5

Development of a recognizer for Bangla text: present status and future challenges

Saima Hossain, Nasreen Akter,
Hasan Sarwar and Chowdhury Mofizur Rahman
United International University
Bangladesh

1. Introduction

Optical Character Recognition (OCR) System, by virtue of its usefulness, has emerged as a major research area since 1950. Now it is becoming a more challenging issue all over the world to have efficient and more accurate recognizers. There are many widely spoken languages in the world like Chinese, Arabic, Hindi, English, Spanish, Bangla, Russian, Japanese etc. Bangla is one of the most widely spoken languages, ranking 5th in the world. 21st February is observed as the international mother language day to pay homage to the martyrs fought for the establishment of Bangla as the mother tongue of Bangladesh. With the automation everywhere, it is a burning issue to digitize huge, volume of Bangla documents by using an efficient OCR. However as of today there is no such good recognizer available for Bangla compared to other languages. From 80s, it took huge interest and now becomes as a major research area particularly in Bangladesh and India. Lots of works have been done in different sections of pattern recognition tasks (i.e, pre-processing, segmentation, feature extraction, classification) but there is a lack of synchronization between these works. That is why we put our effort into a comprehensive review of the current status of research to develop an all-inclusive Bangla OCR which will enable one to understand the difficulties and challenges involved, to know how much progress has been done and to estimate what more to be done to come out with a successful Bangla OCR.

2. Overview

This chapter includes introduction to Bangla language which provides a brief description on it, present status on the development of Bangla OCR and some comparative analysis of several proposed methods on Noise Reduction, Skew Detection and Correction, Segmentation, Feature Extraction, Classification published in different articles. Future works in this field explaining the futures challenges are summarised at the end of this chapter.

3. Introduction to Bangla Language

Bangla is an eastern Indo-Aryan language and evolved from Sanskrit (Barbara F. Grimes, 1997). The direction of the writing policy is left to right. Bangla language consists of 50 basic characters including 11 vowels and 39 consonant characters and 10 numerals. In Bangla, the concept of upper case or lower case letter is not present. Bangla basic characters have characteristics that differ from other languages. Bangla character has headline which is called matraline or matra in Bangla. It is a horizontal line and always situated at the upper portion of the character. Among basic characters, there are 8 characters which are with half matra, 10 characters with no matra and rest of them with full matra. All consonants except ঙ এ ঞ ণ ড় ঢ় ন ৎ ং ঃ ঁ are used as the starting character of a word whereas, vowels are used everywhere. Vowels and consonants have their modified shapes called vowel modifiers and consonant modifiers respectively. Both types of modifiers are used only with consonant characters. There are 10 vowels and 3 consonant modifiers which are used before or after a consonant character, or at the upper or lower portion of a consonant character or on the both sides of a consonant character, likewise, খা খি খু খে খৌ ক্য র্ক ক্র. In Bangla, some special characters are there which are formed by combining two or more consonants and acts as an individual character. These types of characters are known as compound characters. The compound characters may further be classified as touching characters and fused characters. Two characters placed adjacent contact to each other produce a touching character. Touches occur due to horizontal placement of only two characters and/or vertical placement of two or more characters. About 10 touching characters are there in Bangla. Fused characters are formed with more than one basic character. Unlike touching characters, the basic characters lose their original shapes fully or partly. A new shape is used for the fused characters. In sum, there are about 250 special characters in Bangla except basic and modified characters. Table 3.1 illustrates different types of Bangla characters.

Vowels	অ আ ই ঈ উ ঊ ঋ এ ঐ ও ঔ
Consonants	ক খ গ ঘ ঙ চ ছ জ ঝ ঞ ট ঠ ড ঢ ণ ত থ দ ধ ন প ফ ব ভ ম য র ল শ ষ স হ ড় ঢ় য় ৎ ং ঃ ঁ
Vowel Modifiers	া ি ী ু ূ ৃ ে ৈ ো ৌ
Vowel Modifiers attached with consonants	খা খি খী খু খূ খৃ খে খৈ খো খৌ
Consonant Modifiers	্য ্র ্
Consonant Modifiers attached with consonants	ক্য র্ক ক্র

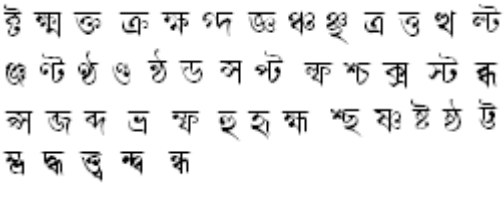
Compound Characters: Horizontal Touching Characters	ড + ড = ড্ড ব + ব = ব্ব হ + ব = হ্ব চ + চ = চ্চ চ + ছ = চ্ছ ধ + ব = ধ্ব ট + ন = ট্ন ঠ + ন = ঠ্ন
Compound Characters: Vertical Touching Characters	
Compound Characters: Fused Characters	
Numerals	০ ১ ২ ৩ ৪ ৫ ৬ ৭ ৮ ৯

Table 3.1. Different types of Bangla characters. A subset of 112 compound characters out of about 250 characters (B.B. Chaudhuri, 1998) is shown here.

The occurrence of vowels and consonants are larger compared to special characters in most of the Bangla documents. A statistical analysis, shown in Table 3.2, took 2 sets of data populated with 100,000 words from Bangla books, newspapers and 60,000 words from Bangla dictionary respectively (B.B. Chaudhuri, 1998).

Global characteristics	1st set of data	2nd set of data
Vowel characters	38.70%	37.10%
Consonant characters	61.30%	62.90%
Compound characters	4.10%	7.20%
Entropy of the alphabet	4.07 (bits/char)	3.54 (bits/char)
Average word length	5.73 (char)	4.99 (char)
Words with suffix	69.80%	0.00%

Table 3.2. Statistical analysis on the occurrence of different characters

Some of the modifiers are there which are used on top of the character (basic or special character) as well as a few in the bottom of the character. Again some basic characters also

have upper portion which is treated as a part of the character. So it can be said that the construction of Bangla characters require 3 zones named upper zone, middle zone and lower zone. Upper portion from matra line is called upper zone. Middle portion that is situated under the matraline is called middle zone. As some modifiers are used at the bottom of the middle zone, this portion is called lower zone. Most of the characters are situated in the middle zone. Fig. 3.1 shows a simple example explaining the construction of a Bangla word.

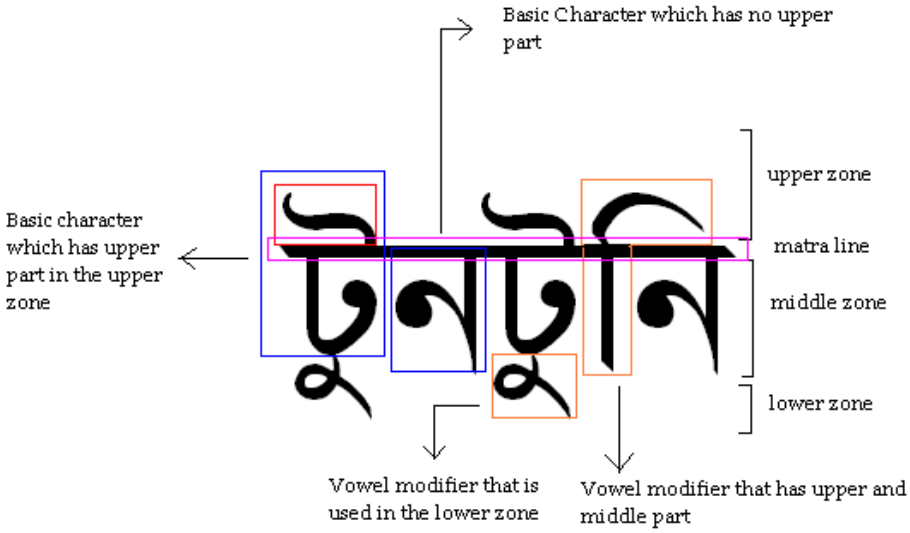


Fig. 3.1. Dissection of a Bangla Word

4. Scanning and Image Digitization

Before going into the ocr process, one must scan the paper through a flat-bed scanner. It is better not to use hand-held scanner, which may create local fluctuation for hand movement (B.B. Chaudhuri, 1998). It is crucial to have good quality printed document scanning. If the quality is poor and the color contrast is too low, it will be hard for the OCR software to read the text and to make correct interpretation. The scanned image is stored, for example, as a jpeg/bmp format file which is converted to a binary image. In order to improve the quality of the image to make the OCR correct interpretation, noise reduction and elimination and skew detection and correction processes are performed.

5. Noise Detection and Removal

Noise is naturally added during scanning process. When documents or papers are scanned, some noises are added automatically into it. There are two different types of noises known as background noise and salt and pepper noise which are given most importance. A histogram-based thresholding approach is used to convert gray tone into two-tone images

(B.B. Chaudhuri, 1998). The histogram shows two prominent peaks corresponding to white and black regions. The threshold value is chosen as the midpoint of the two histogram peaks. The two-tone image is converted into 0-1 labels where 1 and 0 represent object and background, respectively. Authors also claim that their approaches are better than J. N. Kapur, P. K. Sahoo and A. K. C. Wong (J. N. Kapur, 1985), and N. Otsu (N. Otsu 1979). However, salt and pepper noise is not mentioned here. Different authors have suggested for applying noise elimination process at different stages of Preprocessing or Segmentation. (B.B. Chaudhuri, 1998) and (A. Roy, 2002) used it during binarization of the image.

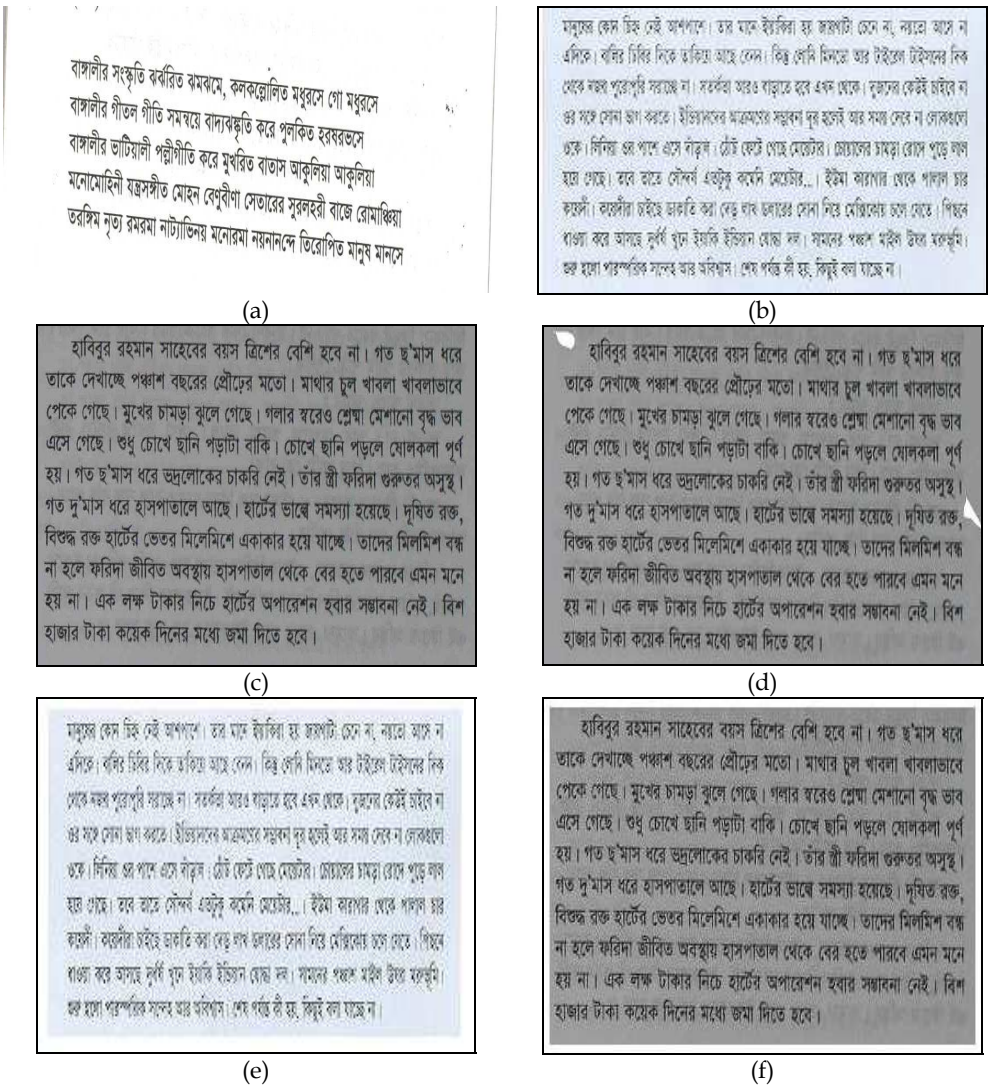


Fig. 5.1. Images (a) – (f) with different types of noises

In (A. Roy, 2002), gray-level images are median filtered and then Otsu's thresholding algorithm is used to binarize the images of word. The binary images are then filtered to obtain smooth images. In (Jalal Uddin Mahmud, 2003), noise is removed from character images. Noise removal includes removal of single pixel component and removal of stair case effect after scaling. Stair case effect occurs when the scaled characters have junctions so thin that inner and outer contour required for chain code representation cannot be found. Each pixel has been replaced by a filtering function to avoid such effect. However, this does not consider background noise and salt and pepper noise. In (Md. Abul Hasnat, 2007), the authors used connected component information and eliminated the noise using statistical analysis for background noise removal. For other type of noise removal and smoothing they used wiener and median filters (Tinku Acharya, 2005). Connected component information is found using boundary finding method (such as edge detection technique). Pixels are sampled only where the boundary probability is high. This method requires elaboration in the case where the characteristics change along the boundary. A comparative performance study in Table 5.1 is given below for some of the images shown in fig. 5.1.

Papers	Fig. 5.1a	Fig. 5.1b	Fig. 5.1c	Fig. 5.1d	Fig. 5.1e	Fig. 5.1f
B.B. Chaudhuri, 1998	yes	yes	x	x	yes	x
A.Roy, 2002	yes	yes	yes	yes	yes	x
Jalal Uddin Mahmud, 2003	yes	yes	Yes	yes	yes	x
Md. Abul Hasnat, 2007	yes	yes	yes	yes	yes	x

Table 5.1. Applicability of different techniques on noisy images shown in fig. 5.1

6. Skew Detection and Correction

Skew is basically an angle that is created due to an angular placement of document in the scanner. (B.B. Chaudhuri, 1998) says that it can be corrected in two steps, i) estimation of skew angle θ_s and ii) rotation of image by θ_s in the opposite direction. Many skew detection and correction algorithms are available. B.B. Chaudhuri (1998) suggests an approach suitable for Bangla scripts. Basically, it tries to detect the head line of document words. Head line is a straight line given on the upper side of a character when used in words. It has been found that average length of Bangla words is six characters (R. M. Bozinovic, 1989), 30-35% of characters are vowel modifiers having very little contribution to head word, 5% is compound character. In Bangla, 41 characters can appear in the first position, of them 30 characters have headlines. Probabilistic analysis reveals that, in 99.39% cases, there will be at least one character with head line in a Bangla word. In this process, firstly a bounding box (an upright rectangle containing a word/component) is defined. The mean box width is b_m and the standard deviation is b_s. Components having boundary box equal to b_m and less than $b_m + 3b_s$ are retained, others are discarded, which fall into categories like dots, punctuation marks, isolated characters and characters without headlines. Next, upper envelop of selected component, G, is found. From each pixel of the uppermost row of the bounding box, a vertical scan is performed until a pixel labelled G is encountered; it is converted into U label, known as the upper envelope. Hough transform technique may be

applied on the upper envelopes for skew estimation. (B.B. Chaudhuri, 1998) has suggested a new idea which is faster, robust and accurate compared to Hough transform. The idea is based on Digital Straight Line (DSL). The upper envelope may contain non-linear parts which require deletion, for which chain code representation has been used. Conditions for straightness of chain code digital arc are given in (J. N. Kapur, 1985).
A subset of DSL is known as SDSL. SDSL consists of runs of pixels in at most two directions which differ by 45^0. For runs of two directions, the run lengths in one of the directions are always one. The run length in the other direction can have at most two values differing by unity. An example is shown in fig-6.1. Here, the angle between two directions d1 and d2 is 45° and run lengths in d1 direction are two (n) or three (n+1) occurring alternately.

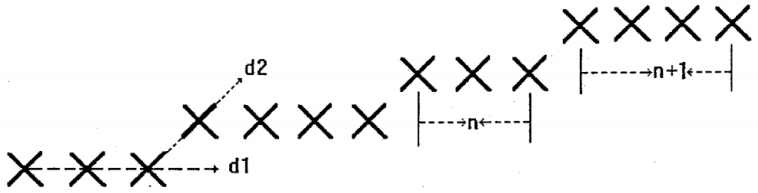
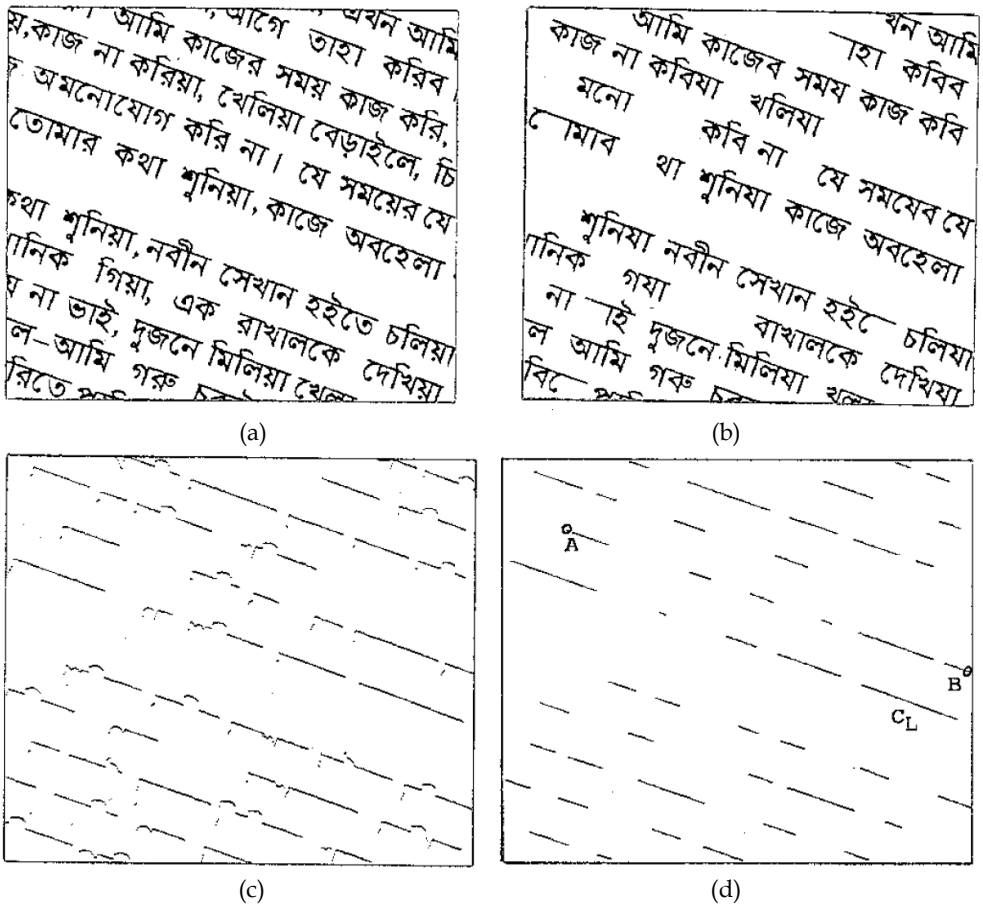


Fig. 6.1. Example of a digital straight line (DSL). Here, "X" denotes DSL pixel

A reasonable number of SDSLs can be used to find the average estimate of skew angle, which is formed between the first and last pixel of every SDSL with horizontal direction. To get a better estimate, clusters of SDSLs are found. Each line of text contains a number of SDSLs, which are grouped into a cluster. The SDSLs that have equal normal distances from a reference point are considered to be members of same cluster. The leftmost and the rightmost pixels of each cluster is taken and considered as the leftmost and rightmost coordinate of a line represented by that cluster. The Skew estimation algorithm is:

- STEP 1: Find connected components in the binary document image and find the mean b_m and standard deviation b_s of their bounding box widths.
- STEP 2: Choose the set S of connected components having bounding box width greater than or equal to b_m and less than $b_m + 3b_s$.
- STEP 3: For each component in S find the upper envelope described above. From each envelope component, find the SDSLs. If more than one SDSL is found choose only the longest one and form the subset R_1. Let the longest SDSL in R_1 be C_L.
- STEP 4: From the line C_L or its continuation, find the normal distances to the leftmost pixel of other SDSLs of R_1.
- STEP 5: Cluster the SDSLs of R_1 corresponding to individual text lines and find the leftmost and rightmost pixel of each cluster, as described above.
- STEP 6: For each cluster find the angle of line joining the leftmost and rightmost pixels (e.g., A and B in Fig. 6.2d) with horizontal direction.
- STEP 7: Average of such angles over all clusters (e.g., text lines) gives an accurate estimate of the skew angle.



(a) (b)

(c) (d)

Fig. 6.2. Skew detection approach (Bangla). (a) An example of Bangla skewed text; (b) Selected components from fig. 6.2a; (c) Upper envelope of selected components of fig. 6.2b; (d) SDSL components of fig. 6.2c.

The efficiency of the above described method (A3) in table 6.1 has been measured and compared with Hough transform applied on original document (A1) and on SDSLs (A2), shown in that table. The average execution times for a document of 512X512 pixels on a SUN 3/60 (with Microprocessor MC68020, and SUN O.S. Version 3.0) machine are 620, 312, and 17.80 seconds for methods A1, A2 and A3, respectively shown by (B.B. Chaudhuri, 1997). It is shown that the methods A2 and A3 are statistically equally accurate. However method A3 takes less execution time.

True skew angle (in deg.) (manual)	Mean and SD of estimated skew angles using method					
	A_1		A_2		A_3	
	mean	SD	mean	SD	mean	SD
40	40.396	0.285	40.034	0.256	39.889	0.301
20	20.174	0.439	20.049	0.3162	20.047	0.242
10	10.271	0.393	10.166	0.201	10.112	0.323
5	5.064	0.458	4.962	0.213	5.188	0.233
2	1.986	0.396	2.151	0.234	2.054	0.307

For each true skew angle the statistics is computed over 20 document images.
A_1: Hough transform over total image.
A_2: Hough transform over SDSLs of upper envelop.
A_3: Proposed quick method.

Table 6.1. Mean and Standard Deviation (SD) of Estimated Skew Angles Obtained by Different Methods shown by (B.B. Chaudhuri, 1997)

7. Segmentation

This is the most vital and important portion for designing an efficient Bangla OCR because feature extraction and recognition process depends on this phase to make the recognition process successful. The output of this phase consists of individual images of basic, modified and compound characters. Segmentation process includes the following steps. They are:

1. Line Detection
2. Matraline or Headline detection
3. Baseline Detection
4. Word Segmentation
5. Character Segmentation

7.1 Line Detection Process

Generally, a document is written in multiple lines considering one or more than one columns. In this chapter, one columned document is considered. The lines of a text block are detected by finding continuous white pixels between two consecutive matralines. Fig: 7.1.1 shows the result.

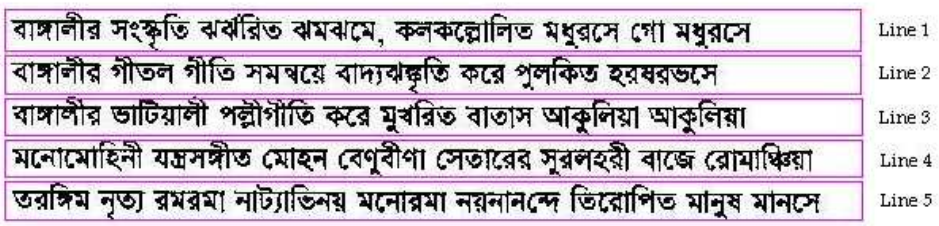


Fig. 7.1.1. Text line detection in a document image

7.2 Matraline or Headline Detection Process

Matraline or headline is an important and distinct feature in bangla. It connects the bangla components together. The matraline consists of highest number of black pixels compared to the upper, middle and lower zone. Under the matraline, the basic shapes of the characters are found. So, if matraline can be detected correctly, it helps to segment the characters in a more flexible way. The row with the highest frequency of black pixels is detected as matraline or headline. It is observed that the height or thickness of the matraline increases in case of larger font size. In those cases we get more rows having similar frequency or nearly close to the row of highest frequency. In order to detect the matraline with its full height, the rows with those frequencies are also treated as matraline (Jalal Uddin Mahmud, 2003). Thus we can say that matraline consists of matra upper line and matra bottom line. Fig. 7.2.1 shows the detection of matarline for different images, (a) and (b).

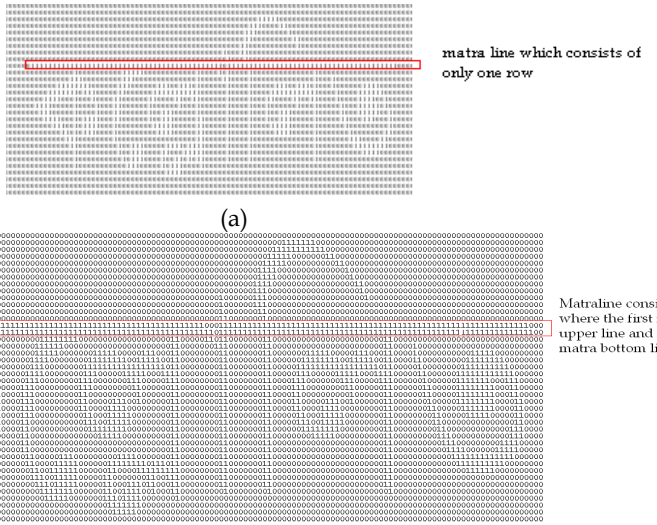


Fig. 7.2.1. Matraline detection process for different mages (a) - (b)

7.3 Baseline Detection Process

A line has a baseline, also named as an imaginary line, which is a row from where the middle zone ends and lower zone starts i.e. a separator between middle and lower zone. Baseline gets equal to end row of the line when the line does not have any lower modifier(s). It is the row where an abrupt change occurs between the previous and next row (S.M. Milky Mahmud, 2004). In a general document, it is observed that about 70% lines hold baseline i.e., 30% lines do not have lower modifiers. So detection of baseline is very important for bangla. Some characters particularly some modifiers are situated here and they are needed to be recognized. Baseline can be detected efficiently searching from lower position of the middle zone to the end row of a line. This is to find out the position from where the black pixels start to increase while they are decreasing in the middle zone. That position is denoted as baseline (Nasreen Akter, 2008). Fig. 7.3.1 shows the baseline.

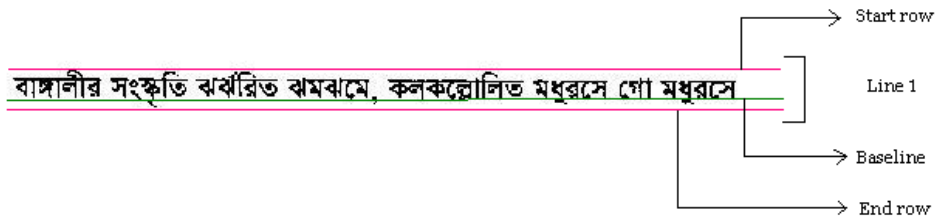


Fig. 7.3.1. Process of baseline detection

7.4 Word Segmentation Process

There are always some white spaces between two words in a text line. Using vertical scan, words are separated by treating the white spaces as a separator. Fig. 7.4.1 shows the word segmentation process.

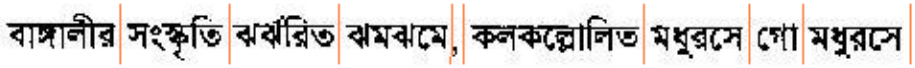


Fig. 7.4.1. Process of word segmentation

During the word segmentation process, peculiar situations occur when some matraless character is used in a word (fig. 7.4.2). These situations create some false separators which cause the word to be broken into small pieces. So the process of word segmentation becomes faulty. To avoid this error, widths of each separator in a line is calculated and an average of them is found. Separators which have widths greater than or equal to the half of the average are considered to be true separator (Nasreen Akter, 2008). (B.B. Chaudhuri, 1998) used the midpoint of a run of at least k_1 consecutive 0s (i. e., white pixels) if the run exits in a vertical projection profile of a line.

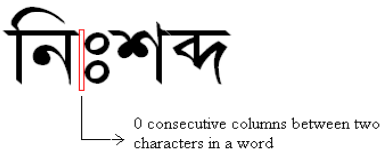


Fig. 7.4.2. Word with 0 consecutive columns between two characters

7.5 Character Segmentation Process

It is the most difficult and challenging part to build printed Bangla OCR. As Bangla is an inflectional language, the ornamentation of the characters in a word causes many peculiarities and makes the segmentation difficult. Many works have been done in order to solve those problems. A word can be constructed either with only taking the basic characters or basic characters and modifiers or basic and compound characters or basic, modified and compound characters. The main parts of all the characters are situated in the middle zone. So the middle zone area is considered as the character segmentation portion. Since matraline connects the characters together to form a word, it is ignored during the character segmentation process to get them topologically disconnected (B.B. Chaudhuri, 1998).

A word constructed with basic characters is segmented into characters in a way by scanning vertically, starting from just beneath the lower row of the matraline to the baseline, considering a column of continuous white pixels as the separator, shown in fig. 7.5.1, between the characters (B.B. Chaudhuri, 1998, Jalal Uddin Mahmud, 2003, Md. Abdus Sattar, 2007, Md. Al Mehedi Hasan, 2005, Nasreen Akter, 2008, S.M. Milky Mahmud, 2004). In this technique, the two characters, গ and গ, get split into two pieces due to ignoring matraline. This problem is overcome by joining the left piece to the right one to make an individual character by considering the fact that a character in the middle zone always touches the baseline (B.B. Chaudhuri, 1998).

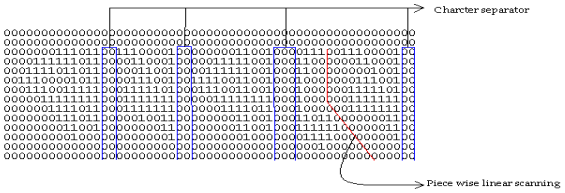


Fig. 7.5.1. Character segmentation ignoring matraline

There are four kinds of modifiers based on their uses. One kind of modifiers, used only in the middle zone, is called middle zone modifiers, for example া, ী, ে and the modifiers which are used only in the lower zone, such as, ্ক, ্ত, ্র - are called the lower zone modifiers. Another kind of modifiers is there which consists of both upper and middle zone. They are like ি, ো, ৌ, ৈ. The last kind of modifier is called the upper zone modifier, such as ঁ.

The four modifiers, ি, ো, ৌ, ৈ - have middle part as well as upper part. Some basic characters are there which also have upper portion of their own, for example, ই ঈ উ ঊ ঢ ঠ এ ঐ. The portions of these characters are found by a set of characteristic functions Md. Al Mehedi Hasan, 2005) or initiating a greedy search from a pixel in order to find a whole character (Jalal Uddin Mahmud, 2003). When the above modifiers, especially ি, ো, ৌ - are used with basic characters, many combinations like কি, কী, কি, কি, কী, কী, কো, কৌ, কৌ. In case of কি, কী, কো - these three modifiers have been used with those basic characters that have no upper portion of their own. As there is a separator, shown in fig. 7.5.2, in the middle zone for each of them, the thickness of the left and right portion of the separator determines which portion is for the modifier. A horizontal scan applying in immediate upper row of the matra upper line from left to right or right to left determines that the basic character does not have the upper part of its own (as because a row of white pixels is found from the scan). After that, the characters, কি, কী, কো - are got segmented keeping their original shapes (Nasreen Akter, 2008). In the time of কি, কি, কী, কী, কৌ, কৌ, a row of white pixels is not found during the horizontal scan. In কি and কি, the first found black pixel determines whether there is ি or ঁ and then কি is segmented to ি and ি in the way shown in fig. 7.5.2 and কি is segmented by calculating the gap in the upper zone and the thickness of the upper part of the basic character (Nasreen Akter, 2008).

In order to segment টা and টী, more than one gaps are found in the immediate upper row of the matra upper line for টী which determines there is ঈ. Then টা is segmented into characters by finding the cutting point in the first and second half of the upper zone and টী is by the thickness column (Nasreen Akter, 2008). Although টী has one gap, it does not get mixed up with টা since no cutting point is found in the first half of the upper zone. The segmentation process of টী is similar with the segmentation process of ফি only differing with the scan direction (Nasreen Akter, 2008). টা is segmented by finding the cutting point (Nasreen Akter, 2008).

More peculiarities are formed if the modifer ◌ঁ or ◌ঃ gets included with those situations. These issues have not been addressed yet by researches. However, modifier, ◌ঁ with basic character, which is used in the upper zone, is detected by considering its distinct feature which is, it always makes a regular angle with matra (Md. Abdus Sattar , 2007).

Three other modifiers are there, ◌ু, ◌ূ, ◌ৃ, which are used in the lower zone. They are extracted by doing DFS (Depth first search) below the baseline ((Jalal Uddin Mahmud, 2003) or finding a column of white pixels after the first black pixel is found (Nasreen Akter, 2008). Sometimes a small part of a middle zone character exceeds the baseline. They are distinguished from lower zone modifiers since they don't touch the end row of the line as the characters in the lower zone always touch that row (Nasreen Akter, 2008).

Rests of the modifiers are used in the middle zone and both of them are segmented as the regular way. However, sometimes basic character and modifier or two basic characters or two modifiers get into each others' region. If the middle zone modifiers and basic characters get into each others' region, a piecewise linear scanning is applied on them to segment the characters (B.B. Chaudhuri, 1998, Jalal Uddin Mahmud, 2003). Above all, during the character segmentation time, sometimes group of characters do not get segmented as they enter each others' region and seems to act like a character. A piece wise linear scanning, shown in fig. 7.5.1, is done to segment those characters (B.B. Chaudhuri, 1998).

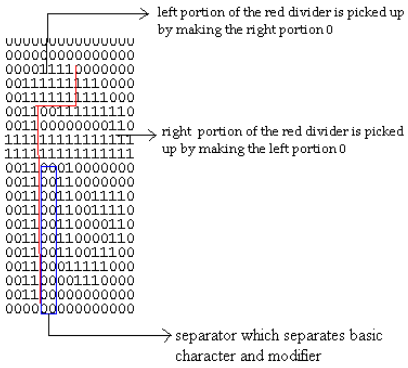


Fig. 7.5.2. Segmentation process of ফি

8. Feature Extraction

Feature extraction is an important and challenging part for any character recognition process. Selection of good features leads to improved recognition rate. Many feature extraction algorithms are proposed for both printed and handwritten bangla document. Some of the potential algorithms are described below with comparative analysis.

8.1 Feature extraction algorithm for printed bangla character

(a) (Jalal Uddin Mahmud, 2003) proposed the following procedure to extract feature from bangla text. In Bangla language, more than one connected components are present. Here, at first, all the connected components of an isolated character are detected using DFS (Depth First Search). Then center of mass has been calculated for each connected component. Center of mass for i th connected component is (X_i,Y_i). Where

$$X_i = \sum_{j=1}^{N_i} P_{ij} / N_i \tag{1}$$

$$Y_i = \sum_{j=1}^{N_i} Q_{ij} / N_i \tag{2}$$

Here,
N_i = Number of Black pixels in connected component i.
P_{ij} = x Coordinate of the j^{th} Black pixel in i^{th} connected component.
Q_{ij} = y Coordinate of the j^{th} Black pixcl in i^{th} connected component.

Then a bounded rectangle of each component is calculated by its minimum and maximum span in x direction and y direction. The freeman chain code is calculated by dividing each component into four regions depending on the center of mass of that component shown in fig. 8.1.1.

Fig. 8.1.1. Four Regions for a connected component

Freeman Chain code is based on the observation that each pixel has eight neighborhood pixels. The 8 transitional positions defined by freeman chain code are then divided into 4 transitional zones in order to keep the correct order of searching. Fig. 8.1.2 describes the freeman chain code.

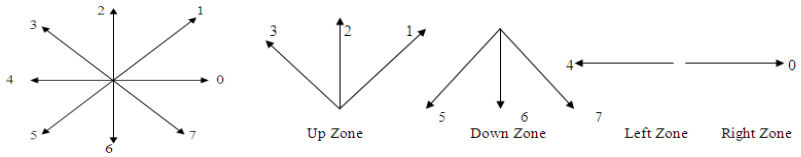
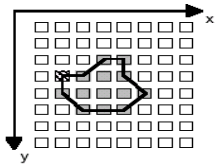


Fig. 8.1.2. (a) Slope Convention for Freeman Chain code, (b) 8 directional slopes divided into 4 direction zones for searching.

Maintaining an anti clock wise order of searching, zonal information is used to modify the chain coded position of the next selected pixel. The algorithm selects the next pixel if it fulfils all of the following criteria:

- The pixel is Black, i.e., it is a part of the character.
- The pixel is within the bounded rectangle of the connected component.
- The pixel is still not visited.
- The pixel is in a zone.



(a)

Zone	Searching Order
DOWN	3,4,5,6,7,0,1,2
RIGHT	5,6,7,0,1,2,3,4
UP	7,0,1,2,3,4,5,6
LEFT	1,2,3,4,5,6,7,0

(b)

Fig. 8.1.3. (a) Chain code generation for an image, (b) Searching order in the four zones

Fig. 8.1.3 (a) shows the chain code generation of an image marked by gray pixels. When the algorithm starts from the hatched pixel (absolute coordinate, x=1, y=3), it marks the current black pixel as visited and initiates its directional zone as DOWN zone. So it searches for an unvisited black pixel in the directional order: 3,4,5,6,7,0,1,2 (Searching order is shown in Fig. 8.1.3 (b) for each zone). In this way the process continues and finally produces the chain code, 06700132454.

The frequency of each directional slope at each region is recorded and updated during the traverse. A total of 32 directional slopes or local features for each component are found. Then they are normalized to 0-1 scale. In Bangla, as there are more than one components in a character, the normalized features for each connected components are then averaged. The calculation of normalized slope distribution is as follows:

If a_1, a_2, a_3, ……,a_8 are 8 directional slopes in region 1, then normalizing constant for region 1 is,

$$N_1 = \sqrt{(a_1*a_1 + a_2*a_2 + \ldots\ldots\ldots + a_8*a_8)} \quad (3)$$

So, normalized slope in region i = $S_{i,j}$/ N_i , where i = 1 to 4 and j = 1 to 8

$S_{i,j}$ = Frequency of j^{th} directional slope in i^{th} region.

N_i = Normalizing constant in i^{th} region.

(b) (Abu Sayeed Md. Sohail, 2005) demonstrated a method where a large two dimensional vector is converted into a small one dimensional vector. Here, initially, character images of fixed resolution are separated from original text and digitized into a large two dimensional vector, later converted into a small one dimensional vector. The algorithm they used is:

i. Find center of the image by Center_x = (width)/2, Center_y = (Height + 1)/2
ii. For each black pixel, a radius is calculated as $r = \sqrt{(Center\_x - x)^2 + (Center\_y - y)^2}$
iii. Sum all pixels within same radius (sum_r_max,……., sum_r_min)
iv. Calculate the membership for each circle or disk using the membership function, $m\_f(bp) = e^{-(|bp - rbp)/total\_pixels}$

This technique is used for the recognition of a single isolated character. It has not been tested for all the characters in a document or at least in a word.
The technique has been used with ANN (Artificial Neural Network). The advantage of this technique, shown by the authors, is that the extracted features are same even if the original character is rotated with rotation angles from 5 to 350 degree having a little bit of distortion. Non rotated characters are recognized with more than 90% of efficiency while rotated characters are recognized over 80- 89% efficiency depending upon the angle of rotation.

(c) (B.B. Chaudhuri, 1998) showed that compound characters occupy only 4-6% of the text corpus. In order to introduce fast, accurate and robust technique, basic, modifier and compound characters are distinguished and identified by the following 3 features listed below.

i. Feature f_1 - Bounding box width.
ii. Feature f_2 - Number of border pixels per unit width, which is computed by dividing the total number of character border pixels by the width of the character.
iii. Feature f_3 - Accumulated curvature per unit width.

They have used a feature-based approach for basic and modifier character recognition and a combination of feature-based and template-matching approach for the compound character recognition.
The authors considered a few stroke features for initial classification of the basic characters by a tree classifier. Apart from them, some other features are also used at some nodes of the tree classifier. In fig. 8.1.4, the principal set of chosen stroke features is shown.

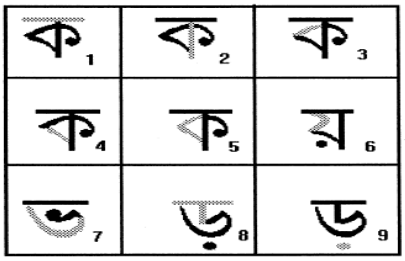


Fig. 8.1.4 Stroke features used for character recognition. (Shaded portions in the character represent the features)

It is mentioned that most of the cases, the strokes, 1, 2, 3, 4, 5 and 8 are correctly detected.

8.2 Feature extraction algorithm for Bangla handwritten characters

(a) (Subhadip Basu, 2005) has used a set of 76 features which includes 24 shadow features, 16 centriod features and 36 longest- run features, computed taking 64 x 64 pixel size binary images.

Shadow features are calculated by dividing the image into 8 octants within minimal square. Lengths of all projections on each of the 24 sides of all octants are summed up to produce 24 shadow features, shown in fig. 8.2.1.

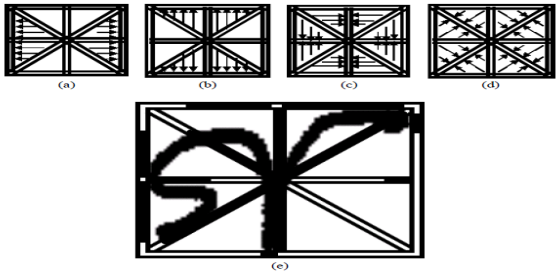


Fig. 8.2.1. An illustration for shadow features. (a-d) Direction of fictitious light rays as assumed for taking the projection of an image segment on each side of all octants. (e) Projection of a sample image

Coordinates of centriods of black pixels in all 8 octants of a digitized image, shown in fig. 8.2.2, are considered to add 16 centriod features.

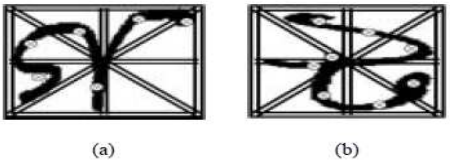


Fig. 8.2.2. Centriod features of two different characters (a)-(b)

Longest- run features are computed dividing the square into 9 overlapping regions and for each, 4 longest-run features are calculated respectively by row wise, column wise along 2 of its diagonal. Thus 36 features are produced (fig. 8.2.3).