B.Sc. in Computer Science and Engineering Thesis

Bangla Character Recognition using Artificial Neural Network Step:Feature Selection

Submitted by

Deepon Debnath
201014038

Maj Md. Abdullah Al Mamun
201014010

Maj Md. Sarwar Hossain
201014003

Supervised by

Dr. Hasan Sarwar
Professor and Head of the Department, CSE
United International University

Department of Computer Science and Engineering
Military Institute of Science and Technology

CERTIFICATION

This thesis paper titled **"Bangla Character Recognition using Artificial Neural Network Step:Feature Selection"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering on December 2013.

Group Members:

Deepon Debnath
Maj Md. Abdullah Al Mamun
Maj Md. Sarwar Hossain

Supervisor:

_____-

Dr. Hasan Sarwar
Professor and Head of the Department, CSE
United International University
House: 80, Road: 8/A, Sat Masjid Road, Dhanmondi, Dhaka-1209

CANDIDATES' DECLARATION

This is to certify that the work presented in this thesis paper is the outcome of the investigation and research carried out by the following students under the supervision of Dr. Hasan Sarwar, Professor and Head of the Department of Computer Science and Engineering, United International University, House: 80, Road: 8/A, Sat Masjid Road, Dhanmondi, Dhaka, Bangladesh.

It is also declared that neither this thesis paper nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

_____

Deepon Debnath
201014038

_____

Maj Md. Abdullah Al Mamun
201014010

_____

Maj Md. Sarwar Hossain
201014003

ACKNOWLEDGEMENT

We are thankful to Almighty Allah for his blessings for the successful completion of our thesis. Our heartiest gratitude, profound indebtedness and deep respect go to our supervisor Dr. Hasan Sarwar, Professor and Head of the Department, CSE, United International University, House: 80, Road: 8/A, Sat Masjid Road, Dhanmondi, Dhaka, Bangladesh, for his constant supervision, affectionate guidance and great encouragement and motivation. His keen interest on the topic and valuable advices throughout the study was of great help in completing thesis.

We are especially grateful to the Department of Computer Science and Engineering (CSE) of Military Institute of Science and Technology (MIST) for providing their all out support during the thesis work.

Finally, we would like to thank our families and our course mates for their appreciable assistance, patience and suggestions during the course of our thesis.

Dhaka
December 2013

Deepon Debnath
Maj Md. Abdullah Al Mamun
Maj Md. Sarwar Hossain

ABSTRACT

Feature selection is an essential step of Optical Character Recognition. Accurate and distinguishable feature plays a significant role to leverage the performance of a classifier. The complexity level of feature identification algorithm differs for alphabet sets of different languages. Apart from generic algorithms to find features of different alphabet sets, these algorithms take care of individual characteristic common for a particular alphabet set. Dominant features of one alphabet set might completely differ from that of another set. Since there always remains the chance that inaccurate features may cause inefficient recognition, special attention should be given to identify the set of optimal features of a character set. Bengali characters also have some specific issues apart from the existing issues of other character sets. For example, there are about 300 basic, modified and compound character shapes in the script, the characters in a word are topologically connected, and Bengali is an inflectional language. Literature survey shows that several authors have used different features and classification algorithms. We have extensively reviewed all these feature sets. In order to identify an optimal feature set, variability analysis has been proposed here. We focused on the specific peculiarities of Bengali alphabet sets, its different usage as vowel and consonant signs, compound, complex and touching characters. We also took care to generate easily computable features that take less time for generation.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATION

OCR : Optical Character Recognition

ANN : Artificial Neural Network

MLP : Multi Layer Perceptron

KNN : Kohenen Neural Network

CHAPTER 1

INTRODUCTION

1.1 Overview

Recognition of printed and handwritten documents is still one of the most challenging areas in pattern recognition with profound implications for the machine vision field. Although many different methods have been reported and some have shown very high performance, none has been able to achieve the accuracy and speed of human readers, which is the ultimate target. So there is ample scope for improvement in this well-researched problem.

Optical Character Recognition (OCR) System, 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. As we all know, Bangla is one of the richest languages of the world, ranking 5^{th} in the world. More than 200 million people use Bangla as their medium of communication. 21^{st} 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. Moreover, with the rapid growth and advertisement of the use of computers in Bangladesh, a digitized method for recognizing Bangla characters is started to receive attention for OCR related research in the recent years.

The selection of feature is very important for Optical Character Recognition.The objective of feature extraction is to capture the essential characteristics of the symbols and it is generally accepted that this is one of the most difficult problems of pattern recognition. Literature survey shows that several authors have used different features and classification algorithms. We have extensively reviewed all these feature sets.We focused on the specific peculiarities of Bengali alphabet sets, its different usage as vowel and consonant signs, compound, complex and touching characters.

1.2 What is OCR?

Optical character recognition, usually abbreviated to OCR, is the mechanical or electronic conversion of scanned images of handwritten, typewritten or printed text into machine-encoded text. It is widely used as a form of data entry from some sort of original paper data source, whether documents, sales receipts, mail or any number of printed records. It is a common method of digitizing printed texts so that they can be electronically searched, stored more compactly, displayed on-line and used in machine processes such as machine translation, text-to-speech and text mining. OCR is a field of research in pattern recognition, artificial intelligence and computer vision [1].

OCR systems require calibration to read a specific font; early versions needed to be programmed with images of each character and worked on one font at a time. "Intelligent" systems with a high degree of recognition accuracy for most fonts are now common. Some systems are capable of reproducing formatted output that closely approximates the original scanned page including images, columns and non-textual components.

1.2.1 The History of OCR

1870	The very first attempt
1940	The modern version of OCR
1950	The first OCR machine
1960 –1965	First generation of OCR
1965 –1975	Second generation of OCR
1975 –1985	Third generation of OCR
1986	OCR to people

Figure 1.1: A short OCR chronology

Methodically, character recognition is a subset of the pattern recognition area. However, it was character recognition that gave the incentives for making pattern recognition and image analysis matured fields of science. Although, OCR machines became commercially

available already in the 1950's, only a few thousand systems had been sold worldwide up to 1986. The main reason for this was the cost of the systems. However, as hardware was getting cheaper and OCR systems started to become available as software packages, the sale increased considerably.

1.2.2 Structure of OCR System

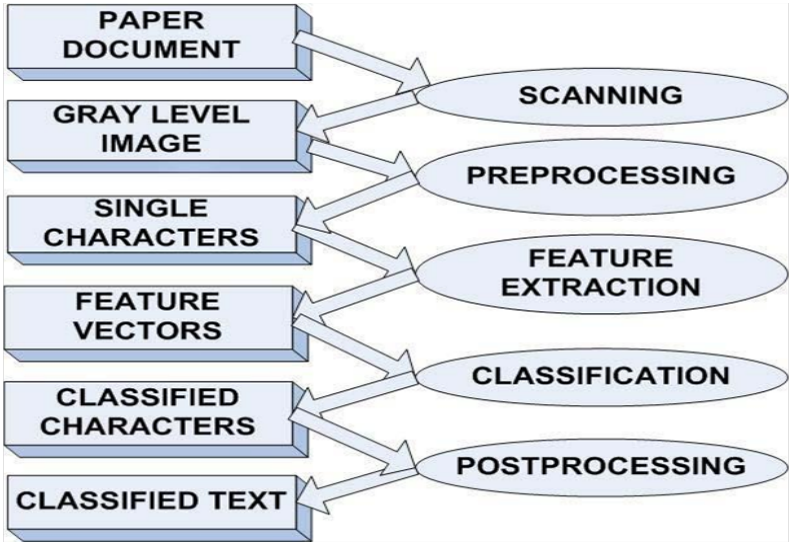


Figure 1.2: Typical structure of OCR system

Although OCR system can be develop for different purposes, for different languages, an OCR system contains some basic steps. Figure-1.2, describes the basic steps of an OCR. A basic OCR system has the following particular processing steps:

1. Scanning.
2. Preprocessing.
3. Feature extraction or pattern recognition.
4. Recognition using one or more classifier.
5. Contextual verification or post processing.

1.3 A Brief on Bangla Alphabet

The origin of character recognition can be found in 1870 when Carey invented the retina scanner; an image-transmission system using a mosaic of photocells [2]. Later in 1890, Nipkow invented the sequential scanner which was a major breakthrough both for modern television and reading machines. Character recognition as an aid to the visually handicapped was at first attempted by the Russian scientist Tyurin in 1900. The OCR technology took a major turn in the middle of 1950s, with the development of digital computer and improved scanning devices. We are concerned here with the recognition of Bangla, the second most popular script and language in the Indian sub-continent. About 200 million people of Eastern India and Bangladesh use this language, making it the fourth most popular in the world .

The structure of Bangla language is quite different than any other European language. It consists of 50 basic characters including 11 vowel and 39 consonant characters. Again, there are vowel and consonant modifiers, touching and compound characters which are formed by touching the adjacent characters and by combining 2 or more characters in the word accordingly. So when they are used to form words, we find that thousands of various combinations (both simple and complex) are formed. Most of the characters have a horizontal line at the upper part. This horizontal line is called 'Headline' or 'Matra- line'. If the first character of a word is a vowel, then it remains in its independent form [3].

1.3.1 Characters of Bangla Script

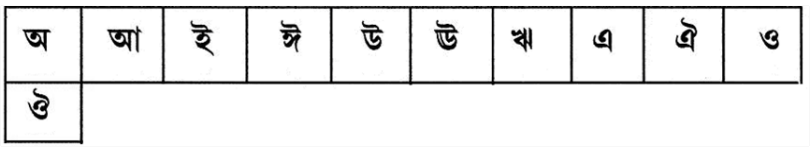


Figure 1.3: Vowels (Basic Characters)

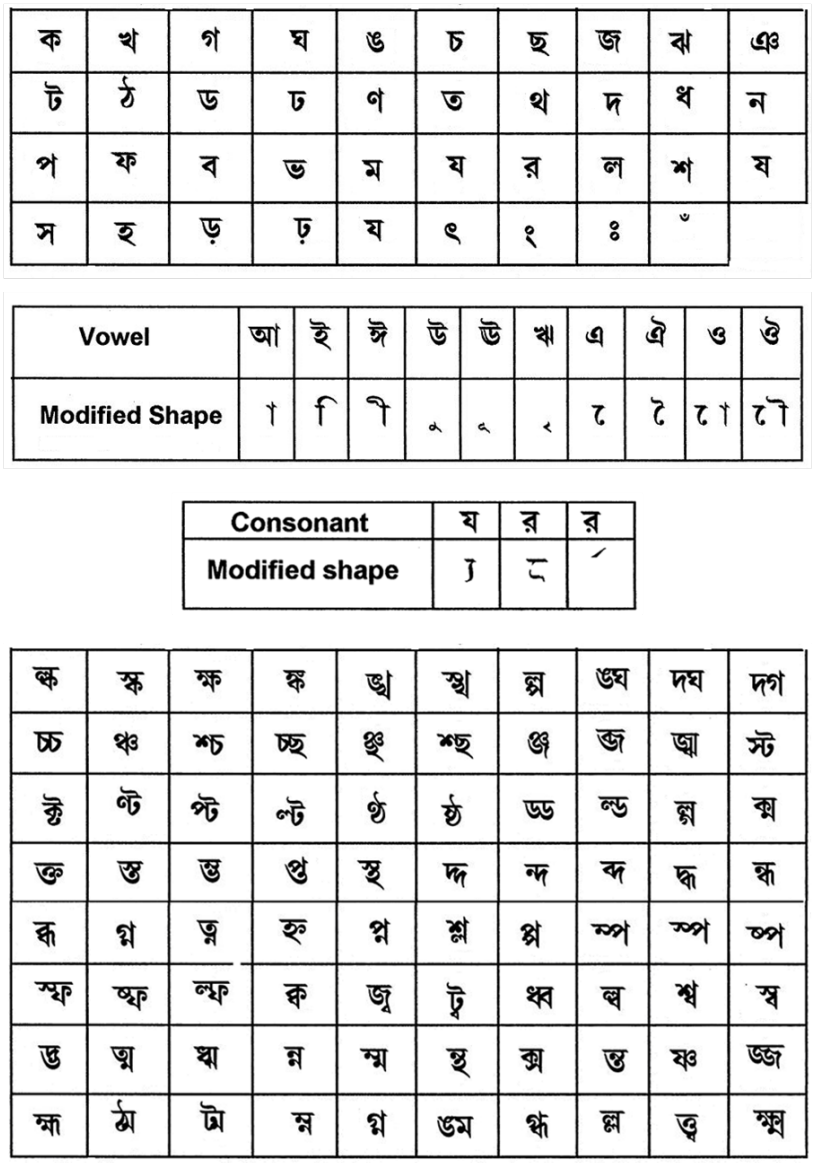


Figure 1.4: Consonants (Basic Characters), Vowel Modifiers, Consonant Modifiers, Some Compound Characters

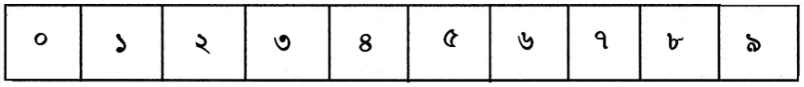


Figure 1.5: Bangla Digits

1.3.2 Distinct Features of Bangla Characters

Some common properties in Bengali language are given below [4]:

- Bengali script flows from left to right.
- Characters are not classified as uppercase or lowercase.
- A vowel with a consonant takes a modified shape known as vowel modifier.
- A consonant with a consonant takes a modified shape known as consonant modifier.
- The vowel always occurs at the beginning of the word.
- There are about 250 compound characters.
- Many characters have “matra” or headline with them. Some characters have a signature extended above the head line.
- In a standard text piece, 95.63% percent of the characters are basic characters and the rest 4.27% are compound characters. Fig-1.6, shows a simple example explaining the construction of a Bangla word.

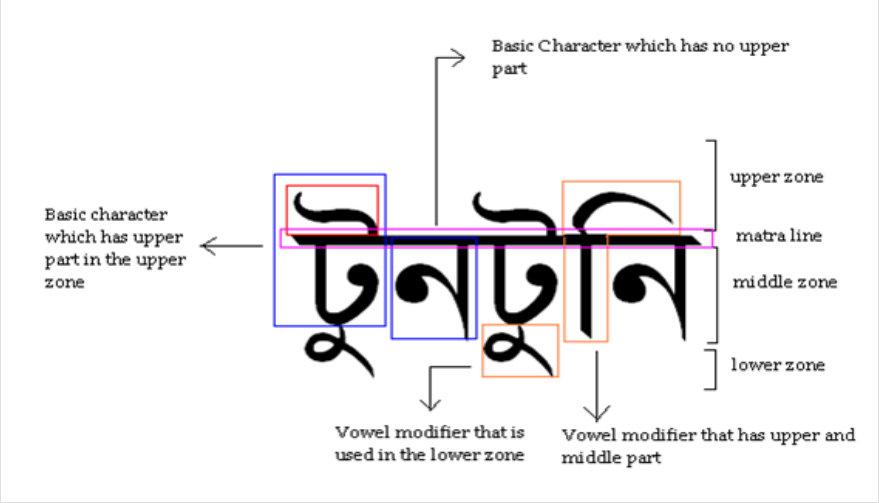


Figure 1.6: Dissection of Bangla word

1.4 Status of Bangla OCR

Researchers have been working on Bangla OCR since 1990. Institutes of India and Bangladesh have conducted different projects and research works, but commercially standard Bangla OCR is not available still now. The available OCR systems are: BORCA[2006], AponaPathak[2006], BanglaOCR[2007]. BanglaOCR are a complete OCR framework, and has a recognition rate of up to 98% (in limited domains) but it also have many limitations.

CHAPTER 2

CHARACTER RECOGNITION SYSTEM

2.1 Introduction

Optical character recognition has become one of the most successful applications of technology in the field of pattern recognition and artificial intelligence. A typical OCR system contains three logical components: an image scanner, OCR software and hardware and an output interface. The image scanner optically captures text images to be recognized. Text images are processed with OCR software and hardware. The process involves three operations: document analysis (extracting individual character images), recognizing these images (based on shape), and contextual processing (either to correct misclassifications made by the recognition algorithm or to limit recognition choices). OCR software attempts to identify characters by comparing shapes to those stored in the software library or database. The software tries to identify words using character proximity and will try to reconstruct the original page layout. High accuracy can be obtained by using sharp, clear scans of high-quality originals. The output interface is responsible for communication of OCR system results to the outside world.

2.2 Steps of Bangla Character Recognition Process

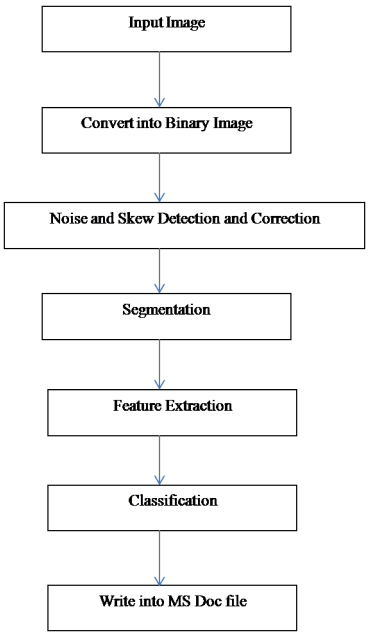


Figure 2.1: Steps of Bangla character recognition process

2.3 Input Variation

There are many forms of inputs which are fed into the segmentation phase of the Bangla OCR system. The types are shown in Figure-2.2, we have picked up Handwritten Bangla documents in this research work.

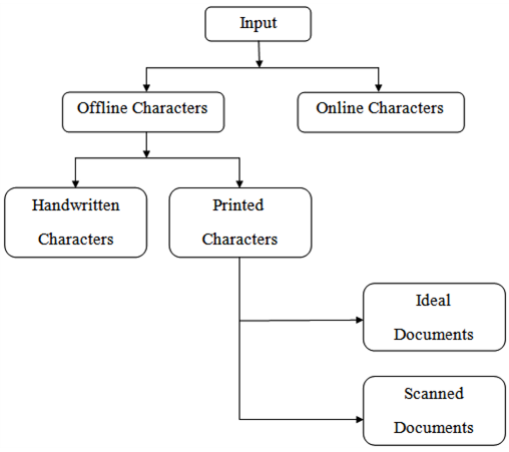


Figure 2.2: Input Categories of a Bangla OCR

2.4 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. 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.

2.5 Binarization Methods

Binarization of scanned gray scale images is the first step in most document image analysis systems. Selection of an appropriate binarization method for an input image domain is a difficult problem. Different Binarization algorithm gives different performances on different data sets. This is especially true in the case of historical documents with variation in contrast and illumination, smearing and smudging of text. The quality of the image has a significant impact on the OCR performance. Noise is detected and eliminated during Binarization. So, selection of appropriate binarization algorithm is very important for OCR performance. Some binarization methods are-

1. Global Fixed Threshod.
2. Local Threshold Approach.
3. Bernsen Method.
4. Niblack Method.
5. Adaptive Niblack's Algorithm (New Approach).
6. Parker Method.
7. Sauvola Binarization.
8. Chang's Method.

2.5.1 Global Fixed Threshold

A natural way of binarization is through threshold. Threshold creates binary images from grey-level ones by turning all pixels below some threshold to zero and all pixels about that threshold to one. The simplest binarization technique is to use a global fixed threshold. If g(x, y) is a threshold version of f(x, y) at some global threshold T,

$$g(x,y) = \begin{cases} 1; if\ f(x,y) & \geq T \\ 0; otherwise & \end{cases}$$

Global threshold method chooses a fixed intensity threshold value T based on the histogram analysis. The major problem with threshold is that we consider only the intensity, not any relationships between the pixels. There is no guarantee that the pixels identified by the threshold process are contiguous.

Another problem with global threshold is that most of the images do not have a bi-model distribution in histogram due to high illumination changes. For such images finding a single threshold value to binarize using global binarization method is difficult

2.5.2 Local Threshold Approach

We can avoid uneven illumination of an image by determining thresholds locally. That is, instead of having a single global threshold, we allow the threshold itself to smoothly vary across the image.

2.5.3 Bernsen Method

For each pixel (x, y), the threshold T(x, y) = (Z_{low} + Z_{high}) / 2 is used, where Z_{low} and Z_{high} are the lowest and highest gray scale pixel value in a square r x r neighborhood centered at (x, y). However, if the contrast measure C(x, y) = (Z_{high} - Z_{low}) $<$ l, then the neighborhood consists only one class, print or background. In our images, wide print areas rarely occur, so the pixel is labeled background in such cases. l = 15 and r = 15 to be good choices [5].

Figure 2.3: Binarization using Bernsen method

2.5.4 Niblack Method

Niblack's algorithm [6] calculates a pixel wise threshold by sliding a rectangular window over the grey level image. The threshold T is computed by using the mean m and standard deviation s, of all the pixels in the window, and is denoted as:

$$
\begin{aligned}
T_{Niblack} &= \mathrm{m} + \mathrm{k} * \mathrm{s} \\
&= \mathrm{m} + \mathrm{k} * \sqrt{\sum (Pi - m)^2 / NP} \\
&= \mathrm{m} + \mathrm{k} * \sqrt{\sum Pi^2 / NP - m^2} \\
&= \mathrm{m} + \mathrm{k}\sqrt{B}
\end{aligned}
$$

Where k is a constant, which determines how much of the total print object edge is retained, and has a value between 0 and 1. The value of k and the size SW of the sliding window define the quality of binarization, where NP is the number of pixels in the gray image, m is the average value of the pixels pi, and k is fixed to -0.2 by the authors. Advantage of Niblack is that it always identifies the text regions correctly as foreground but on the other hand tends to produce a large amount of binarization noise in non-text regions also.

2.5.5 Sauvola Method

The Sauvola method [7] for local binarization does quite well. The basic idea behind Sauvola is that if there is a lot of local contrast, the threshold should be chosen close to the mean value, whereas if there is very little contrast, the threshold should be chosen below the mean, by an amount proportional to the normalized local standard deviation. Sauvola is implemented efficiently by using "integral image" accumulators for the mean and mean-squared pixel values. The latter requires 64 bit floating point arrays, which are expensive for large images.

Sauvola's algorithm is a modification of Niblack's which is claimed to give improved performance on documents in which the background contains light texture, big variations and uneven illumination. In this algorithm, a threshold is computed with the dynamic range of the standard deviation, R, using the equation:

$$T = m * (1 + k(s/R - 1))$$

where m and s are again the mean and standard deviation of the whole window and k is a fixed value.

2.6 Noise Removing

2.6.1 Median Filter

The Median Filter [8] is a nonlinear filtering technique, often used to remove noise. Such noise reduction is a typical pre-processing step to improve the results of later processing. This filtering procedure is used to examine a sample of the input. It is performed using a window consisting of an odd number of input data samples. Median filtering is very widely used in digital image processing, particularly useful to reduce noise, salt and pepper noise. It is one kind of smoothing technique as well. All smoothing techniques are effective at removing noise in smooth patches or smooth regions of a signal, but adversely affect edges.

2.6.2 Gaussian Filter

A Gaussian filter [9] is a filter whose impulse response is a Gaussian function. Gaussian filters are designed to give no overshoot to a step function input while minimizing the rise and fall time. This behavior is closely connected to the fact that the Gaussian filter has the minimum possible group delay. Gaussian filtering is a linear convolution algorithm unrelated to Median filter. The one dimensional Gaussian filter has an impulse response given by,

$$g(x) = \sqrt{\frac{a}{\pi}} \cdot e^{-a.x^2}$$

Or, with the standard deviation as parameter:

$$g(x) = \frac{1}{\sqrt{2\pi}\sigma}.e^{-\frac{x^2}{2\sigma^2}}$$

In two dimensions, it is the product of two such Gaussians, one per direction:

$$g(x,y) = \frac{1}{2\pi\sigma^2}.e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the standard deviation of the Gaussian distribution. Gaussian filtering allows user to make fine adjustment to the amount of partial averaging that occurs in the image.

2.7 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 [10] says that it can be corrected in two steps-

- Estimation of skew angle θ_s and
- Rotation of image by θ_s in the opposite direction.

Many skew detection and correction algorithms are available. At present, skew detection methods can be roughly classified as follows:

- The method based on Hough transforms.
- The skew detection method based on the analysis of the texture complexity.
- The method based on Cross Correlation.
- The method based on the Projection profile.
- The method based on Fourier transformation.
- The K nearest neighbor (K-NN) cluster method.

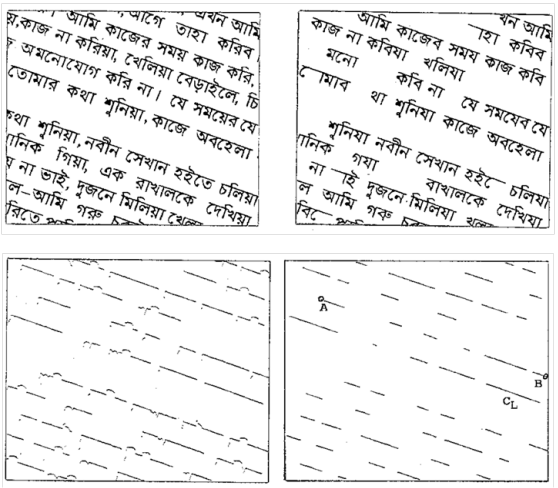


Figure 2.4: An example of skew detection approach (Bangla)

2.8 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-

- Line Detection.
- Matraline or Headline detection.
- Baseline Detection.
- Word Segmentation.

- Character Segmentation.

2.9 Feature Extraction

Feature extraction is an important and challenging part for any character recognition process. It is the process of extracting essential information content from the image segment. It plays an important role in the whole recognition process.By generating a fixed size matrix according to the height and weight of the bitmap image by the color value of the pixels. The black pixels are considered as 1's and the white pixels are as 0's. Figure-2.5, shows a converted feature matrix.

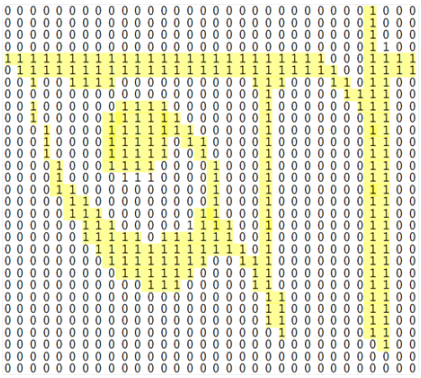


Figure 2.5: Feature Matrix

2.10 Classification

This is the last phase of the whole recognition process. Several approaches have been used to identify a character based on the features extracted using algorithms described in chapter three. However, there is no benchmark databases of character sets to test the performance of any algorithm developed for Bengali character recognition. In choosing classification algorithms, use of Artificial Neural Network (ANN) is a popular popular practice because it works better when input data is affected with noise. Since a detailed definition of different Neural Network is outside the scope of this chapter, the readers are suggested to see established literature for detailed description of these methods. However a brief overview is provided for Decision tree and MLP classifier with some other neural networks.

2.10.1 Decision Tree

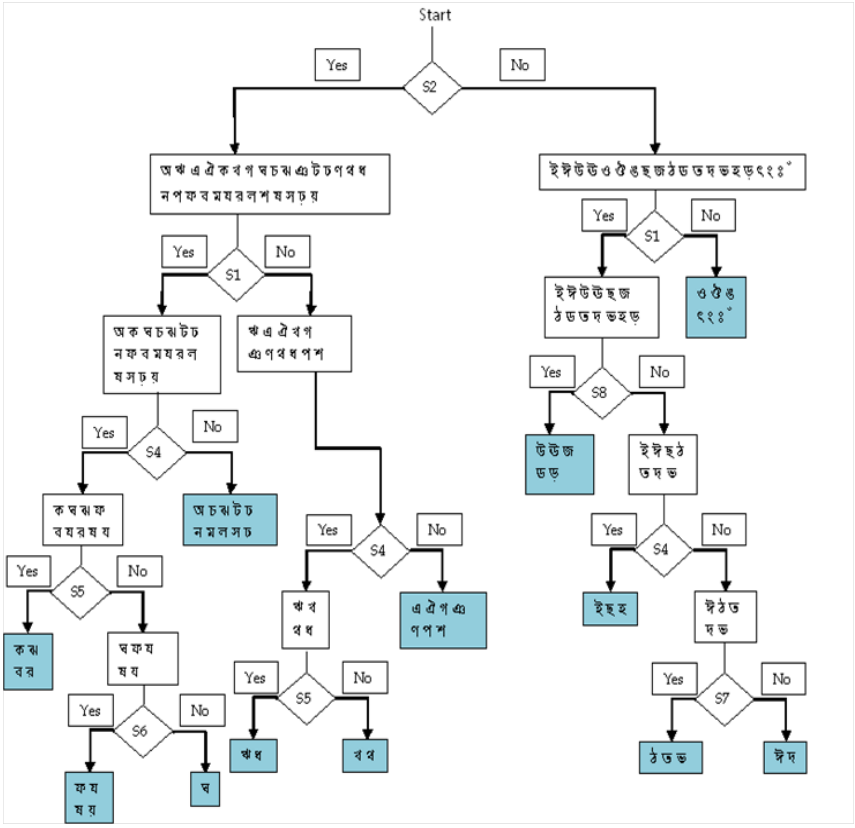


Figure 2.6: A flow chart representation of a certain portion of tree classifier for basic character recognition

B.B. Chaudhuri [10] used a binary tree classification. Only one feature is tested at each non terminal node. The decision rules are mostly binary e.g. Presence or absence of the feature. Here, the features are positional strokes. As one goes down the tree, the number of features to choose from gets reduced. Most basic characters can be recognized by the principle features alone. In some cases, more than one characters share the same non-terminal node of the tree. To separate them additional features are used. Compound character recognition is done in 2 stages. In the first stage, the characters are grouped into small subsets by a feature-based tree classifier. At the second stage, characters in each group are recognized by a sophisticated run-based template matching approach. A terminal node of this tree corresponds to a subset of about 20 characters. These character templates are ranked in terms of their bounding box width and stored during the training phase of the classifier. When a character reaches the terminal node in search phase, firstly bounding box width is

matched, and then a matching score is completed by superimposing the candidate on the template. Different algorithms have been prescribed to compute matching score. In this process, a reasonable amount of character size variation can be accommodated by rescaling.

2.10.2 MLP (Multi-Layer Perceptron)

Subhadip Basu [11] has used Multi-Layer Perceptron (MLP) classifier to classify handwritten alphabetic characters. It is a special kind of ANN, a feed-forward neural network with artificial neurons. An MLP consists of one input layer, one output layer and a number of hidden layers. The output of each neuron is connected to each neuron of the immediate next layer as input. Neurons in the input layer are used to simply pass the information to the next layer. Supervised training is applied. Back Propagation has been used here which minimizes the sum of square error for the training samples by conducting a gradient descent search in weight space. It is found that the recognition performance is increased as the number of neurons in the hidden layer is increased. All samples were scaled to 64x64 pixel images first and then converted to binary images through threshold.

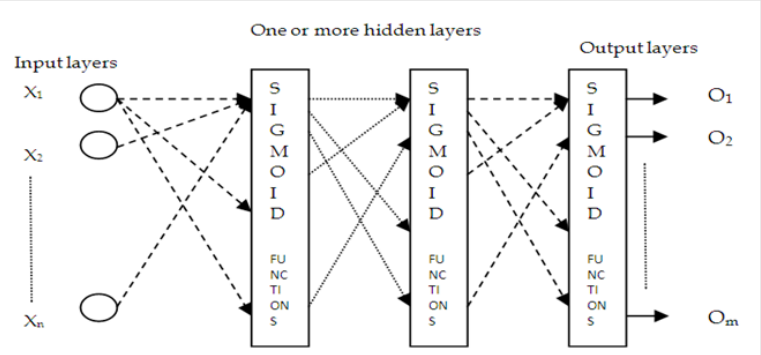


Figure 2.7: A block diagram of MLP shown as a feed forward neural network

2.10.3 Kohenen Neural Network (KNN)

KNN differs from the feed forward back propagation neural network architectures [12]. The first difference is that the Kohonen network in Figure-2.8, does not contain hidden layers; secondly, training and recognition processes are significantly different, that is, it is trained in unsupervised mode; thirdly, it does not use an activation function, and finally, the Kohonen

Network does not use any bias weight in the network. For a particular feature vector of a given pattern, a single neuron will be fired. Input data is first resized to 250x250 pixels, regardless of whether the input image is a single word or character. Skew detection was not taken into consideration. Both computer generated image and scanned image have been used to train the network. A character is converted into a vector of length 625, which also determines the number of input neurons.

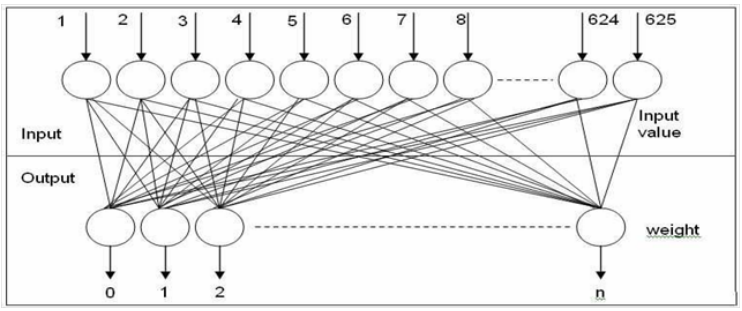


Figure 2.8: Kohenen Neural network Design for Bangla Character Recognition

CHAPTER 3

FEATURE SELECTION

3.1 What is Feature Extraction?

Feature selection is the main focusing point of our thesis work. In feature extraction stage, each character is represented as a feature vector, which becomes its identity. The major goal of feature extraction is to extract a set of features, which maximizes the recognition rate with the least amount of elements and to generate similar feature set for variety of instances of the same symbol. Due to the nature of handwriting with its high degree of variability and imprecision obtaining these features, is a difficult task. Feature extraction methods analyze the input document image and select a set of features that uniquely identifies and classifies the character. Considering the complexity of the problem, we try to give an overview about different feature selection algorithms.

3.2 Types of Features

There are many types of features. This part is the only component of an OCR which has to deal with the specific peculiarities of the shape of each character. So naturally authors have suggested many different features in their corresponding works. Among all the papers on both printed and handwritten characters, the major features identified are curvature based stroke features (Dutta & Chaudhury, 1993), linear stroke features (Chaudhuri & Pal, 1998), shadow, longest run (Das N., Das B., Sarkar, Basu, Kundu & Nasipuri, 2010), quad tree based center of gravity (Das N. et al., 2010), chain code (Alam & Kashem, 2010) , curvelet coefficient (Dutta & Chaudhury, 1993), structural or topological feature (Dutta & Chaudhury, 1993), fuzzy feature (Hoque & Rahman, 2007), the binary image itself, etc. Some of the features have been used for hand written scripts, while some others have been used for printed characters. Here a compilation of most of the features is given.

3.2.1 Stroke Features (Curvature Based)

A stroke is a set of dark pixels such that for all except two of its members there are two dark neighbors from among the members of the set itself [13].

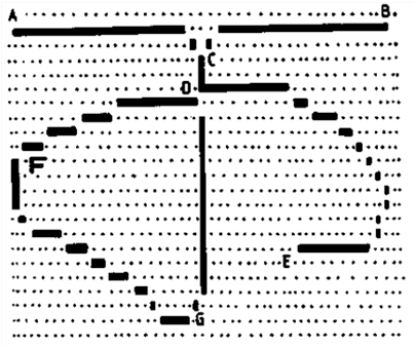


Figure 3.1: DE, AC, BC are segments; ACB, DE are strokes; DEG is a loop; C and D are junctions

A stroke consists of one or more segments. A segment is also a set of dark pixels. Except two pixels, all the other pixels have two dark neighbors from the same set. Of those two pixels, at least one has a junction. There will be no other junction pixel. A junction is a dark pixel, which has at least three dark 8-neighbors. A character may be represented in terms of the structural constraints imposed by junction points and the primitives/segments meeting at junctions. A stroke generates eight feature vectors. Values are-

1. **Number of points of curvature maxima:** If the curvature of a point along a stroke is greater (in magnitude) than that of its two immediate neighbors on both sides, the count of curvature maxima is increased by one.

2. **Number of points of curvature minima:** If the curvature of a point along a stroke is less (in magnitude) than that of its two immediate neighbors on both sides, the count of curvature minima is increased by one.

3. **Number of points of inflexion from -ve to +ve curvature:** For each pixel of a stroke, a count is incremented if its two predecessor points' curvature is non-negative and non-zero and two successor points' curvature is non-positive and non-zero.

4. **Number of points of inflexion from +ve to -ve curvature:** For each pixel of a stroke,

a count is incremented if its two predecessor points' curvature is non-positive and non-zero and two successor points' curvature is non-negative and non-zero.

5. Normalized positions with respect to the stroke-length for the points considered in 1. (Number of components = 4)

6. Normalized positions with respect to the stroke-length for the points considered in 2. (Number of components = 4)

7. Normalized positions with respect to the stroke-length for the points considered in 3. (Number of components = 4)

8. Normalized positions with respect to the stroke-length for the points considered in 4. (Number of components = 4)

3.2.2 Stroke Features (Mostly Linear)

Chaudhuri and Pal [4] have elaborated another way of identifying strokes from a character. These stroke features are mostly linear in structure. A total of 8 stroke features have been used here shown in the Figure-3.2. The information of existence or non-existence of these strokes is used in classification. These strokes are described below-

1. A horizontal continuous line over the character, known as matra line, and assumed to occupy 75% of character width.

2. A vertical continuous line assumed to occupy approximately 75% of the character middle zone.

3. A diagonal line along $+45^{\circ}$ with horizontal, occupies 40% of the height of middle zone.

4. A diagonal line in the lower half part of middle zone along 45° direction.

5. Existence of both stroke 3 and stroke 4.

6. Length of the arms is assumed to be 30% of the width of the middle zone of the character and the angle between them is 315°.

7. Stroke 7 is a cup-shaped feature where the bottom of the cup touches the base line.

8. A combination of stroke 1 and stroke 2, whose length is 40% of the height of the middle zone.

9. It is in the lower part of the middle zone.

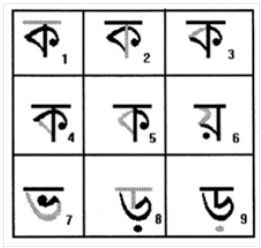


Figure 3.2: Stroke features used for character recognition (Shaded portions in the character represent the features)

3.2.3 Shadow Features

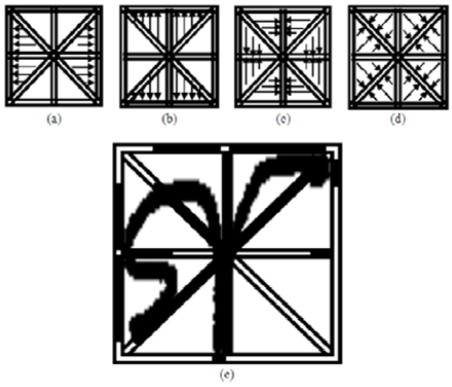


Figure 3.3: An illustration for shadow features. (a-d) Direction of fictitious light rays as assumed for taking the projection of an image segment on each side of all octants, (e) Projection of a sample image

Nibaran et al. [14] used shadow features in recognizing handwritten bangle basic and compound character. Here, an image box is divided into 8 octants as shown in Figure-3.3 . On each side of an octant, length of projection of the image is calculated. Thus 24 shadow

features are extracted from each digit image. These values are normalized by dividing the maximum possible length of projection on a particular side.

3.2.4 Longest Run Features

Along with Shadow features, Nibaran et al. [14] also used Longest Run Features. Within a rectangular image region of a character, longest run features are computed row wise, column wise and two major diagonal wise. Row-wise longest run feature corresponds to the sum of the lengths of the longest bars of consecutive dark pixels along each of all the rows of the region. In fitting a bar with a number of consecutive black pixels within a rectangular region, the bar may extend beyond the boundary of the region if the chain of black pixels is continued there. The three other longest run features within the rectangle are computed in the same way. Each of the longest run feature values is to be normalized by dividing it with the product of the height (h) and the width (w) of the entire image. The product, h $\times$ w, represents the sum of the lengths of the bars that fit consecutive black pixels individually in each of the four directions within the region completely filled with black pixels.

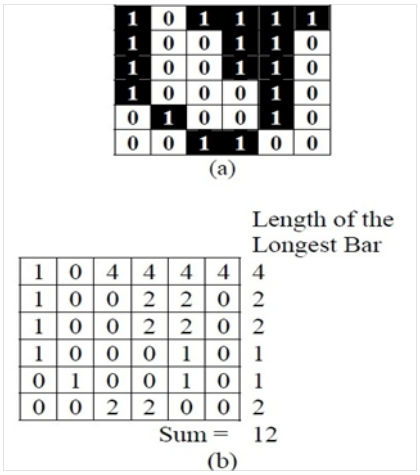


Figure 3.4: An illustration for computation of the row-wise longest run feature. (a) The portion of a binary image enclosed within a rectangular region, (b) Every pixel position in each row of the image is marked with the length of the longest bar that fits consecutive black pixels along the same row

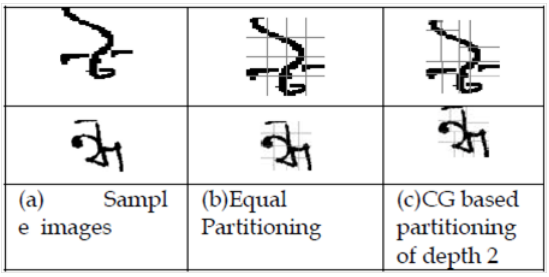
3.2.5 Quad-tree Based Features

A quad-tree is a tree data structure in which each node except the leaf nodes has up to four children. Quad-trees are most often used for representation of a two dimensional space by recursively subdividing it into four equal quadrants or regions. Here, partitioning a character pattern (or a subpart of it) into 4 regions is done by drawing a horizontal and a vertical line through the Centre of Gravity (CG) of black pixels in that region. If the depth of the quad-tree structure is d, then total number of sub images for each digit pattern at leaf nodes would be 4d. The coordinates of the CG of any image frame, (Cx, Cy), is calculated as follows:

$$C_x = \frac{1}{mn}\sum x f(x,y) \text{ and } C_y = \frac{1}{mn}\sum y f(x,y)$$

$$f(x,y) = \begin{cases} 1; \textit{for all black pixels} \\ 0; \textit{otherwise} \end{cases}$$

Where x and y are the pixel coordinates of an image of size m x n pixels. Both equal partitioning and CG based partitioning may be used to generate quad-tree.



(a) Sample images (b)Equal Partitioning (c)CG based partitioning of depth 2

Figure 3.5: An illustration of quad-tree based features

3.2.6 Structural and Topological Features

In recognizing Bangla numerals, Bhattacharya et al. [13] have used the topological feature set for handwritten Bengali numerals. Numerals are represented as graphs. Different parts or units of the graphs, such as, junction, terminal vertex, lowest vertex, lowest terminal vertex, open arm, right open arm, cycle volume, character height, cycle centroid and some more have been used in order to calculate the feature vectors.

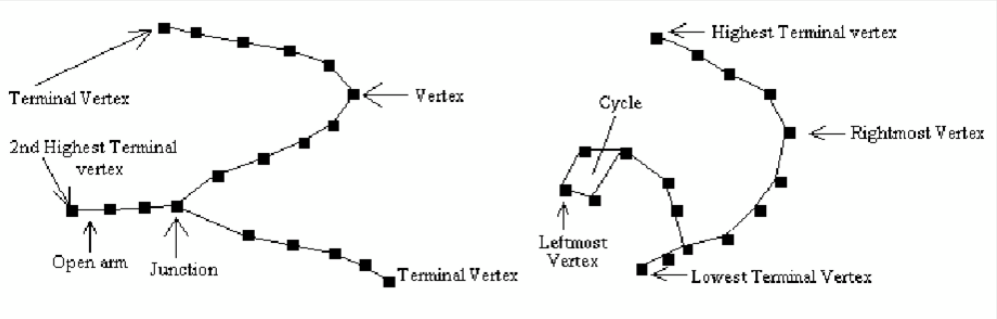


Figure 3.6: Structural features

3.2.7 Watershed Features

Pal and Chaudhuri [14] have used water-flow model from the concept of water overflow. They have used it to find features of hand-written numerals. The principle is, if we pour water from the above of numeral, the position where water is stored as reservoir, the shape of the reservoir as hole, etc are noticed in Figure-3.7. Feature vectors found from this model are-

1. Existence of holes and its number.
2. Position of holes with respect to its bounding box.
3. Ratio of hole length to height of the numerals.
4. Center of gravity of the holes.
5. Number of crossings in a particular region of the numeral.
6. Convexity of holes etc.

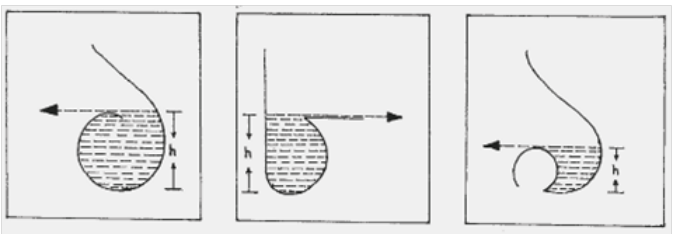


Figure 3.7: Feature using Water Flow Model

3.2.8 Chain Code

Mahmud et al. used chain code to extract feature for connected components of Bengali characters [15]. Each connected component has been divided into four regions indicating four quadrants in 2-D geometric system. There are several chain code convention used for image representation, but the most popular one is Freeman chain code. Freeman Chain Code is based on the observation that each pixel has eight neighborhood pixels.

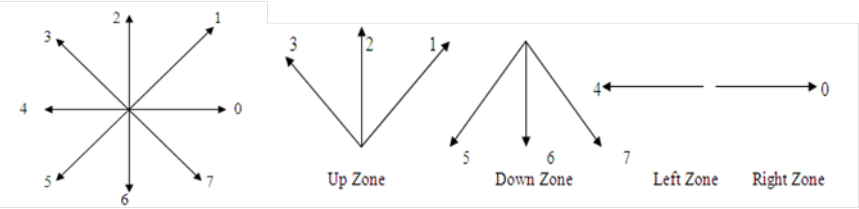
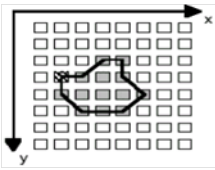


Figure 3.8: (a) Slope convention for Freeman Chain code, (b) 8 directional slopes divided into 4 direction zones for searching

A connected component is divided into 4 regions by a horizontal and a vertical line that go through the center of mass. In each zone, contour of the connected component is traversed. The frequency of each directional slope is counted. There are 8 directional slopes in a region, shown in Figure-3.8. As a result, in total, 32 directional slopes are found. These 32 values are the feature vectors, which are normalized by dividing them with square root of the sum of squares of all feature values. 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.

Figure-3.9, (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



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

Figure 3.9: (a) Chain code generation for an image, (b) Searching order in the four zones

an unvisited black pixel in the directional order: 3,4,5,6,7,0,1,2 (Searching order is shown in Figure-3.9, (b) for each zone). In this way the process continues and finally produces the chain code, 06700132454.

3.3 Optimal Feature Identification

Selection of features may be based on some of the points mentioned below-

Thinning independence - Thinning is a process which has been used by some author. The purpose of thinning is to create a character that contains a broader width for each stroke. Some feature values may become dependent on thinned or thickened character. It should be noted that a thinning independent feature may avoid the process of thinning, thereby, reducing the time required in the process of thinning.

Computational Complexity - This is another parameter which might be a matter of concern. For example, an algorithm that produces features at a cost of more time compared to another algorithm that produces a less qualified feature at an equivalent time period should be taken for consideration. Of course, the differential ability of those features also should be taken care of.

Usage during classification - Features should be chosen depending on their use in the classification scheme. For example, in the first phase of classification where only clusters are identified and where a decision tree can be the choice of tool, a feature capable enough to distinguish two clusters of similar characters, known as Existential Feature, can be used easily.

3.3.1 Choice of Features Having Values

Features that have numeric values can be used as input parameter for different types of classifier. However, no standard way of measuring the distinguishable capacity of these features have been proposed so far. In chapter four we performed some analysis on some of the features and showed their effectiveness. The features we have considered here are longest run features, quad-tree based features and shadow features.

CHAPTER 4

EXPERIMENTAL RESULT ANALYSIS

4.1 Image Dataset

Figure 4.1: Handwritten Bangla Vowels, Consonants, Numerals

4.2 Experimental Results for Longest Run Features

Longest run features are computed here row-wise within a rectangular image region of a character. It corresponds to the sum of the lengths of the longest bars of consecutive dark pixels along each of all the rows of the region. Each of the longest run feature values is to be normalized by dividing it with the product of the height (h) and the width (w) of the entire image.

Table 4.1: Longest Run Features (Bangla Vowels)

Samples	**Longest Sum (Row-wise)**	**Feature Vectors**
1	33	0 0 0 1 1 0 4 1 2 4 0 0 0 0 0 2 0 0 2 1 1 1 1 2 1 0 0 0 0 0 0 0
2	33	0 0 0 0 0 0 0 3 3 2 2 2 1 2 3 2 2 1 2 3 2 1 1 1 0 0 0 0 0 0 0 0
3	71	0 1 0 1 9 1 0 1 0 5 7 8 1 2 1 1 1 2 1 1 2 2 2 4 2 2 2 2 3 3 3 1
4	29	0 0 0 0 0 0 1 1 0 0 7 2 1 2 0 1 2 2 0 2 2 1 1 0 1 1 2 0 0 0 0 0
5	49	0 0 0 3 3 3 2 0 0 0 1 0 12 1 0 1 1 0 0 1 1 1 1 7 1 1 2 7 0 0 0 0
6	56	0 0 0 1 6 2 2 0 0 0 0 1 2 7 1 2 1 1 2 1 1 1 1 2 2 2 3 5 1 3 6 0
7	31	0 0 0 0 0 0 1 2 2 2 1 1 2 1 1 6 2 0 1 2 1 1 1 2 1 0 1 0 0 0 0 0
8	39	0 0 0 0 6 2 2 5 1 1 1 1 0 1 1 1 1 0 1 2 5 3 2 2 1 0 0 0 0 0 0 0
9	60	0 0 1 1 1 1 2 2 1 1 1 1 2 3 2 3 2 4 1 1 1 2 2 1 1 2 7 8 4 2 0 0
10	63	0 0 5 3 3 2 2 3 3 1 1 1 1 1 1 6 4 2 1 2 2 2 2 1 1 2 2 9 0 0 0 0
11	72	0 1 2 1 3 2 2 2 3 3 2 4 3 2 2 3 3 4 3 4 3 1 2 1 1 1 1 2 2 2 7 0

Table 4.2: Longest Run Features (Bangla Consonants (1-10))

Samples	**Longest Sum (Row-wise)**	**Feature Vectors**
1	94	0 0 0 0 0 0 8 18 4 3 3 2 2 2 3 3 3 3 3 4 4 4 4 3 4 3 3 4 3 1 0 0
2	65	0 0 0 0 0 0 3 5 8 2 2 2 2 2 5 5 4 3 2 1 1 4 5 5 2 1 1 0 0 0 0 0
3	71	0 0 0 0 0 5 8 3 3 3 3 3 3 4 3 2 3 1 2 2 2 3 10 3 2 2 1 0 0 0 0 0
4	61	0 0 0 0 0 0 2 5 6 2 2 2 2 3 2 1 1 1 2 2 2 4 2 3 3 3 3 4 3 1 0 0
5	83	0 1 2 2 2 2 2 2 1 2 5 2 2 3 3 7 3 2 2 2 1 2 3 7 3 2 2 2 3 8 3 0
6	77	0 0 0 0 14 16 1 1 1 1 1 1 2 2 2 2 3 4 4 2 2 2 1 2 1 1 1 2 2 2 4 0
7	99	0 0 0 5 25 3 2 2 2 2 2 1 1 1 5 3 2 2 2 2 3 3 5 2 3 3 4 5 4 3 0
8	73	0 0 0 0 18 4 2 2 2 2 2 3 2 2 4 2 3 3 2 7 2 1 1 1 1 1 1 1 2 1 0
9	80	0 0 0 0 0 0 0 7 6 9 2 3 3 4 4 4 6 3 3 5 4 3 4 3 4 3 0 0 0 0 0 0 0
10	55	0 0 0 0 0 0 0 0 0 0 0 0 8 4 2 1 1 3 2 1 1 2 12 4 3 3 8 0 0 0 0 0 0

Table 4.3: Longest Run Features (Bangla Numerals)

Samples	**Longest Sum (Row-wise)**	**Feature Vectors**
1	43	0 3 2 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 2 2 2 2 2 6 0
2	16	0 0 0 0 0 0 0 1 1 1 1 0 0 1 1 0 1 1 1 3 0 1 1 1 1 0 0 0 0 0 0 0
3	44	0 7 1 1 1 1 0 1 1 1 1 1 2 1 1 1 0 1 1 2 2 5 2 1 1 2 2 1 1 1 1 0
4	59	0 0 0 1 2 5 7 4 3 3 2 2 2 2 2 2 2 2 2 2 1 2 2 1 1 2 2 3 0 0 0 0
5	57	0 0 0 7 3 1 1 2 2 2 1 1 2 1 2 1 2 1 1 2 2 2 2 2 1 1 1 2 2 5 5 0
6	76	0 0 8 4 3 3 2 2 2 1 2 1 1 2 1 1 1 1 1 3 8 1 1 1 1 1 1 2 2 12 7 0
7	48	0 0 0 0 0 0 0 1 1 0 1 1 2 1 2 2 2 2 2 2 10 3 2 2 2 4 6 0 0 0 0 0
8	48	4 2 2 2 1 3 2 3 6 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
9	69	1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 8 21 1 1 1 2 1 2 1 2 2 2 2 1 2 2 2
10	59	0 1 1 1 1 1 2 3 3 3 2 2 2 1 1 1 1 1 1 0 1 1 2 2 2 3 5 3 4 4 3 1

4.3 Experimental Results for Quad-tree based Features

Quad-tree based features are calculated by dividing the image into four quadrants. Center of gravity for each quad is calculated by using the formula mentioned above in this paper. Finally, average of these values are calculated which are shown in the last column. The effectiveness of quadrature center of gravity has been analyzed below:

Table 4.4: Center of Gravity (Bangla Consonants (1-20))

Samples	1st Quad	2nd Quad	3rd Quad	4th Quad	Center of Gravity
1	(1.19,1.87)	(1.15,1.49)	(1.62,1.19)	(0.91,0.57)	(1.22,1.11)
2	(1.66,1.39)	(0.27,0.96)	(1.19,0.93)	(0.73,0.73)	(0.96,1.01)
3	(1.24,1.16)	(0.62,1.06)	(1.51,1.48)	(0.45,0.61)	(0.96,1.08)
4	(0.93,0.74)	(0.53,0.78)	(0.82,0.89)	(0.97,0.69)	(0.81,0.77)
5	(0.64,0.48)	(0.99,1.05)	(1.99,1.16)	(1.47,1.17)	(1.27,0.96)
6	(1.12,1.29)	(0.72,1.28)	(0.33,0.49)	(0.83,0.39)	(0.75,0.87)
7	(1.42,1.66)	(0.52,1.26)	(0.72,0.42)	(1.50,1.06)	(1.04,1.09)
8	(0.89,1.11)	(0.18,0.56)	(1.87,1.26)	(0.60,0.62)	(0.89,0.87)
9	(1.78,1.62)	(0.83,1.66)	(1.87,1.11)	(0.72,0.97)	(1.29,1.34)
10	(0.23,0.28)	(0.69,0.52)	(1.46,0.93)	(0.61,1.13)	(0.75,0.72)
11	(0.79,1.13)	(0.56,1.04)	(1.06,0.37)	(0.99,0.39)	(0.85,0.73)
12	(1.43,1.66)	(0.71,1.12)	(0.98,0.23)	(0.68,0.54)	(0.95,0.89)
13	(0.72,1.08)	(0.57,0.90)	(0.89,0.75)	(1.22,1.34)	(0.85,1.02)
14	(0.88,1.28)	(0.54,0.69)	(0.64,0.51)	(0.85,0.86)	(0.72,0.83)
15	(1.06,0.81)	(0.34,0.67)	(1.11,0.77)	(0.45,0.24)	(0.74,0.62)
16	(1.04,1.01)	(0.38,0.52)	(1.47,0.91)	(0.48,0.41)	(0.84,0.71)
17	(1.11,1.14)	(0.51,1.01)	(0.88,1.14)	(0.67,0.52)	(0.79,0.95)
18	(1.24,1.57)	(0.33,0.98)	(0.43,0.13)	(0.86,0.66)	(0.71,0.84)
19	(1.32,1.64)	(0.92,1.41)	(1.72,0.82)	(1.26,0.88)	(1.31,1.19)
20	(0.91,1.01)	(0.78,1.33)	(1.29,1.11)	(0.38,1.35)	(0.84,0.94)

Table 4.5: Center of Gravity (Bangla Vowels)

Samples	1st Quad	2nd Quad	3rd Quad	4th Quad	Center of Gravity
1	(0.30,0.67)	(0.07,0.11)	(0.52,0.44)	(0.20,0.36)	(0.27,0.39)
2	(0.26,0.12)	(0.14,0.42)	(1.32,1.01)	(0.27,0.59)	(0.50,0.54)
3	(0.61,0.90)	(0.32,0.49)	(1.23,0.75)	(0.90,0.52)	(0.76,0.67)
4	(0.42,0.55)	(0.07,0.16)	(0.60,0.24)	(0.38,0.52)	(0.37,0.37)
5	(0.56,0.77)	(0.43,0.65)	(0.59,0.21)	(0.66,0.34)	(0.27,0.49)
6	(0.03,0.09)	(0.88,0.86)	(0.93,0.71)	(0.98,0.78)	(0.71,0.61)
7	(1.03,0.58)	(0.29,0.61)	(0.12,0.07)	(0.21,0.15)	(0.41,0.35)
8	(0.00,0.00)	(0.34,0.41)	(0.81,1.28)	(0.29,0.54)	(0.36,0.56)
9	(0.00,0.00)	(0.85,0.71)	(1.41,1.04)	(1.06,1.28)	(0.83,0.76)
10	(0.56,1.03)	(0.63,0.88)	(1.21,0.65)	(0.69,0.65)	(0.77,0.81)
11	(0.46,0.73)	(0.55,0.66)	(1.65,1.06)	(1.02,0.94)	(0.92,0.85)

Table 4.6: Center of Gravity (Bangla Numerals)

Samples	1st Quad	2nd Quad	3rd Quad	4th Quad	Center of Gravity
1	(0.31,0.42)	(0.67,0.58)	(0.58,0.65)	(0.80,0.62)	(0.58,0.57)
2	(0.03,0.06)	(0.23,0.45)	(0.29,0.14)	(0.17,0.59)	(0.18,0.21)
3	(0.07,0.28)	(0.32,0.59)	(0.51,0.45)	(0.55,0.27)	(0.36,0.39)
4	(0.91,0.44)	(0.39,0.62)	(1.16,1.26)	(0.57,0.82)	(0.76,0.79)
5	(0.67,0.99)	(1.01,1.21)	(0.79,0.35)	(0.65,0.37)	(0.78,0.73)
6	(0.78,0.89)	(1.07,0.91)	(0.66,0.47)	(1.20,1.04)	(0.93,0.83)
7	(0.51,0.22)	(0.54,0.89)	(0.36,0.15)	(1.00,0.89)	(0.61,0.54)
8	(0.28,0.73)	(0.00,0.00)	(0.89,0.78)	(0.53,0.44)	(0.42,0.49)
9	(1.01,0.78)	(1.69,2.03)	(1.49,0.80)	(0.51,0.84)	(1.17,1.12)
10	(0.39,0.64)	(0.97,0.92)	(0.53,0.26)	(1.34,1.22)	(0.81,0.76)

4.4 Experimental Results for Shadow Features

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 for each character.

Bangla Vowels

Figure 4.2: Sample:1

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 1 1 1 1 0 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 1 0 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 0 1 0 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 1 1 0 0 1 0 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 1 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.3: Sample:2

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1
0 1 1 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 1 1 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 1 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 0 1 1 1 0
0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1 0 1 1 1 1 1 0 1
0 1 1 0 0 0 0 0 0 1 1 0 1 0 0 1 0 1 1 0 1 0 1 1
0 1 1 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 1 0 1 0 1 1
1 1 1 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 0 1 0 1 1
0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 1 1
0 0 0 0 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 1 0 1 1
1 0 0 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 0 0 0 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 0 1 1 0 0 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.4: Sample:3

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 0
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 1 1 0 1 0
0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 0 1 1 1 0 1 0
0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 1 0 0 1 1 0 1 0
0 1 1 0 0 0 0 0 0 1 1 0 0 1 1 0 1 0 0 1 1 0 1 1
0 1 1 0 0 0 0 0 1 1 1 1 0 1 1 0 1 0 0 1 0 0 1 1
0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 0 0 1 1
1 0 0 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 0 1 0 0 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 0 1 1 0 1 1 1
1 1 1 1 0 0 0 0 1 0 0 1 1 0 0 1 1 0 1 1 0 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1

Figure 4.5: Sample:4

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 1 0 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 1 1 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 0 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 0 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 0 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 0 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1
0 1 1 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 1 1 0 1 1 1
1 1 1 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.6: Sample:5

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 0 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 0 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 0 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 0 1 1 1 1 0 1
0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 1 1 1 0 0
0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 0 0 1 1 1 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 1 1 0 1 1 0 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 0 1 1
0 1 1 0 0 0 0 0 0 0 0 0 1 0 0 1 0 1 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 0 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 1 1 0 0 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

Bangla Consonants

Figure 4.7: Sample:6

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 1 1 0 0 1 1 0 0 1 1 0 1 1 1 0 1 1 1 1
0 0 0 0 1 1 1 1 0 1 1 0 1 1 1 1 1 1 0 0 1 1 1 1
0 0 0 0 1 1 1 1 0 1 1 0 1 0 0 1 1 1 0 1 1 0 0 1
0 0 0 0 1 1 1 1 0 1 1 0 1 0 0 1 1 1 0 1 1 0 0 1
0 1 1 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 0 0 0 0 0 0
0 1 1 0 0 0 0 0 1 1 1 1 1 0 0 1 1 0 1 1 0 0 0 1
0 1 1 0 0 0 0 0 1 0 0 1 1 0 0 1 1 0 1 1 0 1 1 1
0 1 1 0 0 0 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 1 1 1
1 1 1 1 0 0 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 0 1 1
1 0 0 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 0 0 1 1
1 0 0 1 0 0 0 0 1 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1
1 0 0 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.8: Sample:7

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 1 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 0 1 1 0 1
0 1 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 1 1 1 0 0
0 1 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 1 1 0 0 0
0 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 1 0 0 1
1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 0 1 0 0 1 0 0 1
1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 0 1
0 1 1 0 0 0 0 0 1 0 0 1 1 1 1 1 0 0 0 0 1 0 1 1
0 1 1 0 0 0 0 0 1 1 1 1 1 0 0 1 0 0 0 1 1 0 1 1
0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 0 1 1 0 0 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 1 1 1
0 1 1 0 0 0 0 0 1 0 0 1 0 0 0 0 1 0 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 1 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.9: Sample:8

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 1 1 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 1 1 0 0 1 1 0 1 0 0 1 1 1 0 1 1 1 0 0
0 1 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 1 0 1 1 0 1 0
0 1 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1 1 1 0 1 0 1 0
0 1 1 0 0 1 1 0 1 1 1 1 1 0 0 1 1 1 1 0 1 0 1 0
0 1 1 0 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 0 1 0 1 0
0 1 1 0 1 1 1 1 1 1 1 1 0 0 0 0 0 1 1 0 1 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 0 0 0 0 1 0 0 1
1 1 1 1 0 0 0 0 1 0 0 1 1 1 1 1 0 0 0 1 0 0 0 1
1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 1 0 0 1 1 0 0 1 1
1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 1 0 1 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 1 1 1 1 1 1 1
1 1 1 1 0 0 0 0 1 0 0 1 0 0 0 0 0 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.10: Sample:9

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 1
0 1 1 0 0 1 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1
0 1 1 0 0 1 1 0 0 0 0 0 0 1 1 0 1 1 1 0 1 0 0 1
0 1 1 0 1 1 1 1 0 0 0 0 0 1 1 0 1 1 0 0 1 0 0 0
0 1 1 0 1 1 1 1 0 0 0 0 0 1 1 0 1 1 0 0 1 0 1 0
0 1 1 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 1 0 0 0
1 1 1 1 0 0 0 0 0 1 1 0 1 0 0 1 0 0 0 1 1 0 0 1
1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 1 1 0 0 1 1 0 0 1
1 1 1 1 0 0 0 0 1 1 1 1 1 0 0 1 1 1 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 0 0 1 1 0 0 1 0 1 1 1 1 0 1 1
1 1 1 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 1 0 0 1 0 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.11: Sample:10

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 0 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 0 1 1 1 1
0 1 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 0 0 1 1 1
0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 0 1
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 0 1
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 1 0
0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 0 0 0 1 1
0 0 0 0 1 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 0 0 1 1
0 0 0 0 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 0 0 1 1
1 0 0 1 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 0 0 1 1
1 0 0 1 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1

Bangla Numerals

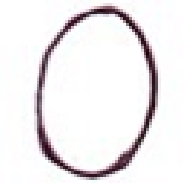


Figure 4.12: Sample:11

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 0 0 0 1 1
0 1 1 0 1 1 1 1 0 0 0 0 0 1 1 0 1 1 1 0 0 0 1 1
0 1 1 0 1 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0
1 0 0 1 1 1 1 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 1 0
0 0 0 0 1 1 1 1 0 0 0 0 1 0 0 1 0 1 1 1 1 1 1 1
0 0 0 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1

Figure 4.13: Sample:12

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 1 1 0 0 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 0 1 1 1 0
0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 1 0 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 0 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.14: Sample:13

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 1 1 1 0
0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 0 0 1 0 1 1 0
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 0 0 1 0 1 1 1
0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 1 1 0 0 1 0 0 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 0 1 1 0 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 0 1 1 0 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 0 1 1 0 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1

Figure 4.15: Sample:14

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
0 1 1 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 1 1 1
1 1 1 1 0 0 0 0 0 1 1 0 0 1 1 0 0 1 1 1 1 1 1 1
1 1 1 1 0 0 0 0 0 1 1 0 0 1 1 0 0 1 1 1 1 1 1 1
1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 0 0 0 1 1 1 1 0 0 1
1 1 1 1 0 0 0 0 1 1 1 1 0 1 1 0 0 1 1 1 0 0 0 0
1 0 0 1 0 0 0 0 1 1 1 1 0 1 1 0 0 1 0 0 0 0 0 0
1 0 0 1 0 0 0 0 1 1 1 1 0 1 1 0 0 1 0 0 0 0 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 1 1 0 0 1 0 0 0 1 1 1
1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 0 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 1 1 1 1
1 0 0 1 0 0 0 0 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

Figure 4.16: Sample:15

Shadow Feature Vectors
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 1 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0 1 1 1 0 1 1 1 1
0 0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 0 1 1 1 0
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 1 1 1 1 1 1 1 0
0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 1 1 1 0 1 1 1 0
0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 0 0 1 1 1 1 0
1 1 1 1 0 0 0 0 1 0 0 1 1 0 0 1 0 0 0 1 1 1 1 0
1 1 1 1 0 1 1 0 1 0 0 1 1 0 0 1 0 0 0 1 0 1 1 0
1 1 1 1 1 1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 0 1 1 0
1 1 1 1 1 1 1 1 1 0 0 1 1 0 0 1 0 1 1 1 0 0 1 0
1 1 1 1 1 1 1 1 1 0 0 1 1 0 0 1 1 1 1 1 1 0 1 1
1 1 1 1 1 1 1 1 1 0 0 1 1 0 0 1 0 1 1 1 1 0 1 1
1 1 1 1 1 1 1 1 1 0 0 1 1 0 0 1 0 1 1 1 1 0 1 1
1 1 1 1 1 1 1 1 1 0 0 1 1 0 0 1 1 0 1 1 0 0 1 1
0 0 0 0 1 0 0 1 1 0 0 1 0 0 0 0 1 1 1 1 1 1 1 1

4.5 Neural Network Testing Output

Neural Network have been trained with feature vectors resulted from feature extraction algorithm and tested with individual characters. Sample outputs of some characters are shown below:

Table 4.7: Test for 1^{st} Bangla Vowel

Training Sample	**Feature Set 1**	**Feature Set 2**	**Feature Set 3**	**Decider**
1	1.015342	0.788991	0.985174	1
2	0.124139	0.124130	-0.010840	0
3	0.102138	0.119756	-0.006071	0
4	0.001780	0.261951	0.013090	0
5	-0.048485	-0.775356	0.000452	0
6	0.058636	-0.607053	0.021240	0
7	-0.062347	-0.146250	0.005501	0
8	-0.092818	0.020854	0.008499	0
9	0.048418	0.083895	-0.001970	0
10	-0.047663	0.283794	0.008785	0
11	-0.002465	0.143186	0.006284	0

Table 4.8: Test for 1^{st} Bangla Consonant

Training Sample	Feature Set 1	Feature Set 2	Feature Set 3	Decider
1	0.993652	0.775908	1.009660	1
2	-0.064340	0.127957	-0.112392	0
3	0.044930	0.021575	-0.160095	0
4	0.030812	-0.021314	0.004965	0
5	0.119977	0.060678	0.109448	0
6	0.252205	-0.096063	0.107831	0
7	0.030292	-0.080495	0.142268	0
8	-0.079425	-0.029955	-0.135517	0
9	-0.078727	0.032368	-0.250091	0
10	-0.091589	-0.097475	-0.102950	0

Table 4.9: Test for 1^{st} Bangla Numeral

Training Sample	Feature Set 1	Feature Set 2	Feature Set3	Decider
1	1.015342	0.581498	1.003273	1
2	0.124139	-0.024589	0.196220	0
3	0.102138	0.510759	-0.108005	0
4	0.001780	0.367731	0.035329	0
5	-0.048485	-0.125310	0.010606	0
6	0.058636	0.147033	0.078735	0
7	-0.062347	0.262405	-0.003842	0
8	-0.092818	0.057164	-0.169264	0
9	0.048418	0.066272	0.062760	0
10	-0.047663	0.087971	0.089598	0

4.5.1 Performance Curve in Neural Network

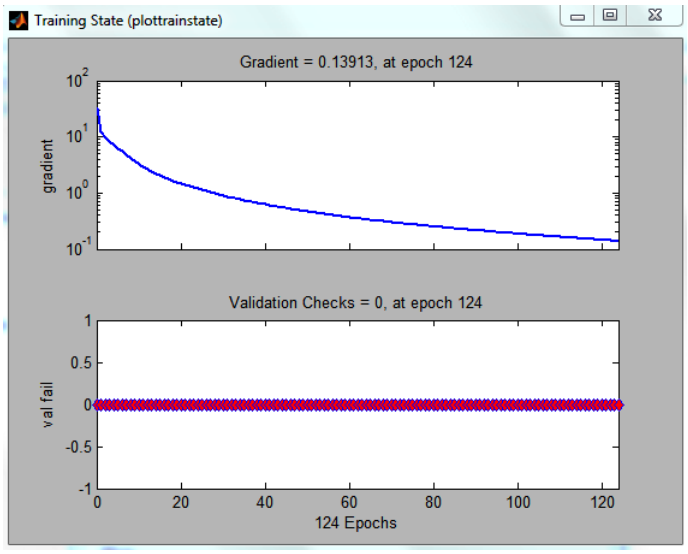


Figure 4.17: Training state curve

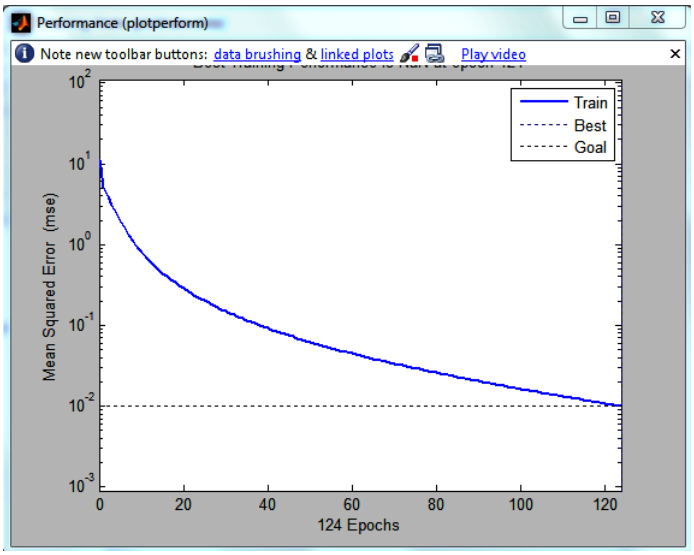


Figure 4.18: Curve based on Longest run feature set

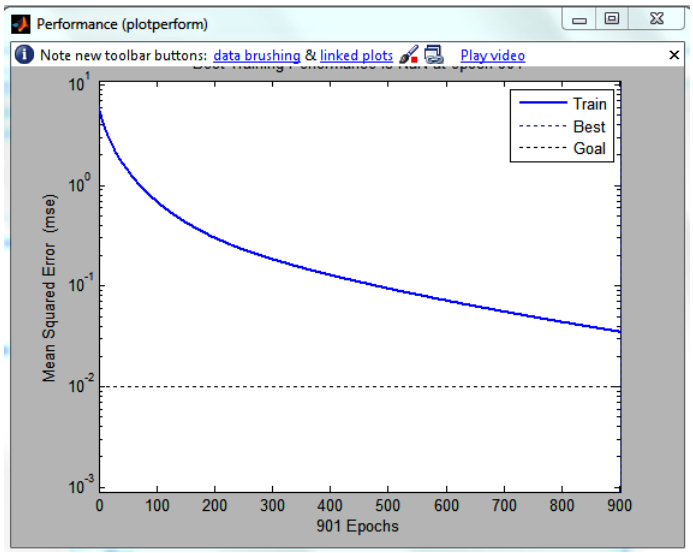


Figure 4.19: Curve based on Quad-tree feature set

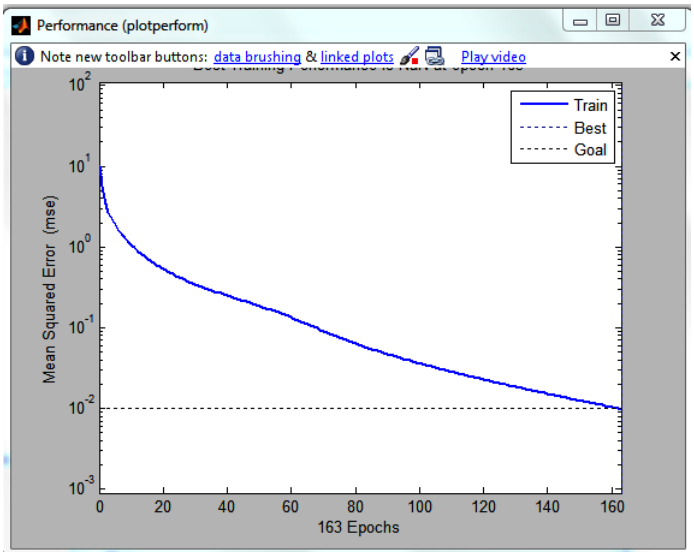


Figure 4.20: Curve based on Shadow feature set

CHAPTER 5

CONCLUSION

The pattern recognition opened the horizon of optical character reading in front of us which is now on use and immensely helping our day to day job at present. OCR software allows us to save a lot of time and effort when creating, processing and repurposing various documents. As being the 5^{th} most spoken language, it is now the demand of the time for Bangla OCR at commercial basis. The rich culture, literary works and much of the documents can be integrated with the flow of modern digital world easily by Bangla OCR. It would then be possible to increases the efficiency and effectiveness of our office work. This faster and cheaper way of digitizing would help our country to advance in a rapid way. Especially the greater accessibility would help all of us. If the accuracy can be ensured then it would decrease the amount of manual labour and time needed. We all will definitely be benefitted by that. The day is not very far when we will have a accurate Bangla OCR in our hand.

5.1 Limitations of the Research Work

We have extensively reviewed most of the features so far found in the literature . Tests of variability among feature values have been performed on some chosen features. Problems that we have identified are summarized below:

Little dataset are available as sample. So rigorous testing of an implementation is not possible. No compound characters have been considered. We have already understood that Bangla has not only basic characters; it is rich with modifiers and compound characters. Again lack of standard or benchmark samples do not allow us to make a comprehensive testing of our algorithm. This suggests that careful investigation would reveal the best possible combination of feature selection algorithms and processes for all the phases of Bangla OCR. It is evident that a full commercial OCR is a demand of the time in this era of digiti-

zation.

5.2 Recommendation for Further Research Work

- More dataset, depending on the availability, need to be collected as sample.
- Rigorous testing of an implementation should have been made.
- We have already understood that Bangla has not only basic characters; it is rich with modifiers and compound characters, so compound characters need to be considered more.
- Comprehensive testing of algorithm should be made basing on the availability of standard or benchmark samples.
- Careful investigation should be made to reveal the best possible combination of feature selection algorithms.

REFERENCES

[1] *http://en.wikipedia.org/wiki/Optical_character_recognition*, [Last visited: 8/11/2013].

[2] J. Mantas, "An overview of character recognition methodologies", *Pattern Recognition*, pp. 425-430, 1986.

[3] Md. Abdul Hasnat, Muttakinur Rahman Chowdhury, Mumit Khan, "Integrating Bangla script recognition support in Tesseract OCR", *Conference on Language & Technology* , 2009.

[4] B.B. Chaudhuri and U. Pal, "A Complete Printed Bangla OCR System", *Pattern Recognition*, vol. 31, no. 5, pp. 531-549, 1998.

[5] J.Bernsen, "Dynamic thresholding of gray-level images", *In proc. 8th Int'l Conf. on pattern recognition*, Paris, France, pp. 1251-1255, 1986.

[6] W.Niblack, "An Introduction to Digital Image Processing", Prentice Hall, pp. 115-116, 1986.

[7] T. V. Ashwin and P. S. Sastry, "A font and size-independent OCR system for printed Kannada documents using support vector machines", *Journal (Sadhana)*, vol. 27, 2002.

[8] Nasreen Akter, Saima Hossain, Md. Tajul Islam & Hasan Sarwar, "An Algorithm For Segmenting Modifies From Bangla Text", *ICCIT, IEEE*, Khulna, Bangladesh, pp. 177-182, 2008.

[9] M. E. Hoque, S. Lahiri, S. Sarkar, "Bangla Academy Byabaharik Bangla Abhidhan", Bangla Academi Press, Dhaka, Bangladesh, September 2003.

[10] B.B. Chaudhuri & U. Pal, "Skew Angel Detection Of Digitized Indian Scripts Documents", Transactions On Pattern Analysis And Machine Intelligence, *IEEE*, pp. 182-186, 1997.

[11] Subhadip Basu, Nibaran Das, Ram Sarkar, MahantapasKundu, Mita Nasipuri & DipakKumarBasu, "Handwritten 'Bangla' Alphabet Recognition Using an MLP Based Classifier", *NCCPB*, Bangladesh, pp. 285-291, 2005.

[12] Adnan Mohammad, Shoeb Shatil and Mumit Khan, "Minimally Segmenting High Performance Bangla Optical Character Recognition using Kohonen Network", Computer Science and Engineering, BRAC University, Dhaka, Bangladesh, 2007.

[13] Dutta, A. & Chaudhury, S., "Bengali alpha-numeric character recognition using curvature features", *Pattern Recognition*, 26(12), pp. 1757-1770, 1993.

[14] Das N., Das B., Sarkar, Basu, Kundu & Nasipuri, "Handwritten bangla basic and compound character recognition using MLP and SVM classifier", *Journal of Computing*, 2(2), pp. 109-115, 2010.

[15] Abdullah, A.B. M. & Rahman, A., "A Survey on Script Segmentation for Bangla OCR: An implementation Perspective", *Proc. of 6th International Conference on Computer and Information Technology (ICCIT)*, pp. 856-860, 2003.

[16] Alam, M. M. & Kashem, D. M. A., "A complete bangla OCR system for printed characters", *Journal of Computer and Information Technology*, 1(1), pp. 30-35, 2010.

[17] *http://www.ijera.com/papers/Vol3_issue1/EN31919926.pdf*, [Last visited: 7/12/2013].

[18] U.Pal & B.B Chaudhuri, "Computer Recognition of Printed Bangla Script", *International Journal of Systems Science*, vol. 26, pp. 2107-2123, 1995.

APPENDIX A