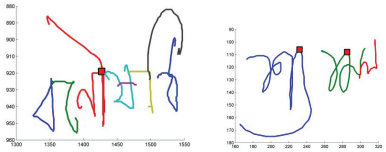
(i) (ii)

Figure 8. (i) Before applying Rule-(c), (ii) After applying Rule-(c).



(i) (ii)

Figure 9. Errors introduced by validations for modifiers AU and YA: words are under- segmented.



(i) (ii)

Figure 10. Corrected results obtained: here both the words of fig. 9 are properly segmented.

V. STROKE ANALYSIS

At first we have done a general analysis on Bangla alphabet to find the number of stroke classes which are sufficient to cover all characters and modifiers. If parts of different characters look similar, they are assigned with a single stroke-id. On the other hand, stroke classes representing one particular character differ from writer to writer. For example, fig. 11(ii) and fig. 11(iv) shows two GAs written by different writers. The left stroke of first GA (fig. 11(ii)) is similar to the right stroke of KA (fig. 11(i)). Also the left stroke of second GA (fig. 11(iv)) is similar to the left stroke in SA (fig. 11(iii)). Hence, in the ground truth file, their codes are also considered similar.

Next we analyze the stroke classes with respect to our segmentation algorithm. We get 11 additional stroke classes because of over-segmentation. If we consider all types of joining between characters and modifiers, we find that some characters can be joined with vowel modifiers like U, UU, R and consonant modifiers like R, RR within a single stroke. As we do not try to segment these modifiers from characters, we consider these joined strokes as separate stroke classes. Finally, we find the set of 85 distinct stroke classes. A few examples of stroke classes are given in fig. 12. Table I shows the characters in which these strokes are used.

(i) (ii) (iii) (iv)

Figure 11. (i) KA, (ii) GA, (iii) SA, (iv) GA. Right (black) stroke of KA and left (black) stroke of GA in (ii) are the same. Left (green) stroke of SA and left (black) stroke of GA in (iv) are the same.

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

Figure 12. Examples of some stroke classes.

TABLE I. STROKES OF FIG. 12 AND THEIR RESPECTIVE CHARACTERS

Stroke of fig. 12	Characters in which the stroke is used
(a)	A, NA, NNA, LA, JHA, R.
(b)	KA, PHA, ANUS, GA.
(c)	O, AU, NYA, GA with modifier U, SHA with modifier U.
(d)	YY, YYA, SSA, PHA, KHA, THA.
(e)	DDA, RRA, U, UU, JA.

VI. FEATURE EXTRACTION AND CLASSIFICATION

We use 64-dimensional feature vector for high-speed stroke recognition. Each stroke is divided into 4x4 cells, i.e. 16 cells and frequencies of the direction codes are computed in these cells. We use chain code of four directions only [0 (horizontal), 1 (+45 degrees from positive x-axis), 2 (vertical) and 3 (+135 degrees from positive x-axis)]. Fig. 13 illustrates chain code directions. We assume chain code of direction 0 and 4, 1 and 5, 2 and 6, 3 and 7, are the same because we find that strokes of characters BA, LA, PA, GA and modifiers E, II, AU, R (consonant) can be written with different orders of pen points within the stroke making the directions just opposite. Thus, for each cell we get four integer values representing the histograms of the four direction codes. Thus 16x4=64 features are found for each stroke. These

features are normalized by dividing by the maximum value.

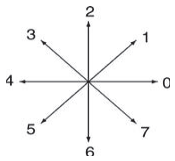


Figure 13. Chain code directions for feature computation.

In our experiment, we have used a Support Vector Machine (SVM) classifier. The SVM is originally defined for two-class problems and it looks for the optimal hyper plane which maximizes the distance and the margin, between the nearest examples of both classes, named support vectors (SVs). Given a training database of M data: $\{x_m| \ m=1... \ M\}$, the linear SVM classifier is then defined as:

$$f(x) = \sum_j \alpha_j x_j \cdot x + b$$

where $\{x_j\}$ is the set of support vectors and the parameters α_j and b have been determined by solving the quadratic problem [13]. The linear SVM can be extended to various non-linear variants; details can be found in [13, 14]. In our experiments, the Gaussian kernel SVM outperformed other non-linear SVM kernels, hence we report our recognition results based on the Gaussian kernel only. Since SVM is available elsewhere, we do not present the details of SVM here.

We have a total number of 10,896 stroke samples. 50% of these samples are used for training and rest for testing.

VII. Results and Discussions

Detailed segmentation result is given in Table II. We noted that 97.89% of the strokes are segmented correctly. Fig. 14 shows a few examples of correctly segmented strokes while fig. 15 shows some examples of incorrectly segmented strokes. In fig. 15(i), modifier AA is not segmented because its height is small and it does not reach the down zone. In fig. 15(ii), modifier I is not segmented because it does not reach the down zone. In fig. 15(iii), character NA is over-segmented because it reaches from down zone to up zone and then it comes to down zone. This part of NA should not reach the up zone in ideal case. Similarly, in fig. 15(iv), character CHA is over-segmented as it reaches the up zone.

Recognition result is given in Table III. From the experiment on 5,448 test samples we obtained 97.68% stroke recognition accuracy. Sample set of 85 stroke classes includes whole characters as well as part of characters. Characters GHA, YY, THA, KHA, PHA look very similar and hence generate some misclassifications. Similarly, CA and DDHA look very similar and generate some errors.

To get idea about the performance of our method, we report here some of the published results. In [10], authors reported basic character recognition accuracy of 81.55% (using point-float feature in HMM) to 91.01% (using chain-code feature in Nearest Neighbour classifier) on 8,616 test character samples where samples include 50 basic characters. In [11], authors selected a lexicon of 100 Bangla words and reported that 3.1% of the strokes segmented suffered from under segmentation. Only properly segmented strokes were used for training and testing of the classifier. Recognition error obtained was 1.22% at stroke level considering 73 stroke classes. In [12], authors reported recognition accuracy of 88% (for holistic recognition) to 93.1% (for context dependent recognition) on 6,516 test word samples where samples include 50 Indian city names.

TABLE II. Detailed Segmentation Result

Number of ideal segmentation points	Number of points under-segmented	Number of points correctly segmented	Number of points causing over-segmentation
2698	57 (2.11%)	2641 (97.89%)	532 (16.47%)

TABLE III. Stroke Recognition Result

Average Recognition rate	Average Error rate
97.68%	2.32%

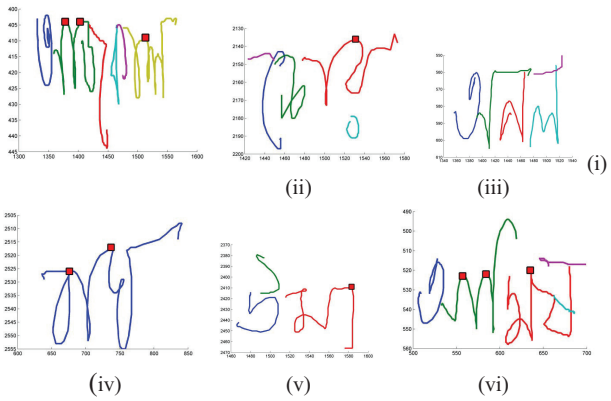


Figure 14. Examples of words which are correctly segmented.

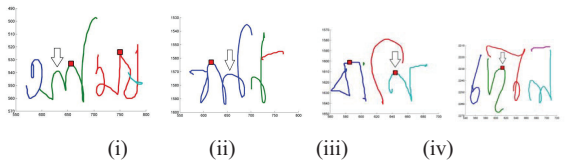


Figure 15. Examples of words which are not segmented correctly (first two words are under segmented, next two words are over segmented). Arrows indicate the positions where under-segmentations and over-segmentations have occurred.

VIII. CONCLUSION

This paper deals with a rule-based scheme to segment online handwritten Bangla (Bengali) cursive words into strokes. It also deals with recognition of the segmented strokes where directional features are used in SVM classifier. Both offline and online information were used for segmentation. Different writing rules of Bangla helped us to validate the candidate points. We obtained stroke segmentation rate of 97.89% from our dataset. From 85 distinct stroke classes, we obtained 97.68% accuracy in stroke recognition without using any pre-processing scheme. Our dataset along with ground truth files will be available for research purpose.

REFERENCES

[1] U. Pal, R. Jayadevan, and N. Sharma, "Handwriting Recognition in Indian Regional Scripts: A Survey of Offline Techniques", ACM Trans. Asian Lang. Inf. Process. 11(1): 1, 2012.

[2] R. Jayadevan, S. R. Kolhe, P. M. Patil, and U. Pal, "Offline Recognition of Devanagari Script: A Survey", IEEE Transactions on Systems, Man, and Cybernetics, Part C 41(6): 782-796, 2011.

[3] A. Bishnu and B. B. Chaudhuri, "Segmentation of Bangla handwritten text into characters by recursive contour following," in Proc. Int. Conf. on Document Analysis and Recognition, 1999, pp. 402–405.

[4] U. Pal and S. Datta, "Segmentation of Bangla Unconstrained Handwritten text," In Proc. 7th ICDAR, pp. 1128-1132, 2003.

[5] S. Basu, R. Sarkar, N. Das, M. Kundu, M. Nasipuri, and D. K. Basu, "A fuzzy technique for segmentation of handwritten Bangla word images," in Int. Conf. on Computer: Theory and Application, 2007, pp. 427–433.

[6] U. Garain, B. B. Chaudhuri, and T. Pal, "Online Handwritten Indian Script Recognition: A Human Motor Function Based Framework," In Proc. 16th Int. Conf. on Pattern Recognition, pp. 164-167, 2002.

[7] K. Roy and N. Sharma, T. Pal, and U. Pal, "Online Bangla Handwriting Recognition System" , In Proc. 6th International Conference on Advances in Pattern Recognition, pp. 121-126, 2007.

[8] S. K. Parui, U. Bhattacharya, B. Shaw, and K. Guin, "A Hidden Markov Models for Recognition of Online Handwritten Bangla Numerals", Proceedings of the 41st National Annual Convention of Computer Society of India, India, pp 27-31, 2006.

[9] U. Bhattacharya, B. K. Gupta, and S. K. Parui, "Direction Code Based Features for Recognition of Online Handwritten Characters of Bangla", Proc. of the 9th ICDAR, vol. 1, pp. 58-62, 2007.

[10] T. Mondal, U. Bhattacharya, S. K. Parui, and K. Das, "On-line handwriting recognition of Indian scripts – the first benchmark", 12th International Conference on Frontiers in Handwriting Recognition, 2010, pp. 200-205.

[11] U. Bhattacharya, A. Nigam, Y. S. Rawat and S. K. Parui, An analytic scheme for online handwritten Bangla cursive word recognition. Proc. of the 11th ICFHR, pp. 320-325, 2008.

[12] Gernot A. Fink, Szil´ard Vajda, Ujjwal Bhattacharya, Swapan K. Parui, and Bidyut B. Chaudhuri, "Online Bangla Word Recognition Using Sub-Stroke Level Features and Hidden Markov Models", 12th International Conference on Frontiers in Handwriting Recognition, 2010, pp. 393-398.

[13] V.Vapnik, "The Nature of Statistical Learning Theory ", Springer Verlang, 1995.

[14] C. Burges, "A Tutorial on support Vector machines for pattern recognition", Data mining and knowledge discovery, pp.1-43, 1998.