



Department of Computer Science and Engineering (CSE)  
Islamic University of Technology (IUT)

## **An End to End System for Online Handwritten Bangla Character Recognition**

Authors

**Sahriar Nur Nahin (170041049)**

**Kazi Fahim Imam (170041058)**

**Nabil Rahman (170041030)**

**Anika Tasnim (170041001)**

**Supervisor:**

A.B.M. Ashikur Rahman  
Assistant Professor,  
Department of CSE

**Co-Supervisor:**

Shahriar Ivan  
Lecturer,  
Department of CSE

**A thesis submitted to the Department of CSE  
in partial fulfillment of the requirements for the degree of B.Sc.**

**Engineering in CSE**

**Academic Year: 2020-21**

**April, 2022**

## Declaration of Authorship

This is to certify that the work presented in this research work is the outcome of the experiments and analysis based on performed by Sahriar Nur Nahin, Kazi Fahim Imam, Nabil Rahman and Anika Tasnim under the supervision of Assistant Professor A.B.M. Ashikur Rahman of Department of Computer Science and Engineering (CSE), Islamic University of Technology (IUT), Dhaka, Bangladesh. It is also declared that neither of this thesis nor any part of this thesis has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

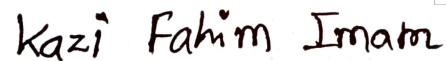
### Authors:



---

Sahriar Nur Nahin

ID : 170041049



---

Kazi Fahim Imam

ID : 170041058



---

Nabil Rahman

ID : 170041030



---

Anika Tasnim

ID : 170041001

***Co-supervisor:***



---

Shahriar Ivan

Lecturer

Department of Computer Science and Engineering

Islamic University of Technology (IUT)

***Supervisor:***



10.05.2022

---

A.B.M. Ashikur Rahman

Assistant Professor

Department of Computer Science and Engineering

Islamic University of Technology (IUT)

# Acknowledgement

We deem it a pleasure to acknowledge our sense of gratitude to our Supervisor A.B.M. Ashikur Rahman, Co-supervisor Shariar Ivan and teachers from CVLab under whom we have carried out the research work. Their incisive and objective guidance and timely advice encouraged us with constant flow of energy and motivation to continue the work throughout.

We shall remain grateful to our teachers from CVLab and all related courses specially, for providing us a strong academic atmosphere by their disciplined teaching techniques to teach us the basics without which we would not have been able to complete the work.

# Abstract

This report summarizes the attempt to find the way towards building an Optical Character Recognition System for handwritten Bangla characters. The complex and unique structure of scripts like Bangla and ever challenging nature of handwritten texts combined makes it really difficult to complete a perfect system to approach to convert the scanned handwritten Bangla scripts to machine editable digital counterpart format of it- as segmentation of the whole image into characters and then classification of the segmented characters is difficult enough to make the task challenging. In our work, we propose to approach the segmentation process (directly segment to words) with Distance Transform and morphological operations for error correction later. Then two zone approach (either side of matra- upper and lower zone) and apply connected component analysis on both zones. We handled or adjusted the failed and not directly successful cases by experimenting with the characteristics of handwritten characters. Then for classification process, we proposed to classify the segmented characters using neural networks trained on the relatively newly available datasets. Multiple column, Mixed characters (Bangla- other languages) and Scene Text Recognition is out of the scope of our study so far. And we could not include the post-processing part for our work for lack of work or mention in existing literature, which might be a great addition in the way of building a complete OCR system.

**Key Words:** Bangla Handwritten Character Recognition, Handwritten Document Recognition, Optical character recognition

# Contents

<b>1</b>	<b>INTRODUCTION</b>	<b>7</b>
1.1	Overview . . . . .	7
1.2	Problem Statement . . . . .	8
1.3	Motivation and Scopes . . . . .	9
1.4	Research Challenges . . . . .	9
1.5	Outline . . . . .	10
<b>2</b>	<b>BACKGROUND STUDY</b>	<b>11</b>
2.1	Characteristics of Bangla Character . . . . .	11
2.2	Some Necessary Definitions . . . . .	13
2.2.1	Zone Information . . . . .	13
2.2.2	Horizontalness and Verticalness . . . . .	14
2.3	Literature Review . . . . .	15
<b>3</b>	<b>Proposed Approach</b>	<b>18</b>
3.1	Overview of Proposed Approach . . . . .	18
3.2	Pre-processing Steps . . . . .	20
3.2.1	Binarization . . . . .	20
3.2.2	Noise Detection and Reduction . . . . .	20
3.2.3	Skew Detection and Correction . . . . .	21
3.3	Segmentation . . . . .	22
3.3.1	Line-Word Segmentation . . . . .	22
3.3.2	Skew Correction (Word Image) . . . . .	23

3.3.3	Character Segmentation . . . . .	25
3.4	Classification . . . . .	31
<b>4</b>	<b>EXPERIMENTATION RESULT AND ANALYSIS</b>	<b>32</b>
4.1	Datasets . . . . .	32
4.2	Our Experiments . . . . .	34
4.2.1	Segmentation . . . . .	34
4.2.2	Classification . . . . .	34
<b>5</b>	<b>FUTURE WORKS AND CONCLUSION</b>	<b>37</b>

# List of Figures

1.1	Characteristics of Bengali word[2]	10
1.2	Bangla Compound Characters [4]	10
2.1	Zone Information	13
2.2	Horizontalness and Verticalness	14
2.3	Projection Profile Method (Line and word segmentation)	16
2.4	Water Reservoir to Skew Correct[6]	17
3.1	Our proposed approach to build OCR system	19
3.2	Active contour Skew Correction	21
3.3	Original Image (Left), EDT of the image (Middle), Binarized Inverted EDT output-after CCL (right)	23
3.4	Connected Component Labeling on EDT image (Top-Left), after applying morphological operations (Top-Right), Output Image (Bottom)	23
3.5	Issue with Active Contour- Original (Top) and Rotated (Bottom)	24
3.6	Junction Points	24
3.7	Potential Matra Region (Left), Points for regression line (Middle), One of the Results (Right)	25
3.8	Issue with three zone approach	25
3.9	Matra removal process (Many pictures are inverted from impleme- ntation, for presentation purpose)	27
3.10	Connected component analysis on matra separated image, Step-06	28
3.11	Matra added single characters on original image	28



3.12 Lower Zone Segmentation . . . . .	29
3.13 Issue Fix 01 (left error, right Adjusted) . . . . .	30
3.14 Issue Fix 02 (left error, right Adjusted) . . . . .	30
3.15 Issue Fix 03 (left error, right Adjusted) . . . . .	30

# List of Tables

4.1	Classification Summary . . . . .	35
-----	----------------------------------	----

# Chapter 1

## INTRODUCTION

### 1.1 Overview

With the digitalization of every field, Optical Character Recognition (OCR) has emerged as a major research and necessary field for the use to convert huge volume of books and other handwritten, typewritten or printed text documents (scanned format or clicked images) into digital machine editable counterpart of it. As the fifth language in terms of native speakers (sixth in terms of total number of speakers), there is no further need to mention the usefulness for Bangla OCR. However, starting from 80's- till today there is no such complete OCR system for Bangla handwritten characters.

At start, authors depended on methods like structural analysis, template matching techniques, pattern recognition, structural feature extraction [1] etc. Which started to move toward neural networks very soon. For segmentation, earlier contributors used projection profile method (horizontal histogram for line segmentation and vertical for word-character segmentation with slight adjustments) [1], [2].

OCR system for printed text achieved quite some success in past years, which enforced the focus into handwritten Bangla Characters- such as the inclusion of massive datasets for both basic and compound characters[3], [4] as well as word

page documents[5]. Before that, research was building toward (starting from projection profile method [1], [2] toward use of advanced stroke properties and deployment based methods[6], [7], [8]) segmenting the document text image into basic characters and modifiers to send for classification stage. Also there was no large enough page image dataset (only a version was CMATERdb[9] was available with small sample size) to compare the methods from various authors. The inspiring contribution for datasets [3][4][5] in the past few years kind of allowed us to look for a comparatively lesser complex method than the ones of recent years for segmentation to put into test again as we get much more classes to work with with more number of images each class, compared to previous of maximum classes for 84 characters. Though most methods [1], [2], [10], [11] won't work for handwritten document images - for the unconstrained nature of it and varying style of writing of different writers combined with the unique characteristics and complex shape of word and characters in Bangla language, we need to find some concrete method at the end of our work to segment the document image into characters, simple or compound. For classification, we propose to use one of the deep learning models available [12], [13]- for the high accuracy over other methods[14], there will be an attempt to make some improvement in the architecture- either in accuracy or in performance cost.

## 1.2 Problem Statement

The goal is to develop a character recognition system that can accurately segment recognizable characters from handwritten document images after some necessary pre-processing steps and then recognize the segmented characters and combine them into machine editable version of the input document. Ideally we should have a clear segmenting approach and near perfect classification model trained. But even then, we will need a post-processing system to error-correct in some way and combine segmented characters back into a single machine editable document.

## 1.3 Motivation and Scopes

Existing accurate OCR systems mostly were used on printed characters until a few years back. There were a few attempts, but but none was perfect enough for regular or standard use. In recent years, we have seen some massive and inspiring focus in the Bangla handwritten character recognition both in terms of datasets and models/architecture to classify the individual characters, which is really inspiring for this field of study. We finally have big enough dedicated datasets for entire document image and more classes than ever to work with for segmentation tasks. Here in this work, we try to find out a way to segment the characters from any given document image keeping the addition of new datasets in mind. In other words, not to segment the compound characters further that has separate class in the dataset, allowing a simpler method for character segmentation. And then classify the segmented characters using a neural network model. Multiple column, Mixed characters (Bangla- other languages) and Scene Text Recognition is out of the scope of our study so far. But the methods can be used for Devanagari characters with slight adjustments- for the similar nature of the two scripts and others like them- as they need some different type of study or processing on top of the work done by us. So the work is expandable in the domain but the study so far does not directly cover them.

## 1.4 Research Challenges

The biggest challenge in our work apart from technical part has been lack of end to end handwritten character system for Bangla characters. They usually divide the process into two parts- segmentation and classification. So we did not find standardized approach to do so. And the non-existence of post-processing system for document image character meant the task of error handling and combining the segmented characters is harder.

The next challenge has been the existence of Matra, that topologically con-

nects characters- adding to already complex and varying nature of handwritten characters. And the nature of Bangla characters- the complex and multiple shapes or version of same character. with multiple characters joining to form

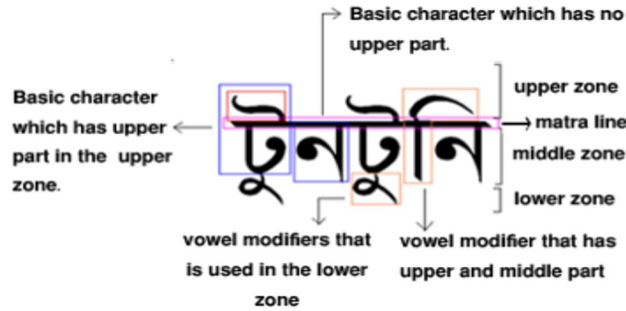


Figure 1.1: Characteristics of Bengali word[2]

compound characters. As well as existence of modifiers for both consonant and vowels. In total we have over 300 possible combination of compound and basic



Figure 1.2: Bangla Compound Characters [4]

characters. So even if we get all the classes necessary- training and classifying for this many characters will always remain challenging.

## 1.5 Outline

We listed our background study of previous literature at **Chapter 2** as well as includes some characteristics of Bangla characters and Necessary definitions for our proposed approach so far, discussed at **Chapter 3**. The results so far and the problems are discussed in **Chapter 4** and the conclusion part completed the report at **Chapter 5**. And this chapter tried to give a overview of the entire process.

# Chapter 2

## BACKGROUND STUDY

### 2.1 Characteristics of Bangla Character

Bangla language uses Bangla-Assamese alphabet, used for Bengali, Assamese, and Sanskrit etc. It also has quite some similarities with Devanagari characters (third most used script in the world), but Bangla Script is less blocky and more sinuous shape than Devanagari characters. Bangla is written horizontally from left to right using a single letter case. Bengali writing system is Abugida, meaning vowel graphemes mainly realized not as independent letters, but the diacritics modifying vowel inherent. Bangla alphabet consists of **11 vowel** (shoroborno), **39 consonant** (Benjonborno) and **10 numerals**. Matra is a unique feature for scripts like Bangla and Devanagari as well as the other Bhramic scripts have- a visible left to right head-stroke. The distinctive horizontal line, runs along top of letters linking most of the characters together.

There exists modifiers or short form for both vowel and consonants known as **Kar** (কার) and **Fola** (ফলা). And different and varying shapes in the alphabet really makes the Bangla script unique. We can have a huge number and variation of compound characters, over 250 of them. Combining consonant-consonant, consonant-modifiers (vowel/consonant), multiple consonants (upto 3-4), compound-modifiers (vowel/ consonant) etc. But the size should ideally

be same (for printed characters especially), regardless of the complexity of the compound character. That helps in both identification and addition of vowel diacritics

The biggest challenge of building optical character recognition system for bangla characters is the existence of Matra (Figure:1.4 and Figure: 2.1), the line topologically connecting the characters of a single word- unlike most other language. And handwritten characters are by nature challenging- for varying human style even across the same document by same writer. The fact doesn't help that Bangla has some characters with similar representation (ঐ-ঔ, ঞ, ঞ) same character with different representations, in the pure form and also in the form of modifiers- behaving different than original form of it. Then comes the complex shapes of Bangla characters and existence of compound characters and the possible combinations cross 300. So segmenting is never going to be easy with such wide variety. And the last one is not from literature but our experiments, the un-archiving the datasets from colab gave us quite a few missing files while classification- which became a real big challenge to overcome.



## 2.2 Some Necessary Definitions

### 2.2.1 Zone Information

Bangla Characters are divided into three zones as mentioned in most of the existing literature- Upper, Lower and Middle. Upper zone consists of the extension of some characters and modifiers, Lower zone consists only of modifiers- sometimes with a direct connection to the characters of middle zone (কু/বু) and sometimes totally separated (the extension dot of (ঝ/ড়/ঢ়) based on font or writing style and characteristics. Not all words or character is segmented into upper and lower zone (30% of the characters don't have a lower zone[2]). Middle zone consists of characters (sometimes part of it) and modifiers or part of them.

Upper zone is defined from the top of the character (R1) till Matra line (R2)- with part of modifier or character, and Lower zone is defined from an imaginary base line (at the end of characters (R4)/start of modifiers) till the end of the word/character (R5). The midpoint between R2 and R4 is considered as R3. Middle zone starts from Matra line (R2) to the bottom base line (R4). The traditional Middle zone (R2 to R4) consists of mostly characters and modifiers. And the lower zone (R4 to R5) consists of modifiers and extension dots. Figure: 2.1 shows different lines described above.

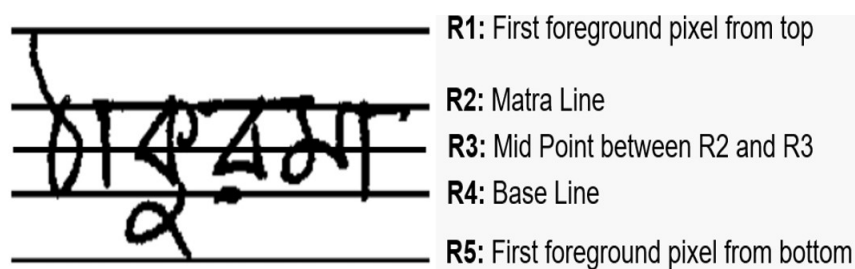


Figure 2.1: Zone Information

## 2.2.2 Horizontalness and Verticalness

Two important features we used in our study was horizontalness and verticalness features of Bangla Characters [8]. Horizontalness is the horizontal longest run of components exceeding the mean. This horizontalness property of the Matra (and some continuous horizontal stroke) may be extracted from the row wise sum of continuous run of black pixels. Later this value is normalized with respect to the maximum longest run value of any pixel within the word image.

Verticalness is the vertical longest run of components exceeding the mean. Many characters and modified shapes in Bangla script have vertical stripe of black pixels, as a part of their shapes. This vertical stripe often appears at the right side, middle or left side of the characters. These stripes touch the Matra of a word image and often extend till the bottom of the respective characters or modified shapes



Figure 2.2: Horizontalness and Verticalness

## 2.3 Literature Review

The OCR involving printed Bangla scripts has been addressed in many of the research papers, starting from 80's. There exists quite a few a number of work has been investigated for isolated handwritten character and digit recognition in Indian script, only a few pieces of work exist towards handwritten end to end OCR system in Bangla Scripts. A mentionable reason behind that was unavailability of datasets with compound characters. The first benchmark dataset for Bangla offline handwritten characters was CMATERdb [9], with only basic characters and also the size of the dataset is always been a concern. Later BanglaLekha isolated[15] came into scene as a larger dataset adding the modifiers, but still no (total of 84- Basic characters- numerals and modifiers) was also an issue. This is one of the reason segmentation [6]-[8], [16]-[18] and classification [12]-[14], [19] was been researched separately. As segmentation process had to be built toward more and more complex methods to further segment the more complex compound characters into basic form. Ekush [3] and Matrivasha [4] were two of the biggest addition for the amount of compound characters classes they have. And also the inclusion of banglaWriting [5] document page image dataset was another milestone- allowing authors to compare and check their work with a large enough document image dataset.

For binarization- a part of preprocessing, we have various methods available[2], of them Otsu's method [20] is simpler and works well enough. Then some noise removal and other steps follow the process [1], [2]. Then the document image needs to be skew corrected, Active contour model [21][22] and Histogram Rotation method [22] both are effective enough to operate on. For Histogram Rotation method, Projection profile is employed along different orientation angles to determine the local orientation. The average difference between peaks and valleys is performed to find the skew angle (looks for sharper peaks and craved valley to find the best fit).

For line segmentation in handwritten characters various methods are used or adopted. Starting with the massively successful method for printed characters- projection profile (Figure: 2.3) method [1], [2]- to Hough transform [10], Viterbi algorithm [11]. For the intrinsic complex nature of handwritten documents, we often don't have a clearer distinctive boundary between lines, also they aren't always straight lines. For non-uniformly spaced and presence of overlapping and touching words, a safer choice is to go for line and word segmentation directly using Euclidian Distance Transform and some error handling [17].



Figure 2.3: Projection Profile Method (Line and word segmentation)

Euclidian Distance Transform (EDT) maps each pixel (divided into Object and Background pixel) into its smallest distance to regions of interest. DT labels each object pixel of the image with distance between that pixel and the nearest background pixel. Then they binarize the EDT of input image (dynamically selected threshold) Apply Connected Component Labeling (CCL).

Most of the research works didn't mention skew or slant correction methods for word images. [6] Used water reservoir principle to correct skew (Figure 2.4)- a method usually used for segmentation. Using bottom reservoir property, they pick the some points on the matra region to draw a regression line. The angle of the regression line with the horizontal axis is the skew angle (rotate it so that that is parallel to the horizontal axis).

There's various methods to segment the characters from word images- from Basic projection profile method [1], [2], water reservoir [18], connected component

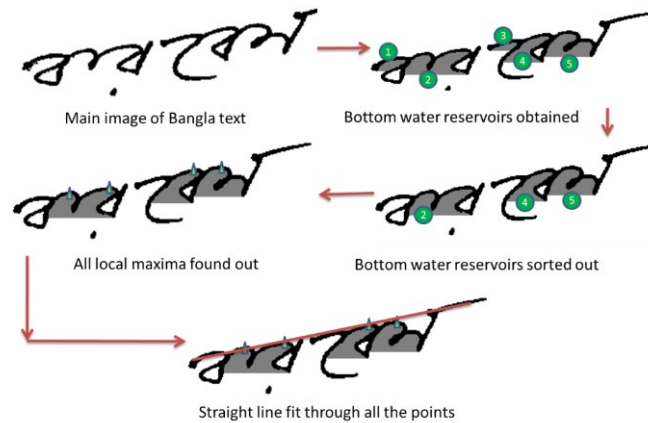


Figure 2.4: Water Reservoir to Skew Correct[6]

analysis [23], various properties of the script [7], [8], [16] to deployment based methods [6], [8]. The methods were built toward more advanced ones later ones able to segment more and more complex character into basic ones.

At start, authors depended on methods like structural analysis, template matching techniques, pattern recognition, structural feature extraction [1] etc. But at recent times, neural network models [12], [13], [19] are pretty much the standard for the higher accuracy it provides than other methods still in discussion [14]. We don't have much work or mention regarding post-processing of character recognition system- for Spell checking, error checking, text editing etc.[2]

# Chapter 3

## Proposed Approach

### 3.1 Overview of Proposed Approach

In our work, we propose a method combining the relevant literature from the ones mentioned above. The Details are discussed in this section. Despite a lot of standardization for the complex nature of Bangla scripts and measures taken for printed characters to avoid confusion and simplify them in the process- the handwritten counterpart doesn't have much of advantage. From the unclear and sometimes non-linear shape of matra, characters touching each other for unconstrained nature of it and also the unavailability of a clear bottom base line adds up to the already complex nature of Bangla characters- to make the segmentation process extremely challenging. We can divide the whole process into four big segments:

A. Pre-processing: Step 01 and 02 from our proposed approach combine for this part.

B. Segmentation: That is the biggest part of our process, combining from step 03 till step 09 (Line-word-Character all three are covered in here).

C. Classification : Recognize the segmented characters using a trained classification model.

D. Post-processing: Error Handling- and recombining the identified characters into a document as machine editable text.

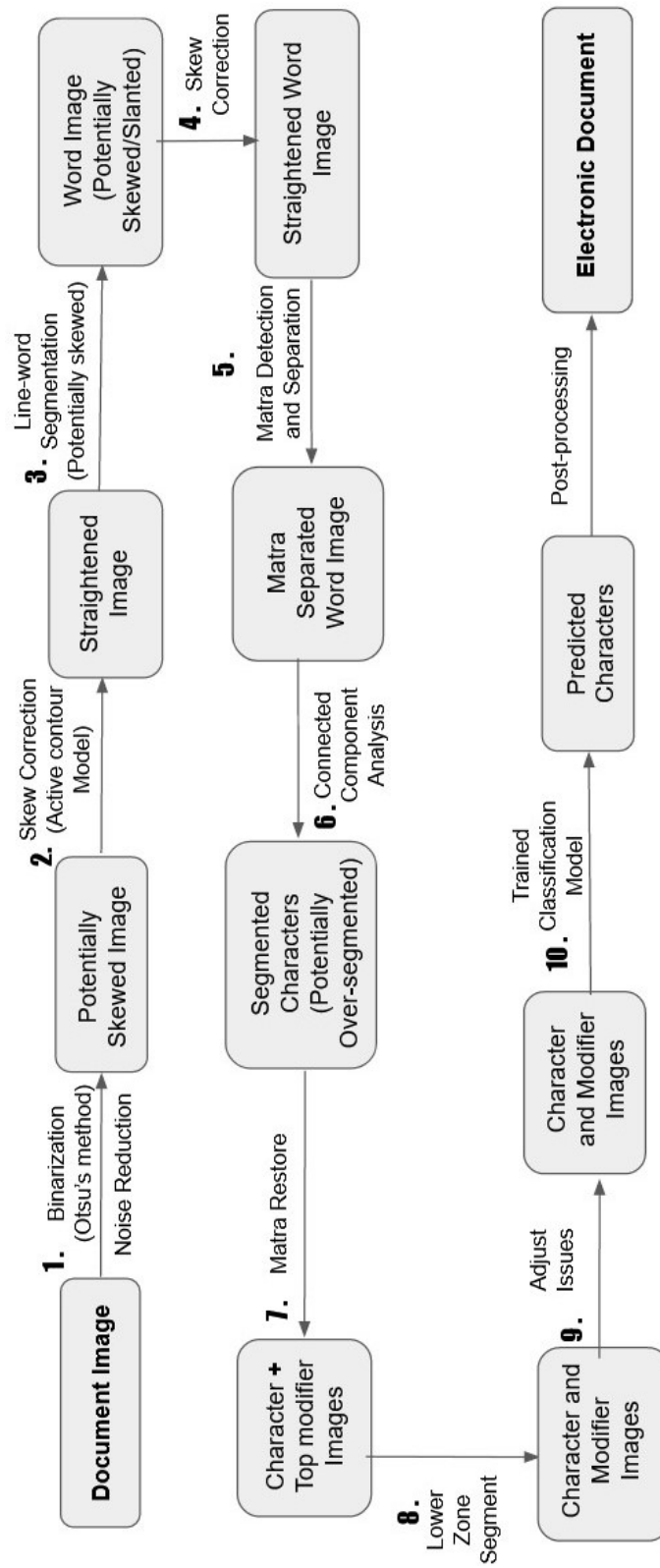


Figure 3.1: Our proposed approach to build OCR system

## 3.2 Pre-processing Steps

Pre-processing is one of the important stages to set the base for other characters to set the stage for methods to be applied next. In this step we try to handle the errors we might face for noise and other factors that make non-ideal situation for the segmentation and classification steps to be applied later

### 3.2.1 Binarization

After scanning the image, it is an important part is to convert the gray-scale image to binary, in order to make the processing simpler and sometimes a bit of noise removal and more clear picture. We propose to use Otsu's method for this step. This simple and yet effective approach divides the image into two classes of pixels (background and foreground) and calculates the optimum threshold separating those two classes so that their combined spread and mismatch (intra-class variation) is minimal [20]

### 3.2.2 Noise Detection and Reduction

Noise removal includes removal of single pixel (or multiple that are not part of the original characters) component and removal of stair case effect after scaling. The single page script datasets we used, CMATERdb [9] and Banglawriting [5] had no additional noise for in image- so we couldn't run a very good experiment in this step. But we applied Gaussian filter – in order to deal with the few images had resolution issue and deal with stair case effect. And a median filter to remove the unintentional points at the script (It should be mentioned that the size of the median filter is very crucial, as we have some characters like র/ড়/ঢ় with extension dots as the part of the character- any big sized filter might treat the dots as noise).



### 3.2.3 Skew Detection and Correction

Skew Correction was done in two stages, on the whole document image and then on the word image, after word segmentation is done. We took two different approaches at two different stages.

For the entire document image we took on active contour model [21][22] (Figure3.2)

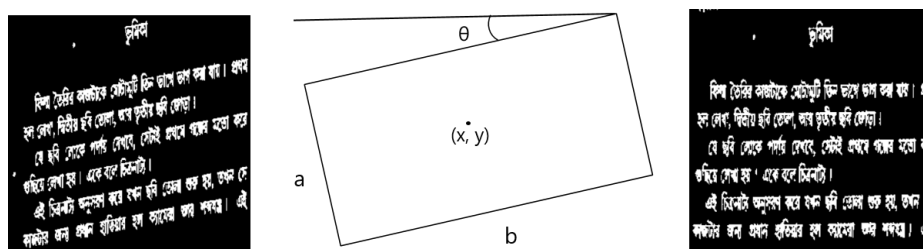


Figure 3.2: Active contour Skew Correction

We have to find the rectangular area bounding the active text region in the whole document image. That area is going to provide the approximately the angle in which the image texts are skewed. We keep the center region  $(x, y)$  intact while rotating the image around the angle.

The issue with skew correction (and adjust the effect of slant) of segmented words with this approach is the top zone region at a side (toward leftmost or rightmost) of the word. That creates an issue while taking the active contour area- so we switch to another method for that step. [22] Mentioned histogram rotation method, Projection profile along different orientation angles is used to determine the local orientation. The average difference between peaks and valleys is performed to find the skew angle (look for sharper peaks and craved valley, which gives higher scores).

## 3.3 Segmentation

Segmentation is the important stage of the whole process of on line character recognition process- as the classification accuracy heavily depends on the segmentation process. Traditionally from whole page to character segmentation process divided into three parts: Line segmentation, Word segmentation and finally Character segmentation.

### 3.3.1 Line-Word Segmentation

Because of the intrinsic complex nature of handwritten documents, we often cannot find any clear enough distinctive boundary between lines- especially if it has congested writing style. Also even if we find a boundary, the boundary lines aren't always straight. For presence of overlapping and touching words across lines and non-uniformly spaced word and lines, a safer choice is to go for line and word segmentation directly [17] (or combined). Another method (handling line and word separately) was [24], the approach showed great results compared to other methods we used with similar approach. The issue was that was somewhat dependent on document size (a few hyper-parameters/predefined values to be specific). We couldn't find the way to calculate them properly for the generic cases. For that reason, we propose to use Euclidean Distance Transform (EDT) with some basic morphological operations for error handling (as opposed to the statistical method described in original paper).

EDT Maps each pixel into its smallest distance to zones of interest. Pixels are already divided into foreground and background pixels. Distance Transform labels each object pixel of the image with distance between that pixel and the nearest background pixel. After EDT, we binarize the EDT of input image, with dynamically selected threshold) And apply connected component labeling to the binarized output (Figure: 3.3).

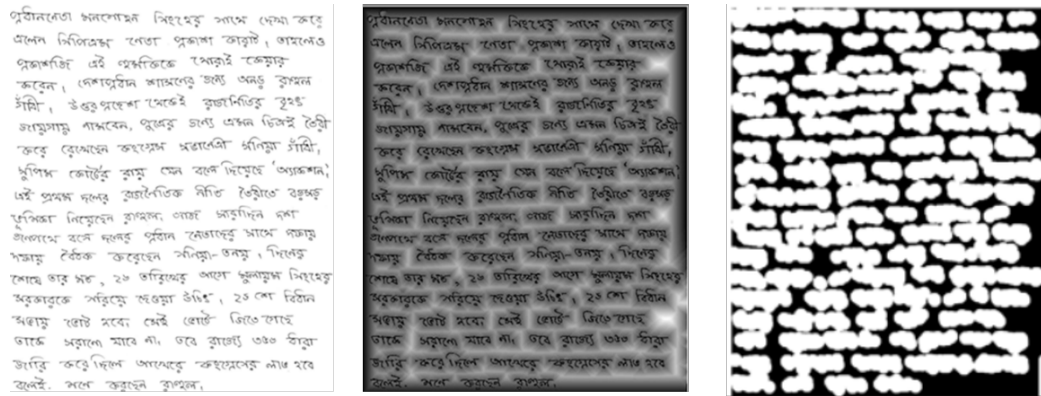


Figure 3.3: Original Image (Left), EDT of the image (Middle), Binarized Inverted EDT output-after CCL (right)

We use erosion, iteratively increasing the number of filter size- until the number of components starts to decrease. That becomes our filter size for opening operation (Figure 3.4). That worked quite better than the error correction formula given in the original work [17].

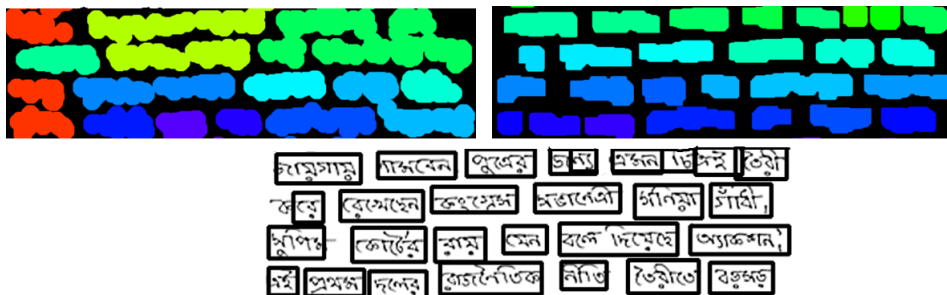


Figure 3.4: Connected Component Labeling on EDT image (Top-Left), after applying morphological operations (Top-Right), Output Image (Bottom)

### 3.3.2 Skew Correction (Word Image)

The issue with skew correction of segmented words with this approach is the top zone region at a side (toward leftmost or rightmost) of the word. That creates an issue while taking the active contour area, so we switch to another method for this step (Figure: 3.5) [6] had a unique approach. They used water



Figure 3.5: Issue with Active Contour- Original (Top) and Rotated (Bottom)

reservoir method- usually used for segmenting, to find the bottom points for drawing regression line to find the skew angle for word image. We faced some issue while finding the points while implementing this method which the issue stands with the writer not giving proper attention on Matra and word/characters without matra.

We took the inspiration from the approach, changing the approach to use the junction points (Figure: 3.6) here instead of bottom water reservoir points. We

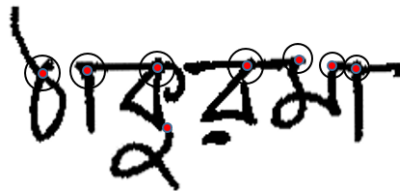


Figure 3.6: Junction Points

propose to pick the top points of verticalness components (5% of image height at either side of peak value) around the highest sum of black pixels of the horizontalness of the image (Figure 3.7). And draw a regression line for those points. The angle of the regression line with the horizontal axis is the skew angle (rotate it so that that is parallel to the horizontal axis).

If this approach fails (not enough points to draw a regression line or if rotated image projection profile peak is lesser than that original image), we move to Histogram Rotation method [22]



Figure 3.7: Potential Matra Region (Left), Points for regression line (Middle), One of the Results (Right)

### 3.3.3 Character Segmentation

Traditionally three zone approach is been used for character segmentation approach, Top zone, Middle zone, Bottom zone. But as 30% words don't have a bottom zone (in traditional sense) modifiers[2], (Figure 3.8) also most of the characters don't have a perfect baseline to choose from.

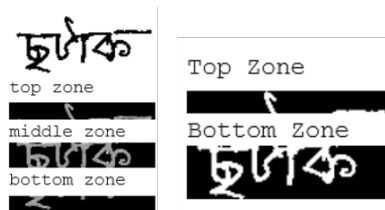


Figure 3.8: Issue with three zone approach

So we propose a two zone approach, the zone above Matra (Top zone) and below matra (Bottom zone- traditional middle and bottom zone combined). Later the lower zone is segmented. One thing to mention is we try not to detach the character with matra, but as we are working with matra line- so by definition that is top-and bottom zone.

### 3.3.3.1 Matra Detection and Separation

Traditionally matra region is considered as the peak value at histogram profile (along with a few other properties. But in some cases, that approach fails (Figure 3.8). For that reason we propose to pick the peak of horizontal histogram at the top half of image for Matra zone.

But directly discarding the matra region (not considering them for the character segm-entation process) would cause detachment of characters like (ঋ, ক, গ, জ, ণ, প, ফ, শ). Now we can improve the issue using some filter/condition based on the characteristics of the characters to work with. But success from this approach would be extremely dependent on the writing style and the datasets we will be using (how diverse is that handling all cases). A better method from our experiment was to try and pinpoint matra line using the matra zone we predicted. We use the skeleton image and extract junction points from there[25]. We used the junction points to recheck the predicted matra zone, specially the cases where matra is not exactly in the top half of the image. If we don't have more than 3 junction points at the region, we will pick the peak of horizontalness histogram for the whole image.

But the approach was not perfect as many Matra lines do not exist in the same horizontal line, even for skew corrected image. For that reason, we use consecutive junction points to determine our matra line (using the potential matra region found above). The steps are as follows:

Step a: Find the skeleton and Potential Matra of the segmented word image.

Step b: Find junction points and verticalness from the skeleton image.

Step c: Keep the verticalness line and junction points from potential matra zone

Step d: In case we have more junction points around same place, we keep the top center point and collapse the points to the centroid.

Step e: Around the top junction points is the "Not to Cut Zone" to preserve the characters that might get detached with matra removal. So from two junction

point distance we cut from  $\square$  to  $\square$ , and the rest is Not to Cut Zone.

Step f: Find the common area to cut the matra and the part that exist in verticalness of skeleton image. Then subtract the skeleton image and the result found in this step. The zones/regions will show matra detached area.

Necessary visuals shown in Figure: 3.9

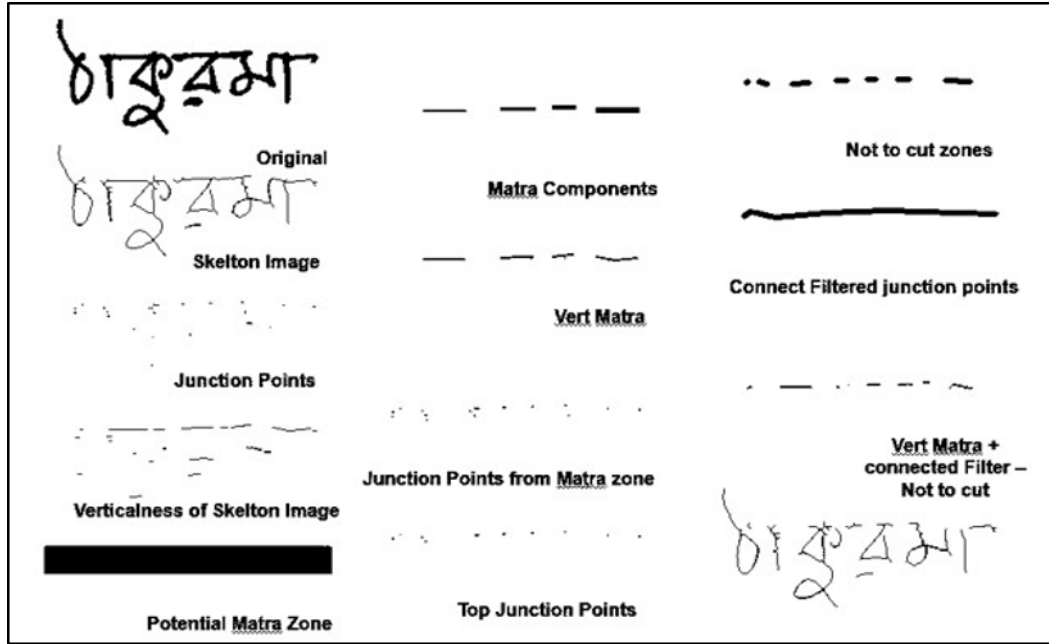


Figure 3.9: Matra removal process (Many pictures are inverted from implementation, for presentation purpose)

### 3.3.3.2 Character Separation

We detach the two zones- top and bottom based on our detected Matra region. We propose to use Connected Component Analysis [23] to initially segment the characters of both top and bottom zone (Figure: 3.10). Only this approach fails in considerable number of cases. To deal with the cases, we propose to find and use the characteristics of Bangla handwritten characters from the observations of different experiments and previous works from various authors. For Top zone segmentation (detached modifiers at top zone) the proposed approach works

just fine. We need something more experiments to give a concrete approach or method for segmenting the bottom zone characters.



Figure 3.10: Connected component analysis on matra separated image, Step-06

After we are done segmenting the top and bottom zone characters, we need to add the removed matra region to our characters (step-07)- Figure: 3.11.

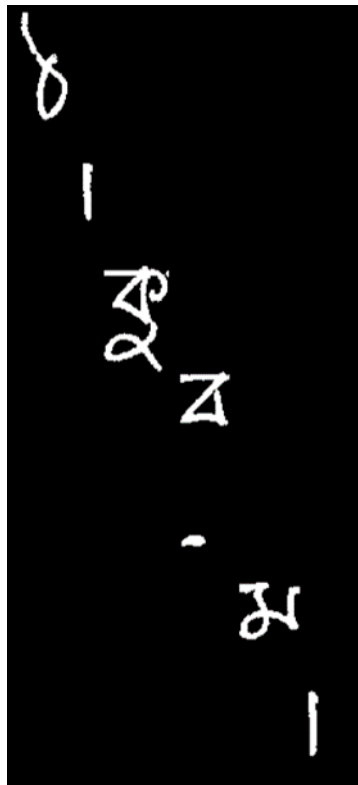


Figure 3.11: Matra added single characters on original image



### 3.3.3.3 Lower Zone Segmentation

We are left with the detecting/ separating the bottom zone modifiers from the whole character we have. The isolated modifiers (or part of character is already taken care of while segmenting the bottom zone. For the connected ones with bottom zone, we propose to use the junction points from the bottom half of the image- to mark the potential areas with lower zone modifier. If there is noticeable difference in foreground pixel count, around the intersecting point at the bottom half of the character- that can be identified as the baseline of the character to separate character and connected the lower modifier. The process is shown in Figure 3.12

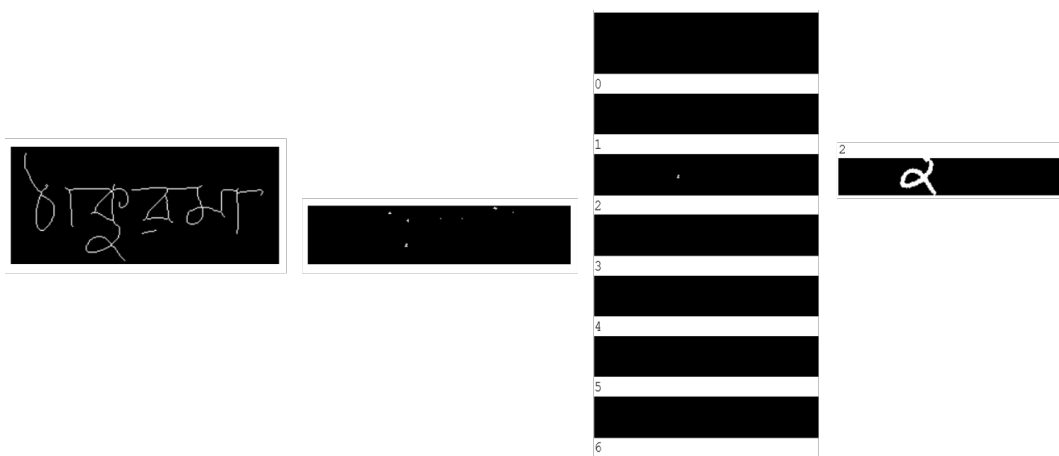


Figure 3.12: Lower Zone Segmentation

### 3.3.3.4 Adjusting Issues

The next step is to add the detached top zone components to its respective bottom zone part and adjust other commonly occurred issues. If any component is bounded by other, discard the smaller one. And add it to the bigger component (Figure: 3.13).



Figure 3.13: Issue Fix 01 (left error, right Adjusted)

If a top zone component has any component right below below it, consider the two as one character. We use Midpoint of rightmost consecutive pixels as the pivot in this case. Otherwise the component is added closest to the left of topmost foreground pixel, with same pivot (Figure: 3.14).



Figure 3.14: Issue Fix 02 (left error, right Adjusted)

The final case is for extension dot of (র, য়, ড়) and characters like ং. If components have similar height-width (around 1:1 ratio), and area is smaller than half the average area- we connect the component to top or bottom of the other bigger available component (Figure: 3.15)..



Figure 3.15: Issue Fix 03 (left error, right Adjusted)

### 3.4 Classification

We propose to use some existing model of deep convolutional neural network [12], [13], [19] (for the high accuracy of CNN based models[14]). In this phase, the task is to train the models beforehand with one of the existing dataset. Then keep the weights saved with most optimal performance different datasets and models. The goal was to improve the accuracy or efficiency after we find a suitable or base model to work around to improve on. Here we have summarized the results from different models found from our experiments with fixed number of epochs which taking similar amount of time to train. Rest is discussed in result section regarding the process and our experiments.

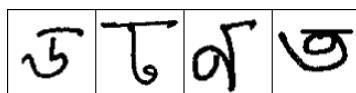
## Chapter 4

# EXPERIMENTATION RESULT AND ANALYSIS

### 4.1 Datasets

The first benchmark dataset for Bangla offline handwritten characters was CMATERdb [9]. A few version of this dataset covered quite some ground- but the size of the dataset is always been a concern (only sixty thousand images covering the basic characters).

- (32x32) pixel noise-free and blocker image edge (No pre-processing)
- 63,278 isolated character images (15,103 Basic + 6000 Numeics + 42,248 Compound Characters)



Later BanglaLekha-isolated [15] was larger dataset (Hundred sixty six thousand) with preprocessed inverted images. Adding the modifiers which is a massive improvement but the limited number of classes (total of 84- Basic characters- numerals and modifiers) was also an issue. And there was mislabeled images in almost every class.

- 166,105 character images (50 character (98950) + 10 numerics(19,748) + 24 selected compound(47407))
- Inverted and resized image (some padding added to be square with preserving the aspect ratio and consistent size)
- Noise removed with the median filter



Ekush [3] and Matrivasha [4] were two of the biggest addition for the amount of compound characters classes they covered as well the number of images is over double the size of Banglalekha-isolated, both having over three hundred thousand images in over 120 classes. Ekush mostly focused on basic characters and some of compound characters.

- 367,018 character images (3068 writers) (Preprocessed; both Black and white background)
- 122-character Classes (10 modifiers+ 50 characters+ 10 numeral+ 52(50+2 5) selected compound letter)
- jpg + csv format



And also the inclusion of banglaWriting [5] document page image dataset of 260 pages (one for each writer) was another milestone to standardize the work of segmentation specially.

- Single page writing x 260 writers [with 21,234 words (5470 unique) and 32,787 characters (174 unique)]
- Manually generated word box (crop and annotation) and label
- Image + JSON (RAW + converted)

## 4.2 Our Experiments

### 4.2.1 Segmentation

We tested our proposed approach of line-word segmentation on banglaWriting [5] page images. The method worked unless the writing was heavily congested/irregular.

Before that we tried to apply projection profile method [1], [2], Hough transform [10], Viterbi algorithm [11]. But the results were not that satisfactory.

For character segmentation experiments, we are running our tests on CMATERdb2.1.1 word image dataset[7]. So far we have found 4.34% (217 words out of 5000) with issues after skew correction and matra detection process. To work around the under-segmentation and over-segmentation issues found in the approach, a few observation/adjustments were addressed in section -3.3.3.4.

Our tests showed over 92% accuracy on segmentation for the characters (from word image). However, one issue was if the matra was too curved- the process had hard time separating matra in some cases. That is because our Do Not Segment region is harder to find, as the curve might not be always within the potential matra region detected. Detecting shapes between each two pair of junction points will most likely help, but if the curve is in our proposed region- the process works. So we have not checked the alternative approach yet.

### 4.2.2 Classification

We ran tests on a few of the existing models to check the classification accuracy. For the experiments, four setup of datasets were used to finalize the model to be used for classifying the segmented characters. Banglalekha isolated, CmaterDB, Ekush and Mixed- combining the classes from Ekush [3] and Matrivasha [4] along with 10 punctuation classes, making it total 252 classes.

Table 4.1: Classification Summary

Model	Parameters	Mixed (242 classes)	Ekush (122 Classes)	BanglaLekha Isolated(84)	Cmaterdb (50 Classes)
Ekush	1.6M	92.41%	94.91%	89.69%	91.23%
Borno	4.2M	91.7%	93.56%	91.02%	92.2%
ResNet50	23.5M	91.46%	91.95%	85.22%	94%
ResNet50-V2	24M	90.01%	91.90%	85.13%	94.88%
DenseNet121	7M	92.38%	93.27%	89.17%	93.5%
VGG-19	13.7M	90.55%	90.82%	86.41%	90.33%

We divide the dataset into 70-20-10 ratio for train-test-validation. Adam optimizer was used with learning rate 0.001 and categorical cross-entropy as loss metric. Results after 30 epochs for different models is shown in (Table- 4.1).

We checked these better performing models with our segmented characters segmented from CmaterDb word image dataset. Mostly the models performed similar given they have the class for the characters they have to predict (around 70% accuracy). But that process was somehow tiring- as we have to do and check everything manually and we don't have that much sample to work around-decide upon. So we used the other datasets with lesser classes than original. (e.g Mixed for rest 3, Ekush with Banglalekha-isolated and Cmaterdb). We used letter frequency from [26] [27] and randomly picked characters from dataset classes based on their frequency. To make the point clear, we used the percentage of that particular character used as the number of character, for each hundred character the data for over 40% frequency we have in our hand[26]. Then for the rest of the characters, we prioritized the vowel modifiers first and then remaining basic characters. Later the consonant modifiers and at the end the compound characters. So the resulting images from the process gave us some representation of what the classification model might see in real life setup. To reiterate the hypothesis in clearer words, we will give our segmented feed to the classification image, which will be segmented characters- single/simple compound. So that

gives us a better idea with dataset real setup simulation with more images and labeled data. And that gives us a better idea how good the model itself is, rather than relying on the mistakes from segmentation process and piling up together as a unit system. Though the accuracies varied on the test set- the performance or somewhat similar in terms of accuracy in that real life simulation. So we can pick model Ekush trained on Ekush Dataset for some convenience in both training and classification



## Chapter 5

# FUTURE WORKS AND CONCLUSION

A very important part to complete our work is the post-processing part, which is still missing in Bangla OCR system, so that might be the top priority for anyone trying to work on this field in future. Specially the error handling part of it. To conclude, the contribution from us to the existing works can be seen in segmentation- We added our new touch to line-word segmentation. Then word skew correction we used new idea from existing approach, and a completely new approach to separate matra. After connected component analysis- matra restore, issue adjustment and lower zone segment had our new touch in it too. We might've missed some of issues that will need to be found and fixed later with further study.

At the end, we just summarized the classification models giving us better results. So optimizing the model for our task either on accuracy or computation cost or both will be another priority for further works in this field. Though handwritten ocr is not fully explored yet. But the recent focus on it will surely progress well. Hopefully our humble effort in this field to add some guidance to the later works going to be added in this field, OCR for handwritten bangla characters and take steps forward toward completion, the goal we share all along with our work.

## REFERENCES

- [1] B. B. Chaudhuri and U. Pal, "A complete printed Bangla OCR system," *Pattern Recognit.*, vol. 31, no. 5, pp. 531–549, 1998, doi: 10.1016/S0031-3203(97)00078-2.
- [2] F. Yeasmin Ome, S. Shabbir Himel, and A. Naser Bikas, "A Complete Workflow for Development of Bangla OCR," *Int. J. Comput. Appl.*, vol. 21, no. 9, pp. 1–6, 2011, doi: 10.5120/2543-3483.
- [3] A. K. M. S. A. Rabby, S. Haque, M. S. Islam, S. Abujar, and S. A. Hossain, *Ekush: A Multipurpose and Multitype Comprehensive Database for Online Off-Line Bangla Handwritten Characters*, vol. 1037. Springer Singapore, 2019.
- [4] J. Ferdous, S. Karmaker, A. K. M. S. A. Rabby, and S. A. Hossain, "MatriVasha: A Multipurpose Comprehensive Database for Bangla Handwritten Compound Characters," *Lect. Notes Networks Syst.*, vol. 164, pp. 813–821, 2021, doi: 10.1007/978-981-15-9774-9\_74
- [5] M. F. Mridha, A. Q. Ohi, M. A. Ali, M. I. Emon, and M. M. Kabir, "BanglaWriting: A multi-purpose offline Bangla handwriting dataset," *Data Br.*, vol. 34, p. 106633, 2021, doi: 10.1016/j.dib.2020.106633.
- [6] P. P. Roy, A. K. Bhunia, A. Das, P. Dey, and U. Pal, "HMM-based Indic handwritten word recognition using zone segmentation," *Pattern Recognit.*, vol. 60, pp. 1057–1075, 2016, doi: 10.1016/j.patcog.2016.04.012.
- [7] S. Malakar, R. Sarkar, S. Basu, M. Kundu, and M. Nasipuri, "An image database of handwritten Bangla words with automatic benchmarking facilities for character segmentation algorithms," *Neural Comput. Appl.*, vol. 33, no. 1, pp. 449–468, 2021, doi: 10.1007/s00521-020-04981-w.
- [8] S. Basu, R. Sarkar, N. Das, M. Kundu, M. Nasipuri, and D. K. Basu, "A fuzzy technique for segmentation of handwritten Bangla word images," *Proc. - Int. Conf. Comput. Theory Appl. ICCTA 2007*, no. March, pp. 427–432, 2007, doi: 10.1109/ICCTA.2007.7.

- [9] R. Sarkar, N. Das, S. Basu, M. Kundu, M. Nasipuri, and D. K. Basu, "CMATERdb1: A database of unconstrained handwritten Bangla and Bangla-English mixed script document image," *Int. J. Doc. Anal. Recognit.*, vol. 15, no. 1, pp. 71–83, 2012, doi: 10.1007/s10032-011-0148-6.
- [10] Y. Park, "Discrete Hough transform using line segment representation for line detection," *Opt. Eng.*, vol. 50, no. 8, p. 087004, 2011, doi: 10.1117/1.3607414.
- [11] V. Papavassiliou, T. Stafylakis, V. Katsouros, and G. Carayannis, "Handwritten document image segmentation into text lines and words," *Pattern Recognit.*, vol. 43, no. 1, pp. 369–377, 2010, doi: 10.1016/j.patcog.2009.05.007.
- [12] A. K. M. Shahariar, A. Rabby, S. Haque, S. Abujar, and S. A. Hossain, "EkushNet : Using Convolutional Neural Network for Bangla Handwritten character recognition" *Procedia Comput. Sci.*, vol. 143, no. December, pp. 603–610, 2018, doi: 10.1016/j.procs.2018.10.437.
- [13] S. Haque, S. Abujar, S. Abujar, S. Akhter, and S. A. Hossain, "BornoNet : Bangla Handwritten Characters Recognition Using Convolutional Neural Network Convolutional Neural Network," vol. 00, 2018.
- [14] S. Irfan and A. Meerza, "Performance Evaluation of Different Algorithms for Handwritten Isolated Bangla Character Recognition," 2019 *Int. Conf. Robot. Signal Process. Tech.*, pp. 412–416, 2019.
- [15] M. Biswas et al., "BanglaLekha-Isolated: A Comprehensive Bangla Handwritten Character Dataset," pp. 1–4, 2017, [Online]. Available: <http://arxiv.org/abs/1703.10661>.
- [16] R. Sarkar, S. Malakar, N. Das, S. Basu, M. Kundu, and M. Nasipuri, "Word extraction and character segmentation from text lines of unconstrained handwritten Bangla document images," *J. Intell. Syst.*, vol. 20, no. 3, pp. 227–260, 2011, doi: 10.1515/JISYS.2011.013.

- [17] P. K. Singh, S. Sinha, S. P. Chowdhury, R. Sarkar, and M. Nasipuri, "Word segmentation from unconstrained handwritten Bangla document images using distance transform," 6th Int. Conf. Adv. Comput. Control. Telecommun. Technol. ACT 2015, pp. 473–484, 2015, doi: 10.1515/9783110450101-041.
- [18] U. Pal and S. Datta, "Segmentation of Bangla unconstrained handwritten text," Proc. Int. Conf. Doc. Anal. Recognition, ICDAR, vol. 2003-Janua, no. August, pp. 1128–1132, 2003, doi: 10.1109/ICDAR.2003.1227832.
- [19] M. Z. Alom, P. Sidike, M. Hasan, T. M. Taha, and V. K. Asari, "Handwritten Bangla Character Recognition Using the State-of-the-Art Deep Convolutional Neural Networks," Comput. Intell. Neurosci., vol. 2018, pp. 1–12, 2018, doi: 10.1155/2018/6747098.
- [20] D. Liu and J. Yu, "Otsu method and K-means," Proc. - 2009 9th Int. Conf. Hybrid Intell. Syst. HIS 2009, vol. 1, no. 2, pp. 344–349, 2009, doi: 10.1109/HIS.2009.74.
- [21] H. Fan, L. Zhu, and Y. Tang, "Skew detection in document images based on rectangular active contour," Int. J. Doc. Anal. Recognit., vol. 13, no. 4, pp. 261–269, 2010, doi: 10.1007/s10032-010-0119-3.
- [22] N. Ouwayed and A. Belaïd, "A general approach for multi-oriented text line extraction of handwritten documents," Int. J. Doc. Anal. Recognit., vol. 15, no. 4, pp. 297–314, 2012, doi: 10.1007/s10032-011-0172-6.
- [23] L. Zhou, Y. Lu, and C. L. Tan, "Bangla/English script identification based on analysis of connected component profiles," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 3872 LNCS, pp. 243–254, 2006, doi: 10.1007/11669487\_22.
- [24] "GitHub - subashis-dev/Bangla-handwritten-word-segmentation-from-document: Segment all the words from a Bengali handwritten document easily." <https://github.com/subashis-dev/Bangla-handwritten-word-segmentation-from-document> (accessed Feb. 23, 2022).

- [25] H. Kong, “A MEDIAL AXIS BASED THINNING STRATEGY AND STRUCTURAL FEATURE Soumen Bag and Gaurav Harit Department of Computer Science and Engineering Indian Institute of Technology Kharagpur , Kharagpur-721 302 , India,” pp. 2173–2176, 2010.
- [26] B. B. Chaudhuri, “Corpus-based empirical analysis of form , function and frequency of characters used in Bangla.”
- [27] “Letter frequency.” <http://simia.net/letters/> (accessed Mar. 03, 2022).