An Integrated Approach Towards Online Bangla Handwriting Recognition

Thesis submitted by

SHIBAPRASAD SEN

DOCTOR OF PHILOSOPHY(Engineering)

Department of Computer Science and Engineering,
Faculty Council of Engineering & Technology
Jadavpur University
Kolkata-700032,India

2018

JADAVPUR UNIVERSITY
KOLKATA-700032, INDIA

INDEX NO. 276/16/E

1. TITLE OF THE THESIS:

An Integrated Approach Towards Online Bangla Handwriting Recognition

2. NAME, DESIGNATION & INSTITUTION OF THE SUPERVISORS:

i. **DR. RAM SARKAR**
ASSOCIATE PROFESSOR,
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, JADAVPUR
UNIVERSITY KOLKATA-700032

ii. **DR. KAUSHIK ROY**
PROFESSOR,
DEPARTMENT OF COMPUTER SCIENCE
WEST BENGAL STATE
UNIVERSITY, BARASAT,
KOLKATA-700126

3. LIST OF PUBLICATIONS:

a. JOURNAL:

i. **Sen, S.**, Bhattacharyya, A., Singh, P. K., Sarkar, R., Roy, K., Doermann, D. (2018). "Application of structural and topological features to recognize online handwritten bangla characters. " ***ACM Transactions on Asian and Low-Resource Language Information Processing,*** 17(3):20, 1394-1404.

ii. **Sen, S.**, Chowdhury, S., Mitra, M., Schwenker, F., Sarkar, R., Roy, K. (2018). " A Novel Segmentation Technique for online Handwritten Bangla Words." **Pattern Recognition Letters,** 40(4): 809-818.

b. CONFERENCE:

i. **Sen, S.**, Shaoo, D., Sarkar, R., Roy, K. (2018). " DFA based

Online Bangla Character Recognition." In, International Conference on Information Technology and Applied Mathematics, pp. 668-673.

ii. **Sen, S.**, Shaoo, D., Paul, S., Sarkar, R., Roy, K. (2018). " Online Handwritten Bangla Character Recognition using CNN: A Deep Learning approach." In, 5th International Conference on Frontiers in Intelligent Computing: Theory and Application, pp. 674-679.

iii. **Sen, S.**, Mitra, M., Chowdhury, S., Sarkar, R., Roy, K. (2016). " Quad-tree based Image Segmentation and Feature Extraction to Recognize Online Handwritten Bangla Characters." In 7th IAPR TC3 Workshop on Artificial Neural Networks in Pattern Recognition, Ulm, Germany, pp. 837-844.

iv. **Sen, S.**, Sarkar, R., Roy, K. (2016). " Recognize Online Handwritten Bangla Characters using Hausdorff Distance based Feature." In 5th International Conference on Frontiers in Intelligent Computing: Theory and Application, pp. 541-549.

v. **Sen, S.**, Sarkar, R., Roy, K. (2016). " An Approach to Stroke-based Online Handwritten Bangla Character Recognition." In 3rd International Doctoral Symposium on Applied Computation and Security Systems, pp. 153-163.

vi. **Sen, S.**, Bhattacharyya, A., Das, A., Sarkar, R., Roy, K. (2016). " Design of Novel Feature Vector for Recognition of Online Handwritten Bangla Basic Characters." In 1st International Conference on Intelligent Computing & Communication, pp. 485-494.

vii. **Sen, S.**, Sarkar, R., Roy, K. (2015). " A Simple and Effective Technique for Online Handwritten Bangla Character Recognition." In 4th International Conference on Frontiers in Intelligent Computing: Theory and Application, pp. 201-209.

viii. **Sen, S.** , Paul, S. S., Sarkar, R., Roy, K., Das, N. (2015). " Analysis of different classifiers for On-line Bangla Character Recognition by Combining both Online and Offline Information." In proceedings of 2nd International Doctoral Symposium on Applied Computation and Security Systems.

4. LIST OF PATENTS: NONE
5. LIST OF PRESENTATIONS IN INTERNATIONAL/NATIONAL CONFERENCE:

i. "DFA based Online Bangla Character Recognition ". In, International Conference on Information Technology and Applied Mathematics, Haldia Institute of Technology, West Bengal, India.

ii. "An Approach to Stroke-based Online Handwritten Bangla Character Recognition ". In 3rd International Doctoral Symposium on Applied Computation and Security Systems, University of Calcutta, West Bengal, India.

iii. "A Simple and Effective Technique for Online Handwritten Bangla Character Recognition ". In 4th International Conference on Frontiers in Intelligent Computing: Theory and Application, NIT Durgapur, West Bengal, India.

Certificate from the Supervisors

*This is to certify that the thesis entitled "***An Integrated Approach Towards Online Bangla Handwriting Recognition***" submitted by Shri Shibaprasad Sen, who got his name registered on 3rd june 2016 for the award of Ph.D.(Engg.) degree of Jadavpur University is absolutely based upon his own work under the supervision of Dr. Ram Sarkar and Prof. Kaushik Roy and that neither his thesis nor any part of the thesis has been submitted for any degree/diploma or any other academic award any where before.*

1.......................................
(Dr. Ram Sarkar)

Signature of the Supervisor
and date with Office Seal

2.................................
(Dr. Kaushik Roy)

Signature of the Supervisor
and date with Office Seal

Dedicated to my parents

Acknowledgements

I am deeply indebted to many people who have contributed in making the completion of this dissertation work possible. Firstly, I would like to express my earnest gratitude and respect to my supervisors, Dr. Ram Sarkar and Prof. Kaushik Roy, for their guidance, constant support and monitoring throughout the course of this work. I would also like to express my sincere regards to Prof. Mita Nasipuri and Prof. Mahantapas Kundu, Department of Computer Science and Engineering, Jadavpur University, for providing me continuous inspiration, support and useful suggestions during this thesis work.

I am grateful to Prof. Sudip Naskar, Prof. Subhadip Basu, Jadvapur University and Prof. Rituparna Chaki, University of Calcutta for their valuable suggestions during preparation of my thesis.

I am also thankful to other faculty members of Department of Computer Science and Engineering, Jadavpur University and Department of Computer Science, West Bengal State University for their support during my course of work.

I would also like to thank Dr. Amitabha Biswas, Department of Bengali, West Bengal State University, for his help in designing the data collection form for my research.

In this research work, I had to collect lots of handwritten samples from different people. I am thankful to every individual who had contributed their handwriting voluntarily for preparation of my database used in this thesis.

I am thankful to my fellow researchers including Dr. Pawan kumar Singh, Mr. Samir Malakar, Mr. Neelotpal Chakraborty, Mr. Showmik Bhowmik, Mr. Suman Bera, Mrs. Shemim Begum of CMATER Laboratory Jadavpur University and Mrs. Sahana Das, Asst. Prof., Narula Institute of Technology; Mrs. Jaya Paul, Asst. Prof., Government College of Engineering and Leather Technology; Ms. Payel Rakshit, Asst. Prof., Maheshtala college; Mr. Sourav Ganguly, Ms. Ankita Dhar and Mr. Himadri Mukherjee of West Bengal State University, for their friendly support and encouragement during my research work.

Finally, I owe my encompassing debt to my respected parents, my wife Reshma Sen, daughter Anushka Sen and my niece Rimpa Dey, whose love and sacrifice, continued support, encouragement, good wishes and inspiration enabled me to complete this thesis.

Kolkata

Date: Shibaprasad Sen

Abstract

In this thesis, an effort has been made to recognize online handwritten Bangla characters and words. Due to unavailability of public database, the thesis work has started with developing both character and word database. Various feature extraction techniques and classification schemes are proposed here for character recognition following the holistic and stroke-based approaches. Going by the current trend, deep learning based procedure is also tried out for character recognition purpose. Segmentation of online handwritten Bangla words is successfully addressed and used as the first step of word recognition. Segmentation of words helps to find the constituent strokes of the corresponding word samples, which is one of the key contributions of this research work. Different feature extraction techniques (like distance based, point based, curvature based) are experimented for the recognition of segmented strokes obtained after applying word segmentation procedure. A Hidden Markov Model (HMM) based model is developed then to construct the word samples from recognized strokes by considering top-5 stroke recognition choices. Finally, a Language Model (LM) scheme is adopted to rectify the predictions generated by HMM to achieve the correct word choice. By following the above mentioned procedures, satisfactory results for character and word recognition have been achieved.

CONTENT

LIST OF FIGURES

LIST OF TABLES

1
Introduction

Handwriting recognition is considered as a technological revolution in the domain of man-machine interfaces especially because of handwriting has continued to persist as the most used means of communication and recording information in our day-to-day life. The Handwriting is static/off-line when using normal pen and paper. In counterpart, this is on-line mode when special pen or medium is used and the dynamics of the writing is available to machine. In off-line mode, the recognition is done from image(s) obtained by a scanner, digital camera, or by other digital input sources and using static pixels information. The image is binarized through some thresholding technique based on the colour pattern, so that the image pixels are stored as either 1 or 0. Then, features are extracted from the image and different classifiers are involved to classify the images based on the extracted features. The on-line scheme, deals with the recognition of handwritten text captured by a tablet PC, digital pen or a similar touch-sensitive device, and uses the digitized trace of the pen to recognize the symbols. Temporal information about how the symbols are formed is given including the number of strokes, the order of the strokes, the direction of writing for each stroke, and the speed of the writing within each stroke. A stroke is the pattern written from pen-down to pen-up moments [27]. Due to this dynamic information about writing, developing online handwriting recognition system performs better than its offline counterpart. Systems such as PDA, electronic pad and smart-phone have integrated online handwriting recognition capability for languages like Roman, Chinese etc. though none of them are equipped with system to recognize Indic languages.

With the increase in popularity of portable computing devices such as PDAs, smart phones, and pocket PCs, non-keyboard based methods for data entry are receiving more attention in the research communities and industrial sector. The most natural options are pen-based and voice-based inputs. One of the most promising option is on-line handwriting because of providing data in natural way for communication and can be stored easily. Digitizing devices like Smart boards and computing platforms such as the Iball Takenote, Digimemo, Wacom tablet, Trans Note and Tablet PCs have a pen-based user interface [17, 49, 58, 62]. Their main objective is to improve the interaction between man and machine while developing simple and ergonomic man-machine interfaces. Pen-based interfaces combined with automatic handwriting recognition seem to be the most promising and efficient solution. They offer a very easy and natural way to input since handwriting is one of the most familiar communication media and digital pen is often preferable to use instead of keyboard. Due to this reason, research on handwriting recognition has gained much attention from the researchers across the world. However, it is not a new research topic at this time; wide ranges of digitizer with different technologies are available in the market. Digital pen has been used as a human-computer interface years ago because of its flexibility in writing any kind of text. In fact, after more than 30 years of continuous and intensive effort devoted in solving the challenges of online handwriting recognition which can be defined as the ability of a computer to receive and interpret handwritten input, progress in recent years has been very promising [119].

Handwriting recognition can be performed at character, word, sentence and at document level. A character recognizer needs to be trained with sample characters only from the alphabet used in the language. The recognition module can follow holistic or stroke-based procedure. In holistic approach, features are extracted from the whole character and the extracted features are fed to the different classifiers for the recognition of the same. For stroke level approach, constituent strokes of the character are extracted first and then different features are extracted at stroke level. Constituent strokes are then recognized using generated features with the help of a suitable classifier. Characters are then formed from recognized strokes by implementing appropriate procedures.

Word recognizers are complex compared to character recognizer and can be designed using holistic or segmentation based approach. In holistic approach, words are recognized as a whole based on individual word model. Holistic approach is useful for small lexicon, but unable to recognize large lexicon accurately. In segmentation-based recognition scheme, word samples are segmented into valid component strokes (primitives) by adopting some suitable segmentation techniques. Features are then extracted from segmented strokes for recognition of the same. Finally, words are formed from recognized strokes by implementing suitable model.

1.1 Bangla Script: an Overview

Bangla, originated from an ancient Indic script Brahmi. It is used for Bangla (Bengali), Assameselanguages of India. Out of 23 constitutionally recognized languages in India, Bangla is the second most popular language in India, after Hindi written in Devanagari. The official and national language of Bangladesh is Modern Standard Bengali (Literary Bengali) [1–4]. It serves as the lingua franca of the nation, with 98% of Bangladeshis being fluent in Bengali (including dialects) as their first language. [5, 6] Within India, Bangla is the official language of the states of West Bengal, Tripura and the Barak Valley in the state of Assam. It is also spoken in different parts of the Brahmaputra valley of Assam. It is also the most widely spoken language in the Andaman and Nicobar Islands in the Bay of Bengal, and is spoken by significant minorities in other states including Jharkhand, Bihar, Mizoram, Meghalaya, and Odisha. With approximately 250–300 million total speakers worldwide, [55] Bengali is usually counted as the 7th most spoken native language in the world by population [7, 8].

Apart from that, Bangla script has a very rich and complex alphabet of more than 300 characters. The alphabet consists of 11 vowels and 39 consonants, collectively called Basic characters, 10 vowel Modifiers, 3 consonant Modifiers, one diacritic and nearly 269 Compound characters. Typical printed images of Bangla Basic characters are shown in Fig 1.1.1.

In Bangla script, a vowel preceded by a consonant takes a special shape called the vowel Modifier. Typical images of these Modifiers are shown in Table 1.1.1. With references to the corresponding vowels how these Modifiers look like when they appear with a sample consonant ক is also illustrated in the same table. The modified shapes can connect the consonant in left, right, base, top or combination of them. For example, the last two modifiers, as shown in the last two entries of Table 1.1.1, appear in two parts, separated by characters. Similar thing can be found for the information of compound characters. A few examples of Bangla words are shown in Table 1.1.2.

অ	আ	ই	ঈ	উ	ঊ	ঋ	এ	ঐ	ও	ঔ

ক	খ	গ	ঘ	ঙ	চ	ছ	জ	ঝ	ঞ
ট	ঠ	ড	ঢ	ণ	ত	থ	দ	ধ	ন
প	ফ	ব	ভ	ম	য	র	ল	শ	ষ
স	হ	ড়	ঢ়	য়	ৎ	◌ং	◌ঃ	◌ঁ	

Figure 1.1.1: Basic characters of Bangla alphabet

Table 1.1.1: Vowels, glyph and when used with consonant ক

Vowel	Glyph	Glyph used with consonant ক
আ	◌া	কা
ই	ি◌	কি
ঈ	◌ী	কী
উ	◌ু	কু
ঊ	◌ূ	কূ
ঋ	◌ৃ	কৃ
এ	ে◌	কে
ঐ	ৈ◌	কৈ
ও	ে◌া	কো
ঔ	ে◌ৗ	কৌ

Table 1.1.2: Example of some Bangla words

ডুমুর	দই	ঐরাবত	তরল	দমদম	কবি

Certain characters or Modifiers have an elongated portion crossing over their Matra, which are called ascendants, and certain characters, especially some Modifiers and the diacritic, appear below the base line of a character. These are in general called descendants. In 2nd column of 1st row of Table 1.1.2, a sample Bangla words appears with a diacritic as a descendant. Three alphabetic characters র, ড় and য় each have an isolated period. A sample Bangla word is shown in Fig. 1.1.2 with labelling for an ascendant, one descendant, and the common Matra of the word. Bangla script also consists of certain delimiters or punctuation marks like Dari (|), Coma (,), and Apostrophes (') etc., which are not considered for this work. For simplicity of readers the phonetic composition of all the basic characters of Bangla alphabet with Unicode information is shown in Table 1.1.3, while the phonetic encoding of few Bangla words are shown in Table 1.1.4

Table 1.1.3: Phonetic encoding of basic characters of Bangla alphabet

Bangla Character	Phonetic code	Unicode	Bangla Character	Phonetic code	Unicode
অ	A	\u0985	ন	Na	\u09A8
আ	AA	\u0986	প	Pa	\u09AA
ই	I	\u0987	ফ	Pha	\u09AB
ঈ	Ī	\u0988	ব	Ba	\u09AC
উ	U	\u0989	ভ	Bha	\u09AD
ঊ	Ū	\u098A	ম	Ma	\u09AE
ঋ	R	\u098B	য	Ya	\u09AF
এ	Ē	\u098F	র	Ra	\u09B0
ঐ	Ai	\u0990	ল	La	\u09B2
ও	Ō	\u0993	শ	Śa	\u09B6
ঔ	Au	\u0994	ষ	SSA	\u09B7
ক	Ka	\u0995	স	Sa	\u09B8
খ	Kha	\u0996	হ	Ha	\u09B9
গ	Ga	\u0997	ড়	RRA	\u09DC
ঘ	Gha	\u0998	ঢ়	DDHA	\u09A2
ঙ	NGA	\u0999	য়	YYA	\u09DF
চ	Ca	\u099A	ৎ	T	\u09CE
ছ	Cha	\u099B	◌ং	ANUSVARA	\u0982
জ	Ja	\u099C	◌ঃ	VISARGA	\u0983
ঝ	Jha	\u099D	◌ঁ	CHANDRABINDU	\u0981
ঞ	Ña	\u099E	◌া	SIGN AA	\u09BE
ট	TTA	\u099F	◌ি	SIGN I	\u09BF
ঠ	TTHA	\u09A0	◌ী	SIGN II	\u09C0
ড	DDA	\u09A1	◌ু	SIGN U	\u09C1
ঢ	DDHA	\u09A2	◌ূ	SIGN UU	\u09C2
ণ	NNA	\u09A3	◌ে	SIGN E	\u09C7
ত	Ta	\u09A4	◌ৈ	SIGN AI	\u09C8
থ	Tha	\u09A5	◌ো	SIGN O	\u09CB
দ	Da	\u09A6	◌ৌ	SIGN AU	\u09CC
ধ	Dha	\u09A7	◌্	HASANT	\u09CD

1.2 Online Handwriting Recognition

Online handwriting recognition implies automatic conversion of the text written by using digital pen, tablet PC etc. These devices accurately capture the x-y coordinate of pen-tip movement. The journey of on-line handwriting recognition started from late 1950's. This intense activity lasted through the 1960's ebbed in the 1970's, and was renewed in the 1980's. The renewed interest in on-line handwriting recognition stems from a number of factors. Like compared to the 1960's, recent time has seen more sophisticated electronic tablets, more compact and powerful computers, and better recognition algorithms. However, there are

Table 1.1.4: Phonetic encoding of few Bangla words

Bangla Word	Phonetic Composition	Bangla Word	Phonetic Composition
আম	Āma	ইমন	Imana
আজকাল	Ājakāla	জল	Jala
আমিষ	Āmisa	ঝালদা	Jhāladā
অনল	Anala	ঝুমুর	Jhumura
অরূপ	Arūpa	জীবিকা	Jībikā
বাবা	Bābā	কবি	Kabi
বাঘ	Bāgha	ঠিকানা	Thikānā
বাক	Bāka	থলে	Thalē
বালক	Bālaka	টিকিট	Tikita
বাঙুর	Bānura	উমা	Umā
বানী	Bānī	কৈখালি	Kaikhāli
বনৌষধী	Banausadhī	মাছি	Māchi
বারুদ	Bāruda	মিঞা	Miñā
ভয়	Bhaya	নিগূঢ়	Nigūrha
ভুল	Bhula	ঐরাবৎ	Airābata
বৈরাগ	Bairāga	ওল	Ōla
চামড়া	Cāmrrā	ঔষধ	Ausadha
ছেলে	Chēlē	পবন	Pabana
ছিরি	Chiri	ফল	Phala
ছবি	Chabi	ফুল	Phula
চোর	Cōra	রাজীব	Rājība
ডাব	Dāba	ঋষি	Rsi
ডুমুর	Dumura	শাখা	Śākhā
ঢাক	Dhāka	শৃগাল	Srgāla
দই	Da'i	তবলা	Tabalā
দমদম	Damadama	তৈল	Taila
এগরা	Ēgarā	টাকা	Tākā
এটেঁল	Ētēla	তরল	Tarala
গাছ	Gācha	থালা	Thālā
ঘোল	Ghōla	ঠাসা	Thāsā
ঘোষ	Ghōsa	যাযাবর	Yāyābara
ঘৃনা	Ghrnā	ঊষা	Ūsā
গৃহ	Grha	জনক	Janaka
গুহ	Guha	সং	Sam
হাবড়া	Hābarā	দৃশ্‌য	Drsya
ঈর্ষা	Īrsā	পে্‌রনা	Prēranā

additional and perhaps more important reasons to work with on-line mode. First, the advancement of recent hardware, combining tablets and flat displays bring input and output into the same surface. This combination permits the use of electronic ink, providing immediate feedback to the writer of the digitized writing. Electronic ink is the instantaneous display of the trace of the motion of the stylus tip. Second, efforts in automating office work have increased interest in more natural methods of entering data into machines. Third, people know more about user-

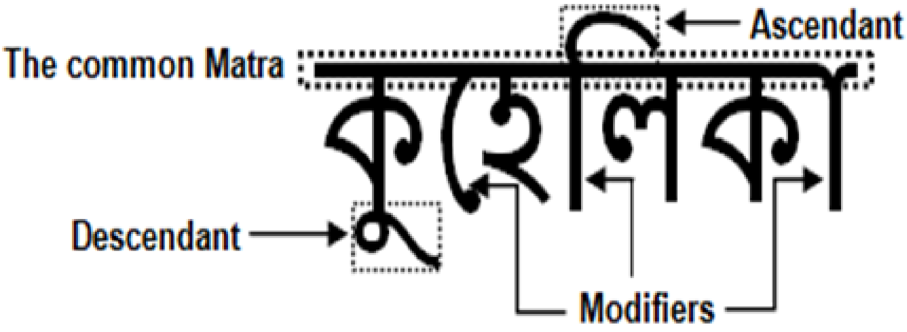


Figure 1.1.2: The common Matra, an Ascendant and one Descendant, shown in a sample Bangla word image [94]

interface design, particularly about issues of usability and user friendliness. Finally, researchers can now clearly understand the applications appropriate for handwriting recognition. For preparing a first draft and concentrating on content creation, pencil and paper are often favoured over the keyboard. Handwriting recognition offers the same advantage. In contrast, for transcription (copying text into the machine), the keyboard is faster than handwriting for small-alphabet languages, like English. However, for large-alphabet languages, like Bangla, Chinese, keyboards are cumbersome [27]. Other important uses of handwriting recognition are editing, annotating, and some applications that are heavily interactive and use direct pointing and manipulation. Tablets are also a powerful tool for input of sketches and drawings since they can accept both writing and graphics. Few research work related to both off-line and on-line handwriting recognition, cursive script recognition, and the recognition of machine-printed as well as handwritten characters are available in [16, 52, 63, 66, 77, 115].

1.2.1 On-line versus Off-line

Handwriting recognition technologies are devised focusing on either online or offline texts. For online handwriting recognition, a special digital pen is required to write on an electronically sensitive surface, such as digitizer combined with a liquid crystal display. As the pen is moved on the surface, the 2D coordinates of the successive points on the trajectory of the pen tip are recorded as a function of time. With this, the pressure, velocity, angle, acceleration of the pen tip and also the stroke information are available for of each character as soon as it is written. Temporal information about how the symbol is formed is given including the number of strokes, order of the strokes, direction of the writing for each stroke, and the speed of the writing of each stroke. Nowadays, certain mobile phones provide the facility of writing online text messages by moving a stylus on its display. A Take Note and a mobile phone capturing images of handwritten Bangla text are shown in Fig 1.2.1 (a-b).

As opposed to this, for offline handwriting recognition, a page of text is optically scanned to form an image. This image is passed through phases, such as text graphics separation, text line and word extraction, character segmentation and recognition, and character code generation. So the problems of offline handwriting recognition are more diverse than its online counterpart. To make the problem simpler, nowadays Take Note like devices with specially designed pens and paper holder boards are used. In such devices, the specially designed pen is used to sample pen tip positions on the paper at regular time intervals to store an image file of the handwriting, created with the pen movements in an on-board memory unit. The image file of handwriting so created may later be transferred into a computer through an USB port for further processing. Mobile phones with cameras can nowadays be used as a substitute for an optical scanner to capture images of text documents with or without graphics.

1.2.2 Why On-line?

Since few decades, with the advancement of handwriting recognition technology, applications are becoming more challenging as well as interesting [70, 78]. For example, optical character recognition (OCR) is becoming an integral part of document scanners, and is used in many applications such as mail sorting, security (signature verification, for instance) and language identification.

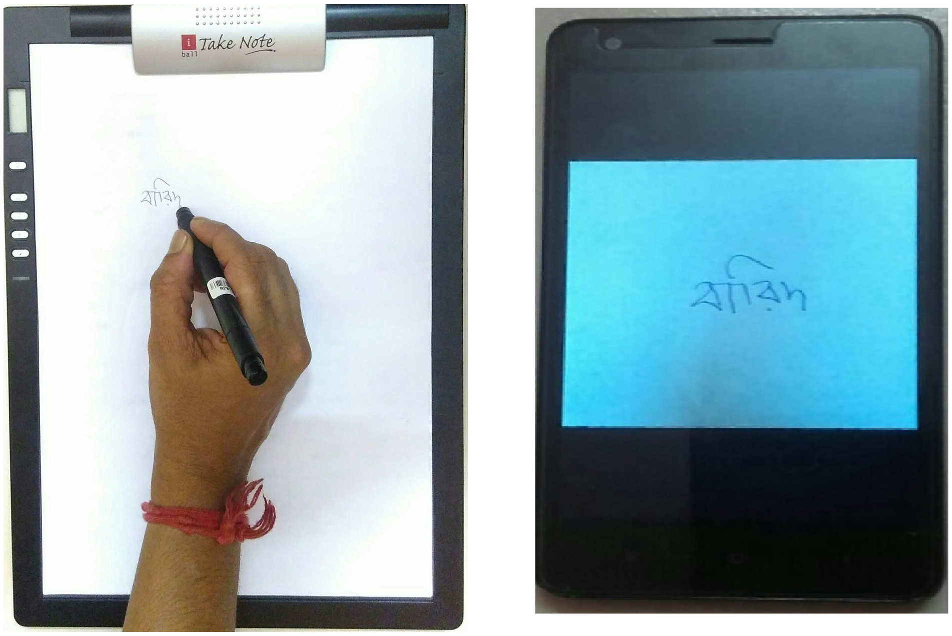


Figure 1.2.1: A take note and a mobile phone capturing images of Bangla text

State-of-the-art of handwritten character recognition reveals that off-line handwriting recognition yields less classification rate compared to online handwriting recognition [72, 78]. On-line data also offers significant reduction in memory and therefore reduce space complexity. Another advantage is that the digital pen on a tablet device immediately transforms the handwriting into a digital representation that can be reused later without having any risk of degradation usually associated with conventional paper-based handwriting. Historically, pen computing (defined as a computer system employing a user-interface using a pointing device plus handwriting recognition as the primary means for interactive user input) defines the use of a mouse and graphical display by at least two decades, starting with the Stylator Dimond [36] and RAND tablet Groner [48] systems of the 1950s and early 1960s. Even, one can cite a few examples [28, 34, 43, 125] where they mainly focus on temporal information as well as writing order recovery from handwriting image. This is why on-line handwriting recognition systems can provide interesting results.

On-line character recognition involves the automatic conversion of stroke as it is written on a special digitizer or PDA, where a sensor picks up the pen-tip movements as well as pen-up/pen-down switching. Such data is known as digital ink and can be regarded as a dynamic representation of handwriting. The obtained signal is converted into any standard encoding format which are understandable by computer and text-processing applications like MS-Office. Fig. 1.2.2 reflects the on-line stroke sequences in the form of 2D (x, y) coordinates. In this figure, initial pen-tip position is coloured in red and pen-up (final point) is coloured in blue. Normally the elements of an on-line handwriting recognition interface typically include:

- a pen or stylus for the user to write with.
- a touch sensitive surface, which may be integrated with, or adjacent to, an output display.
- a software application i.e., a recognizer which interprets the movements of the stylus across the writing surface, translating the resulting strokes into digital character.

In a broader way, it resembles one of the applications of pen computing [119] i.e., computer user-interface using a pen (or stylus) and tablet, rather than devices such as a keyboard, joysticks or a mouse. Pen computing can be extended to the usage of mobile devices such as wireless tablet personal computers, PDAs and GPS receivers.

1.2.3 Digitizer technology

The concept of a pen computer was first proposed by Kay in 1968 [97]. Since then many innovations and improvements in handheld technology have been occurred. To compare available digitizers, different characteristics can be used, such as resolution, accuracy, and sampling rate which typically varies from 50 to 200Hz

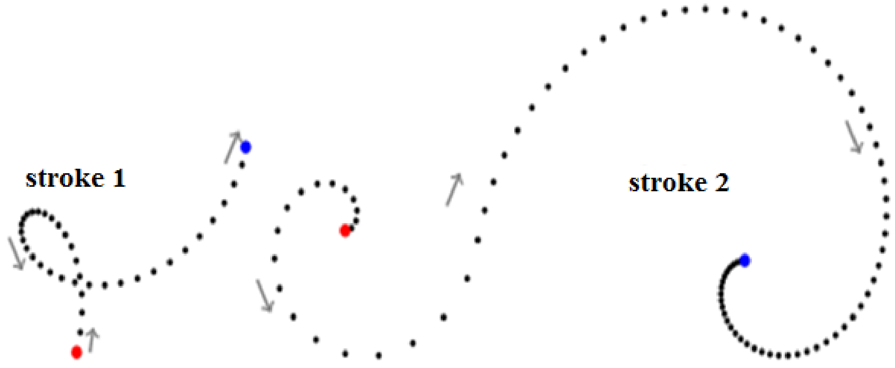


Figure 1.2.2: On-line stroke sequences in the form of 2D (x, y) coordinates. In this illustration, initial pen-tip position is red coloured and pen-up (final point) is coloured in blue

depending on the application. One of the interesting technologies is the combination of the digitizing tablet with the display screen, providing high level of interactivity similar to that of drawing or writing using the usual pen and paper. Resistive and inductive pen-sensing technologies are the most widely used touch screen technologies, but each one has unique characteristics that can make it the preferred choice for certain mobile applications. Resistive technology [97] offers a fast, reasonably accurate, and affordable technology that recognizes the touch input from any stylus, finger, gloved hand, and pen/tool. Further, due to its availability and low cost, resistive technology is most used.

A tablet consists of a plastic or electronic pen and a pressure sensitive writing surface, on which the user scribes one's handwriting with the pen. When moving a pen, the digitizer is able to detect information like x and y coordinates of a point, the state of whether the pen touches the surface or not. The information is sent to the connected computer for recognition as shown in Fig. 1.2.3.

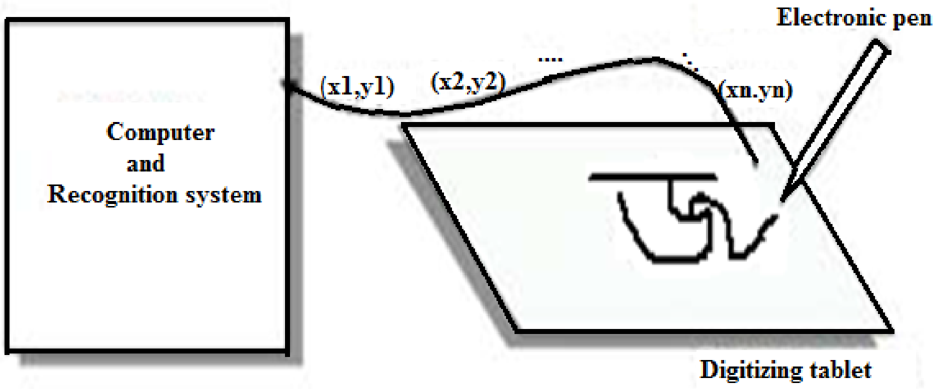


Figure 1.2.3: A tablet digitizer, input sampling and communication to the computer

1.2.4 Recognition procedure

Online handwriting recognition can be performed by any of the following approaches mentioned below:

- Holistic approach In this method handwritten characters/words are considered as a single unit and from therein features are computed using different strategies. These features are then fed to the different classifiers for the recognition of the same.

- Stroke based approach Any character or word is generally written by using more than one stroke. The features can be extracted at stroke level and the system can be trained accordingly for the recognition of strokes. However, if any stroke represents more than one symbol or part of symbol, then strokes are further segmented. Features are then computed from segmented strokes and fed to the classifier for classification. Final character/word recognition can be carried out using recognized strokes by adopting suitable model.

1.3 Challenges in on-line handwriting recognition

Various significant issues need to be considered to develop an OHR (Online Handwriting Recognition) system for Bangla script. Studying various existing online handwritten character and word recognition methods, it is observed that compared to the literature available in the said field for other scripts like English, Devanagari, Gurumukhi, etc., the volume of work on Bangla is relatively very less. In a multi-lingual, multi-script country like India, the amount of research on online handwritten character/word recognition gets divided among the various Indic scripts. On top of that, Bangla script, itself is complex in nature which has various simple, complex and compound character formations. So, automatic text level analysis on the same is an adequately challenging task. The lack of proper database on Bangla handwriting is also an major area of concern. Preparation of a handwriting database is a challenging, tedious and time consuming job. The variations in number and order of strokes to represent the same character is an another important concern mentioned in Fig. 1.3.1 (a-b) [82]. From the Fig. 1.3.1(a), it is noticed that different number of strokes may be involved to represent the same character sample. Fig. 1.3.1(b) highlights the different possible combinations of stroke sequence to represent the same character when considering a particular case. Hence, when considered all the cases, it can be easily guessed that different number of stroke sequence combinations are possible to represent the same character. Fig. 1.3.2 highlights the situation where a stroke can appear in different positions to represent different meaning. For example, in the first word (in আম) this stroke helps to form the modifier া, in the second word (চাষ) the same shape helps to form the character ষ from character য. Again the absence of this symbol may represent different valid character য. In the third word, the appearance of this shape puts more stress on the pronunciation of last character ত. The presence of this symbol as ascendant over ব in the fourth word signifies presence of র in front of ব (e.g. গরব্). Hence, all these make the recognition process a challenging task. Writing speed variation also adds extra complexity for online handwriting recognition. Due to speed variation, same information (at stroke/character/word level) contains different number of pixel points. Writing with lesser speed contributes more pixels for the same information. Fig. 1.3.3 (a-b) illustrates three different scenarios, where the same character contains different number of pixels due to writing variation.

Handwriting of each individual may vary at different times but each writer has some kind of uniqueness in their writing pattern which enables experts to analyze and distinguish writers. This variation of writing is of two types: intra-writer variation (the variation within a person's own handwriting samples) which is less challenging than that of the inter-writer variation (the variation between the handwriting samples of two different people)[50]. Therefore, during analysis of writing, two main points of concern are identification of inter-writer variation and at the same time minimizing the effect of intra-writer variation. These two variations for Bangla handwriting can be observed in Fig. 1.3.4 (a-b). In (a), five isolated Bangla characters from five different writers are given and in (b), five instances of five characters among these characters from a single writer is presented. Observing these, it can be said that the character 'গ' (g) has more inter-writer variability than other characters. From (a) and (b), one can also infer that the variability differs from character to character. For example, REF (/) has little inter-writer or intra-writer variation but character 'গ' (g) has more variation when written by different writers compared to when written by the same writer.

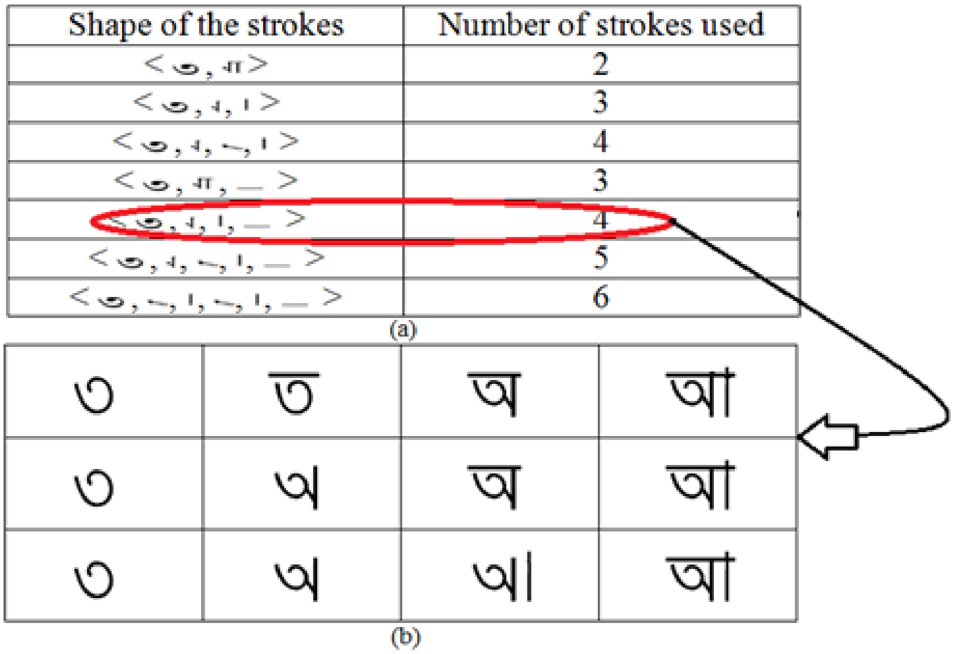


Figure 1.3.1: Different ways of writing 'আ' using some of the possible strokes in some possible combinations with different stroke-orders for a character having four strokes

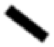
Stroke	Position with respect to character	Word sample containing the stroke
	In top position	আম
	In middle position	চাষ
	In bottom position	হটাত্
	In upper position	গর্ব

Figure 1.3.2: An example of complex case where same stroke lies in different positions in different words

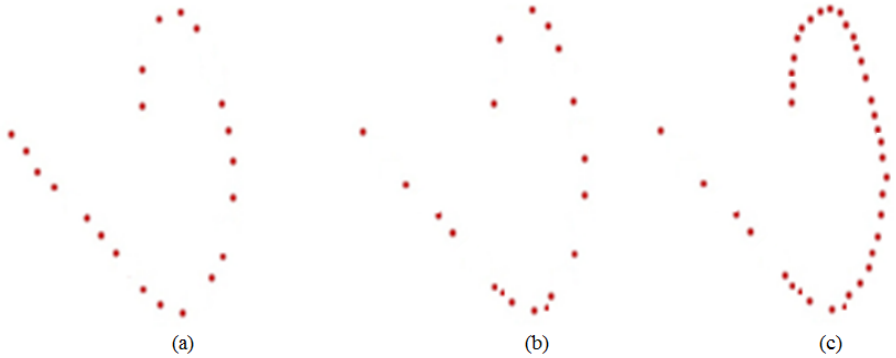


Figure 1.3.3: Same character having different numbers of pixels to form it

Word recognition systems are complex compared to character recognizer. Word recognition can be done by using either holistic or segmentation based approaches. In holistic recognition, words are recognized as a whole unit. Suitable features are extracted from this unit and classification is performed using these feature values. Holistic approach is very useful for small lexicon (i.e. limited number of words), but unable to recognize large lexicon accurately. On the other hand, for segmentation-based on-line word recognition scheme, samples are segmented into primitives by adopting some suitable segmentation technique. Features are then extracted from the segmented strokes and sent to the recognition module for the recognition of segmented strokes. Finally, words are formed from the recognized strokes by implementing some model. However, good word recognition score is heavily dependent on applied segmentation algorithm. In other words, good recognition result can be obtained if the segmentation algorithm can minimize over and under segmentation issues. The key problems of segmentation are the variation of writing style of different individuals, presence of skew, slant, no standard spacing between text lines, words and characters.

1.4 Motivation

Many trivial as well as non-trivial challenges related to online Bangla character and word recognition system have been played as the prime motivations for develop-

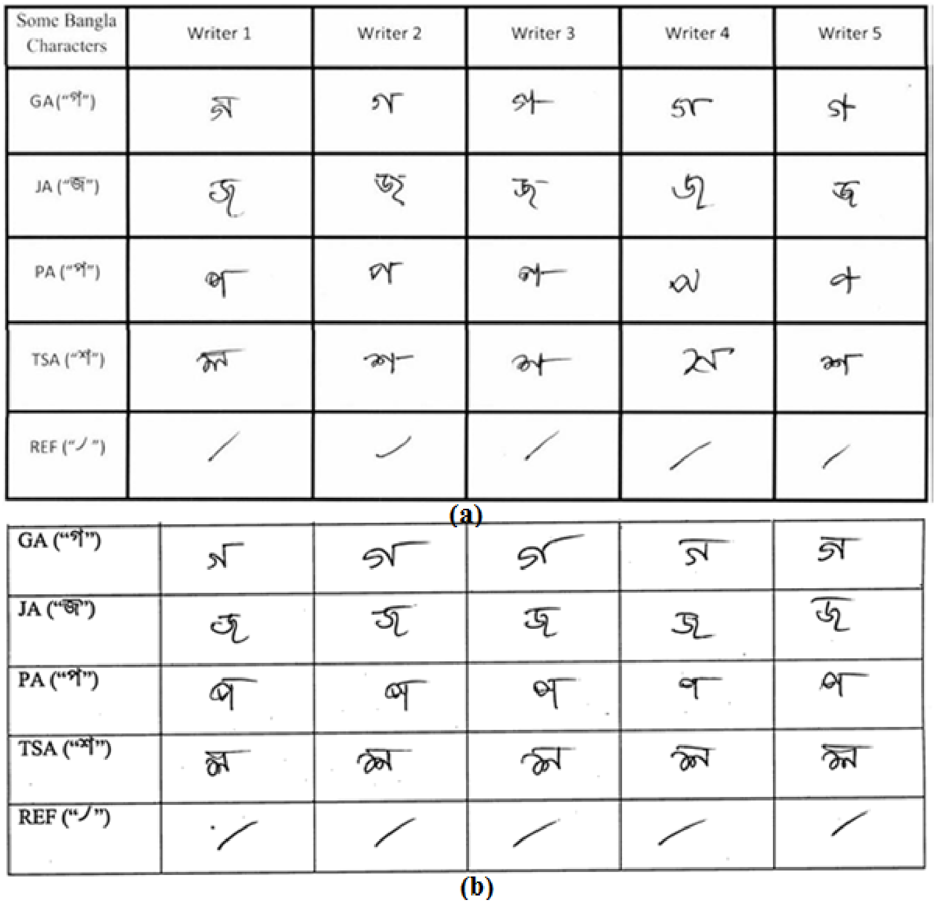


Figure 1.3.4: Intra-variability and inter-variability of Bangla handwriting are shown for some selected characters (a) from five different writers, (b) from a single writer [50]

ment of the same. Apart from the challenges there are some others aspects too which have inspired to work under this research area. These aspects can be formally pointed out as follows:

- Being a Bangla speaker, it motivates me to contribute something for my mother tounge. I plan to develop an online Bangla character/word recognition system which will have a great social impact especially in south Asia region.

- Despite the importance of Bangla language, there is a lack of complete system for online handwritten Bangla character/word recognition. In addition, the unavailability of standard online Bangla handwriting database for both isolated characters and words also motivates us to develop the same.

- The challenging nature of handwriting recognition has attracted the attention of researchers from both academic and industry circles. The huge part of these researches deals with Latin, Chinese, Gurumukhi, Devanagari scripts. Interest in Bangla script has developed years later, and so state-of-the-art for online Bangla handwriting recognition is less advanced.
- The absence of online writer and signature identification or verification system, that can be applied in Banking, crime control and other sectors, motivated to work under this domain.

1.5 Scope of the work

The primary objective of this thesis is to develop an automatic character/word recognition system for online Bangla handwriting with relatively high accuracy. The performance of any pattern recognition system depends on the pre-processing, feature extraction and classification strategies. Good pre-processing facilitates to produce good feature set. So the work is planned to concentrate on different pre-processing, feature extraction and classification stages in the form of either suggesting some the new approaches or modifying the available ones. An OHR system has to address issues such as variation in writing style of individuals at different times and also among different individuals (size, shape, variation in terms of number and order of constituent strokes, etc). These issues generally limit the accuracy of OHR systems. Due to these challenges, design of online OHR system has become an interesting research topic in the areas of image processing and pattern recognition. To evaluate performances of all such stages related to online handwritten Bangla character recognition, word segmentation and recognition, benchmark databases, which are not available till date, are necessary. Development of such databases has also been undertaken here.

The scope of the thesis work formally defined below:

The databases developed under this work can be partitioned into three broad categories, character database, word database and stroke database. Character database contains online handwritten characters taken from the form designed to collect isolated Bangla characters. Word database is developed by preparing two different types of forms. One type of form is used to collect 50 different word samples and second one is the online handwritten text documents. As stroke are not written individually, they are extracted from characters and words. In this work, the word database developed from online handwritten Bangla documents, is accompanied with the ground truth. To make the ground truth, words in each text line goes through a word segmentation module that helps to segment the word sample into valid components. To handle the errors generated by the word segmentation algorithm, a software module is developed to rectify such errors when detected through visual observation.

The developed stroke, character and word databases are considered for the experimentation and evaluation of different character and word recognition methodologies applied here. Holistic, stroke based and deep learning based approaches for character recognition are described in this current work. This work also includes segmentation based online handwritten word recognition procedure.

In holistic character recognition procedure, character samples first go through some pre-processing stages like size normalization, point normalization in order to overcome the challenges due to size and speed variation of writing. After preprocessing, there is a good scope to develop some powerful features that can differentiate the character patterns meticulously. Hence, under this work, some new and customized features are extracted from handwritten characters and fed to different classifiers for classification purpose.

For stroke based character recognition, strokes are collected from online handwritten character information. A character can be written by using a single or multiple strokes. Now, in stroke based procedure, features are extracted at stroke level and thus character recognition score is totally dependent on stroke classification results. Hence, there is a requirement to design more powerful features to enhance stroke recognition results and thereby improving overall character recognition performance. Under this work, after going through some pre-processing steps, handcrafted features are generated from constituent strokes of the character sample using different feature generation algorithms (like distance based, zone based path traversal) mentioned through [101, 105]. Extracted features are fed to the classifier for stroke classification purpose. The recognized strokes are then used to construct the character sample by implementing rule based and deterministic-finite-automata (DFA) based approaches. All the said pre-processing steps and feature extraction techniques are described in chapter 4.

In most of the cases, result of any pattern recognition problem depends on the strength of the generated feature vector. Again, designing of powerful handcrafted features is not only time consuming but also many a times it does not ensure to produce good result. Hence, as an alternative deep learning based approach can be applied where the generation of handcrafted features are not required, rather those are generated automatically. Clearly, this methodology can minimize both the overhead and time to design handcrafted features. Under this work, Convolutional Neural Network (CNN), a deep learning architecture has been used for character recognition purpose [106]. In this procedure, features are generated automatically in convolution layer. This layer involves arbitrary number of learnable filters to move along the width and height of the image to produce feature map. A filter can be considered as an array of numbers where the numbers are called weights or parameters. After sliding the filter over all the locations, activation map or feature map is generated. The last layer of CNN architecture is a fully connected network, used for character classification purpose.

The holistic recognition result of online handwritten Bangla characters by applying five types of features (distance based, hausdorff distance based, transition count, CG based circular with topological feature and few local features in terms of area, chord length, mass distribution) have been discussed in Chapter 5. This chapter includes the stroke classification and character recognition scores in stroke based character recognition schemes when adopted rule-based or DFA based approach.

In order to build a segmentation-based word recognition system, the design of an efficient word segmentation algorithm is primary requirement to find the valid components of the word sample. The work mentioned in [19, 20] have tried to generate valid components from the target word by estimating busy zone over it and then applying Down->Up->Down (DUD) approach for segmentation. Off-line and on-line information are combined to form busy zone over the word sample. Few rules are also developed and used when applying DUD within busy zone in order to achieve valid components for further processing. However, the procedure is time consuming as it considers both off-line and online information. It also suffers from under and over segmentation issues due to presence of multi directional skew and different writing variations. Hence, there is a clear scope to design a word segmentation algorithm that can efficiently handle such multi-directional skew, different writing variations and thus produce good segmentation result by minimizing under and over segmentation issues. Keeping this fact in mind, in this thesis work, an algorithm has been proposed for the segmentation of online handwritten Bangla word samples into valid components [104]. In this technique, constituent strokes of the word sample are classified into three categories based on their positional information; upper zone, lower zone and strokes-need-segmentation (SNS). A procedure has been proposed to form busy zone over SNS strokes to handle skew. Then a sub-zoning scheme within busy zone followed by a modified DUD concept has been used to find the segmentation points. The sub-zoning scheme helps to handle different writing variations. The advantage and technical details of the stroke-level segmentation technique have been reported elaborately in Chapter 6.

Handcrafted features are then extracted from the segmented strokes using distance based, point based and curvature based techniques. The generated features are then fed to the classifier for stroke recognition. The effect of applying early and late fusion techniques have also been tried out in this work. After stroke classification, a mechanism is needed to construct words from recognized strokes. Hence, an hidden markov model (HMM) model has been proposed under the current work in order to predict correct word from recognized strokes. A Language model

(LM) concept has been used to rectify the prediction shown by HMM [107]. An exclusive analysis regarding the evaluation of performances of different classifiers on different features sets (and/or combinations) for the recognition of online handwritten Bangla words is required to facilitate the researchers in this domain and this is done in Chapter 6.

1.6 Organization of the thesis

This thesis presents different methods and strategies used to design an online handwritten isolated Bangla character/ text recognition. The subsequent parts are organized as follows:

A comprehensive literature survey is presented in Chapter 2 which deals with the works proposed by different researchers in order to build efficient online character/ text recognition systems for various scripts like English, Gurumukhi, Devanagari etc. This chapter also describes the research works done so far for online Bangla character/word recognition.

In Chapter 3 detailed description about the creation of both isolated character and word databases are provided. The details of data collection process are also described. The methods for database preparation from the collected samples are discussed in Section 3.3. Section 3.4 deals with document level ground truth generation process. Section 3.5 describes different statistics of collected character and word databases.

Chapter 4 describes different pre-processing and feature extraction methodologies applied on characters, strokes. Section 4.1 describes different pre-processing steps like duplicate point removal, point normalization, size normalization in order to cope up with speed and shape variations. Some structural, shape based, local and topological features (distance based, hausdorff distance based, CG based circle approach, transition count, crossing point, area feature, mass distribution and chord length, ZPT, curvature based) are developed for isolated character/stroke recognition purpose in Section 4.2. Section 4.3 reports the brief working principle of different classifiers that are used for classification purpose.

The experimental results for the recognition of isolated character are presented in Chapter 5. The character recognition results using holistic approach are mentioned in Section 5.1.1. Section 5.1.2 deals with results of character recognition using stroke based approaches, where, Section 5.1.2.1 describes recognition results of strokes and designing a rule based approach for the construction of character from recognized strokes. Section 5.1.2.2 describes a DFA based approach for the construction of characters. Character recognition result using deep learning approach (here CNN) is mentioned in Section 5.1.3.

Chapter 6 deals with detailed description about online Bangla word segmentation strategy which comprises finding out component strokes from a word sample, recognition of segmented strokes using different feature extraction procedures and construction of word from recognized strokes. Section 6.1.1 presents the detailed description of the proposed word segmentation procedure. Applied feature extraction methodology is mentioned in Section 6.1.2. The experimental details after application of word segmentation algorithm on online handwritten Bangla word samples and the effectiveness of features (modified distance based, point based and curvature based) on segmented primitives are mentioned in Section 6.1.3. Section 6.2 describes the mechanism to build HMM for the construction word samples from recognized strokes and Section 6.3 highlights the word recognition result by adopting HMM followed by LM.

Finally, in Chapter 7, the work presented in this thesis is concluded with future plans for extending this research.

2
Literature Review

The advancement of online handwriting systems for Indic scripts like Bangla, Devanagari are still in the preliminary stage because of inadequate database. Hence, preparing a standard database is one of the most important tasks for OHR systems for different regional languages. This task is a tedious and time consuming one that not only needs proper attention to every details (so that e.g. shape/size variations, speed variations etc.), but also to ensure that the data contains all the information that can be used in future. For example, the database should contain variations in terms of shape/size of different strokes, variation of number of pixels to form the same stroke due to different writing speed, variation in number and order of strokes to represent the same information, variation of different writing style for different age groups, genders, educational background etc. Again, the information of contributors must also be recorded because; online character or word recognition can be further extended to develop writer identification/verification system.

Therefore, appropriate type of data need to be incorporated in the database with rigorous evaluation of the data collection system. Efficient design of data collection forms and planning of appropriate data collection strategies are essential as it should be taken into consideration that replicating the same scenario again is loss of precious time and efforts. Considering these scenarios, still erroneous samples may exist in the preliminary collected database due to natural variations and human habits. Therefore, to finalize the database before publishing, manual checking of the whole database is essential. The goal is to obtain a database which will have the natural variation that can be encountered in real life situations.

2.1 Character Recognition

Recent economic globalization, digital advancement and increased business transactions across the globe have elevated the urge to develop OHR systems. It has gained renewed interest in recent times for its potential. Various approaches for online character recognition based on holistic, stroke/sub-stroke approaches are discussed in the literature. Among these, various significant and well established studies on OHR system can be found on English, Devanagari, Gurumukhi etc. In this section, some significant studies on online character recognition along with few recognition techniques in off-line mode are discussed.

Tappert et al. in [27] have reported a survey that describes state-of-the-art of online handwriting recognition. It is based on an extensive review of the literature, including journal articles, conference proceedings, and patents. Online versus offline recognition, digitizer technology, handwriting properties and recognition problems are discussed. Shape recognition algorithms, pre-processing and post-processing techniques, experimental systems, and commercial products are also examined in this survey.

According to the authors in [84], it is possible to have the same sequence of primitives for many different characters, especially for writer-independent environment introducing ambiguity in recognition of such characters. Authors have introduced a novel concept of ”Relative Connectivity” among the subsequent primitives to remove the ambiguity between different characters having the same sequence of primitives. Their observations have shown that almost all ambiguities in which different characters are having same sequence of primitives have been removed by using ”Relative Connectivity” approach. As this technique succeeds in removing ambiguities, very good recognition results, 98.3% for digits and 99.2% for uppercase letters, have also been observed.

The main goal of the work mentioned in [64] is to develop an on-line handwriting recognition system that will be able to recognize handwritten characters of several different writing styles. Due to the temporal nature of online data, this work has possible application to the domain of speech recognition as well. The work in this research has aimed to investigate various features of handwritten letters, their use and discriminative power, and to find reliable feature extraction methods, in order to recognize them. A feature set consists of 22 elements, extracted from sub-character primitives has been proposed using direction based feature extraction approach. This approach has succeeded in having robust pattern recognition features, while maintaining feature's domain space to a small, optimum quantity. Back-propagation neural network (BPN) technique has been used as classifier and recognition rate up to 87% has been achieved even for highly distorted handwritten characters.

Farha et al. in [39] have expressed handwriting processing as a domain in great expansion. According to them, the interest devoted to this field is not only explained by the exciting challenges involved, but also the huge benefits that a system, designed in the context of a commercial application. It helps to distinguish two classes of recognition systems are usually: online systems for which handwriting data are captured during the writing process, which makes available the information on the ordering of the strokes, and offline systems for which recognition takes place on a static image captured once the writing process is over. The field of personal computing has begun to make a transition from the desktop to handheld devices, thereby requiring input paradigms that are more suited for single hand entry than a keyboard. Online handwriting recognition allows for such input modalities. On-line handwritten scripts are usually dealt with pen tip traces from pen-down to pen-up positions.

The main objective is to recognize online handwritten documents, which includes characters, words, lines, paragraphs etc. There is extensive work in the field of handwriting recognition, and a number of reviews exits. Their approach was to recognize handwriting by using templates. Along with this they have maintained a unique user accounts, which enables a particular user to create his/her training sets. Three steps are involved in the recognition of a character in this project. These methods scan the segmented character of the user and matches with the stored training set or templates. Each of the methods increases the probabilities of the templates which closely match with the user input. The three methods are shape template matching, pixel density matching and stroke movement matching.

Kumara et al. in [61], have proposed an approach for classifying multivariate time series with applications to the problem of writer independent online handwritten character recognition. They have used UNIPEN dataset for the experiment. Each time-series is approximated by a sum of piecewise polynomials in a suitably defined Reproducing Kernel Hilbert Space (RKHS). Using the associated kernel function, a large margin classification formulation is proposed which can discriminate between two such functions belonging to the RKHS. The associated problem turns out to be an instance of convex quadratic programming. The resultant classification scheme has been applied to many time-series discrimination tasks and reflected encouraging results when applied to online handwriting recognition tasks. The main contribution of the work is to propose a kernel function for time-series which can be used not only for classification, but other applications like clustering, novelty detection, etc. Some important contribution is to generalize the piecewise linear interpolation scheme used in [111] by Sivaramakrishnan and Bhattacharyya to piecewise polynomials in a RKHS setting. This is interesting in its own right as it might open up new applications which involve interpolation. Lastly the paper also gives an O(n) algorithm for computing the interpolation step rather than an O(n^3) algorithm proposed in [111].

In [13], authors have described a comprehensive description of writer independent online handwriting recognition system ”frog on hand”. The focus of this work concerns the presentation of the classification/training approach, which they call cluster generative statistical dynamic time warping (CSDTW). CSDTW is a general, scalable, HMM-based method for variable-sized, sequential data that holistically combines cluster analysis and statistical sequence modelling. It can handle general classification problems that rely on this sequential type of data, e.g., speech recognition, genome processing, robotics, etc. Contrary to previous attempts, clustering and statistical sequence modelling are embedded in a single feature space and they have used a closely related distance measure. They have shown character recognition experiments of "frog on hand" using CSDTW on the UNIPEN online handwriting database. The recognition accuracy is significantly encouraging.

In [33], authors have attempted to recognize 40 major Devanagari characters that occur in the core-strip of the writing. In this experiment they have not considered the modifier symbols and conjunct formation. 5 samples of each character from 20 different individuals are collected using a IBM/Cross Pad. A combination of HMM and nearest neighbour classifiers are used for classification, capturing different levels of on-line and off-line information. Three samples of each writer are used for training and the remaining two samples have been used for testing. A recognition rate of 69.2% is achieved using a traditional HMM-based on-line recognizer. An examination of the substitution errors, revealed that the HMM model could not capture many of the structural properties needed for classification. Using an iterative procedure, new classifiers have been designed, one at a time, in an attempt to capture some of the structural properties of the characters. Through a combination of these classifiers, the classification accuracy has been improved to 86.5%.

Authors in [57], have described a system for the automatic recognition of isolated handwritten Devanagari characters. Owing to the large number of characters and resulting demands on data acquisition, they have used structural recognition techniques to reduce some characters to others. The residual characters are then classified using the subspace method. The results of structural recognition and feature-based matching are mapped to give final output. The proposed system is evaluated for the writer dependent scenario. Here the average accuracy on the FREQ subset has been observed as 93%.

A system for recognition of online handwritten characters has been presented for Indian writing system for Devanagari, Telegu and Tamil scripts in [117, 118]. According to the authors, handwritten character can be represented as a sequence of strokes whose features are extracted and classified. In [118], authors have used spatiostructural features for stroke recognition purpose. Support vector machine (SVM) have been used for constructing the stroke recognition engine. The recognition accuracy is really encouraging.

Authors in [60] have presented a novel scheme, which can be implemented on the iPhone, for the recognition of online handwritten basic isolated characters of the Devanagari script. According to them, unconstrained Devanagari writing is more complex than English cursive due to the possible variations in the order number, direction and shape of constituent strokes. A manual study of various characters have been performed and 42 stroke classes are created. A stroke based recognition approach has been designed where strokes are recognized using HMM. One HMM is constructed for each stroke class. A second stage of classification has been designed and is used for recognition of characters using stroke classification results along with look up tables.

In [121], Tripathi et al. have proposed a scheme for standardization of stroke order thereby training the user and using the strategy for future isolated Devanagari character recognition on iPhone. On the basis of manual study of stroke classes, one HMM is created for each class which is chosen for character recognition in that HMM class. Standardization of stroke order reduces the number of HMM promoting higher probability and faster recognition of isolated Devanagari character. A few characters among the 20 most occurring characters are taken as samples and their manual study generates certain stroke classes. The second phase deals with the training and recognition of isolated Devanagari character.

Lajish et al. [62] have described a new feature set named as extended directional features (EDF) for use in the recognition of online handwritten strokes. They have used EDF specifically to recognize strokes that form a basis for producing Devanagari script, which is the most widely used Indian language script. Experiments are conducted for the automatic recognition of isolated handwritten strokes. Initially authors have described the proposed feature set (EDF) and then have shown how this feature can be effectively utilized for writer independent script recognition through stroke recognition. Experimental results show that the extended directional feature set performs well with about more than 65% stroke level recognition accuracy for writer independent dataset.

A neural network based framework has been proposed by Kubatur et al. [59] to classify online Devanagari characters into one of 46 characters in the alphabet set. The uniqueness of the work is three-fold: (a) feature extraction is based on the Discrete Cosine Transform (DCT) of the temporal sequence of the character points (utilizing the nature of online data input). They have also shown that simple feature set yielded by the DCT can be very reliable for accurate recognition of handwriting, (2) the mode of character input is through a computer mouse, and (3) they have built the online handwritten database of Devanagari characters from scratch, and there are some unique features in the way authors have built up the database. Lastly, the testing has been carried out on 2760 characters, and recognition rate of up to 97.2% is achieved.

Santosh and Iwata, in [88], have categorized character recognition system into two modules: learning and testing. In learning or training module, handwritten strokes are learnt or stored. In the testing module, every test stroke is matched with the templates (learnt in training module) so that authors can find the most similar one. This procedure is then repeated for all available test strokes. At the end, aggregating all the matching scores provides an idea of the test character closer to the template. In this study, an established as well as validated approach (based on previous studies by Santosh & Nattee in [89–91], Santosh et al. in [92], in [93]) has been presented for on-line handwritten Devanagari character recognition. It applies the number of strokes used to complete a symbol and their spatial relations. Considering the dataset, the success rate has been observed approximately 97% in less than 2 seconds per character on average. The evaluation protocol reduces the errors (mainly due to multi-class similarity) and the optimized DTW reduces the delay in processing – which has been new attestation in comparison to the previous studies. The proposed approach is able to handle handwritten symbols of any stroke and order. Moreover, the stroke-matching technique is completely controllable. It is primarily due to symbol categorization and the use of stroke spatial information in template management.

Santosh et al. in [93], have proposed a new scheme for Devanagari handwritten character recognition. It is primarily based on spatial similarity-based stroke clustering. Feature of a stroke consists of a string of pen-tip positions and directions at every pen-tip position along the trajectory. It uses the DTW algorithm to align handwritten strokes with stored stroke templates and determine their similarity. Experiments are carried out for 25 native writers and a recognition rate of approximately 95% is achieved.

Santosh et al. in [87], describe a method for isolated handwritten character recognition using dynamic programming for matching the non-linear multi-projection profiles that are produced from the Radon transform. The idea is to use DTW algorithm to match corresponding pairs of the Radon features for all possible projections. By using DTW, they have claimed that they have avoided compressing feature matrix into a single vector which may miss some important information. It can handle character images in different shapes and sizes that are usually occurred in natural handwriting in addition to difficulties such as multi-class similarities, deformations and possible defects.

In [65] authors have presented an implementation to recognize online handwritten Gurumukhi strokes using SVM. This implementation starts with pre processing, which consists of 5 basic algorithms. Prior to these algorithms, a basic step called stroke capturing is done, which samples data points along the trajectory of an input device. After pre-processing, recognition of Gurumukhi stroke is done using SVM with the help of two cross validation techniques, namely, holdout and k-fold. The recognition is based on the unique IDs identified as the strokes in order to represent a word. These strokes are taken from the one hundred Gurumukhi words written by 3 different writers.

In [108], authors have implemented a elastic matching based technique to recognize online handwritten Gurumukhi characters. They have discussed the process to recognize characters in two stages. The first stage recognizes the strokes by implementing elastic matching for online handwritten Gurumukhi strokes, in second stage, character is evaluated on the basis of recognized strokes. Feature are computed to strengthen recognition results. Features are classified into two categories, namely, low-level and high-level features. A few neighbouring points estimate low-level features where as high-level features are estimated on larger scale than low-level features. The features computed in their study include both low-level and high-level features. Authors have used linearity, curliness, width, height, aspect ratio, slope, area etc. as low-level features. Loops, crossings, straight line, headline and dot are used as high-level features. The database for strokes stores script number, stroke number and stroke sample number for every point of a stroke. With 60 writer's writing and a set of 41 Gurumukhi characters, they have obtained recognition rate as 90.08%.

Authors in [109], have presented a HMM-based online handwritten character recognition for Gurumukhi script. They have discussed about a procedure to develop a HMM in order to recognize Gurumukhi characters. A test with 60 handwritten samples, where each sample includes 41 Gurumukhi characters, have shown a 91.95% recognition rate, and an average recognition speed of 0.112 seconds per stroke.

The work mentioned in [10] deals with the offline recognition of handwritten Gurumukhi characters. Here two sets of features based on gradient and curvature of character image are computed. The extracted features are then fused together to form a composite feature vector. Two ways of generating this composite feature vector is presented in this paper. Dimensionality of the generated composite feature vectors is 400. The efficiency of these feature sets is tested on a handwritten database of Gurumukhi characters containing 7000 sample character images. The experimental result demonstrates the usefulness of curvature-based feature guided by gradient information and recognition rate of 98.56% is obtained. SVM is used for classification purpose.

Apart from holistic, stroke/sub-stroke based recognition by using handcrafted features, many authors have tried to recognize the same by following deep learning based strategies. In deep learning approach, features need not to extract manually rather features are generated automatically in convolution phase. Some of the works related to online handwriting recognition for different scripts are mentioned below.

According to the authors in [11], character recognition has been widely used since its inception in applications involved processing of scanned or camera captured documents. There exist multiple scripts in which the languages are written. The scripts could broadly be divided into cursive and non-cursive scripts. The recurrent neural networks have been proved to obtain state-of-the-art results for optical character recognition. Here authors have presented a investigation of the performance of Recurrent Neural Network (RNN) for cursive and non-cursive scripts. They have employed bidirectional long short-term memory (BLSTM) networks, which is a variant of the standard RNN. The output layer of the architecture used to carry out our investigation is a special layer called connectionist temporal classification (CTC) which does the sequence alignment. The CTC layer takes as an input the activations of LSTM and aligns the target labels with the inputs. The results obtained at the character level for both cursive Urdu and non-cursive English scripts are significant and suggest that the BLSTM technique is potentially more useful than the existing OCR algorithms.

In [37], authors have proposed a novel framework of writer adaptation based on deeply learned features for online handwritten Chinese character recognition. Their motivation is to further boost state-of-the-art deep learning-based recognizer by using writer adaptation techniques. First, to perform an effective and flexible writer adaptation, they have proposed a tandem architecture design for the feature extraction and classification. Specifically, a deep neural network (DNN) or CNN is adopted to extract the deeply learned features which are used to build a discriminatively trained prototype-based classifier initialized by Linde–Buzo–Gray clustering techniques. In this way, the feature extractor can fully utilize the useful information of a DNN or CNN. Meanwhile, the prototype-based classifier could be designed for more compact and efficient solution in practical. Second, the writer adaption is performed via a linear transformation of the deeply learned features which are optimized with a sample separation margin-based minimum classification error criterion. Furthermore, authors have improved the generalization capability of the previously proposed discriminative linear regression approach for writer adaptation by using the linear interpolation of two transformations and adaptation data perturbation. The experiments on the tasks of both the CASIA-OLHWDB benchmark and an in-house corpus with a vocabulary of 20,936 characters demonstrate the effectiveness of their proposed approach.

The work mentioned in [38], has mainly focussed on the recognition of handwritten Arabic characters that face several challenges, including the unlimited variation in human handwriting. The work provides a deep learning technique that have been effectively applied to recognize Arabic handwritten digits. LeNet-5, a form of CNN trained and tested MADBase database that contain 60000 training and 10000 testing images. A comparison is held amongst the results, and it is shown by the end that the use of CNN is leaded to significant improvements across different classification algorithms.

Authors in [120] have proposed a novel methodology that uses the geometric characteristics of line-segment representations to optimize the hyper-parameters for the deep networks. The methodology is applied to a line-segment-based stacked auto-encoder to verify its effectiveness. It is found that the line-segment-based visualizations can increase the interpretability of the deep models and facilitate the configurations for the hyper-parameters.

Inspired by the theory of Leitner's learning box from the field of psychology, authors in [126] have proposed DropSample, a new method for training deep convolutional neural networks (DCNNs), and applied it to large-scale online handwritten Chinese character recognition (HCCR). According to the principle of DropSample, each training sample is associated with a quota function that is dynamically adjusted on the basis of the classification confidence given by the DCNN softmax output. After a learning iteration, samples with low confidence will have a higher frequency of being selected as training data; in contrast, well-trained and well-recognized samples with very high confidence will have a lower frequency of being involved in the ongoing training and can be gradually eliminated. As a result, the learning process becomes more efficient as it progresses. Furthermore, authors have investigated the use of domain-specific knowledge to enhance the performance of DCNN by adding a domain knowledge layer before the traditional CNN. By adopting DropSample together with different types of domain-specific knowledge, the accuracy of HCCR is improved efficiently. Experiments on the CASIA-OLHDWB 1.0, CASIA-OLHWDB 1.1, and ICDAR 2013 online HCCR competition datasets yield outstanding recognition rates of 97.33%, 97.06%, and 97.51% respectively.

According to the authors in [124], CNNs have better performance in feature extraction and classification. Most of the applications are based on a traditional structure of CNNs. However, due to the fixed structure, it may not be effective for large dataset which will spend much time for training. So, they have used a new algorithm to optimize CNNs, called directly connected convolutional neural networks (DCCNNs). In DCCNNs, the down-sampling layer can directly connect the output layer with three-dimensional matrix operation, without full connection (i.e., matrix vectorization). Thus, DCCNNs have less weights and neurons than CNNs. They have also conducted the comparison experiments on five image databases: MNIST, COIL-20, AR, Extended Yale B, and ORL. The experiments show that the model has better recognition accuracy and faster convergence than CNNs. Furthermore, two applications (i.e., water quality evaluation and image classification) following the proposed concepts further confirm the generality and capability of DCCNNs.

Fu et al. in [42] give an experimental study on the stability of an extreme learning machine (ELM) and its generalization capability. Focusing on the relationship between uncertainty of an ELM's output on the training set and the ELM's generalization capability, the experiments have shown some new results in the viewpoint of classical pattern recognition. The study provides some useful guidelines to choose a fraction of ELMs with better generalization from an ensemble for classification problems.

Some good research works for character recognition in English [13, 27, 39, 61, 64, 84], Devanagari [33, 57, 59, 60, 62, 87, 88, 93, 117, 118, 121] and Gurumukhi [10, 65, 108, 109] scripts are reported in the literature as mentioned above. In addition, many authors have started adopting cutting-edge machine learning approaches like deep learning [11, 37, 38, 120, 123, 124, 126], extreme learning machine (ELM) [42] for different scripts. In contrast, for Bangla script, researchers still have not explored the problem much and limited research materials available in the literature.

Roy et al. in [81], have referred to OHR as a problem for elucidation of handwritten input image which is confined as a stream of pen positions collected by using either a digitizer or other pen position sensors. Both sequential and dynamic information are extracted from pen movements on the writing pads. These information acts as features for their proposed online Bangla handwriting recognition system. These features are then fed to quadratic classifier for the recognition purpose. Authors have tested the system on 2500 Bangla numeral data and 12500 Bangla character data and obtained 98.42% accuracy on numeral data and 91.13% accuracy on character data from the proposed system.

Authors in [68] have presented the benchmark recognition results on four most popular Indic scripts based on two existing feature extraction techniques known as point-float and direction code histogram. Efficiency of their system has been tested through Nearest Neighbour, HMM and MLP classifiers.

Authors in [45] proposed a new scheme for online handwriting recognition for Indic script. The primary concern of the approach is the modelling of human motor functionality while writing characters. This is achieved by looking at the whole pen trajectory where the time evaluation of the pen coordinates plays a crucial role. A low complexity classifier has been designed and the proposed similarity measure appears to be quite robust against wide variations in writing styles. Initially, the approach has been applied for online recognition of handwritten characters in Devnagari and Bangla. A test on a dataset of considerable size shows promising recognition rates of: 97.29% for Devnagari and 96.34% for Bangla.

In [22], authors have reported an approach for handwritten Bangla character recognition of with the help of direction code based features. They have implemented the proposed approach on a database of 7043 online handwritten Bangla character samples. This is a 50-class recognition problem and they have achieved 93.90% and 83.61% recognition accuracies respectively on its training and test sets.

According to the authors in [14], a representation at stroke level can be prepared based on their structure or shape, where each stroke corresponds to a sequence of shape features. An un-known stroke is then recognized by means of a previously built stroke database using DTW technique. After identifying all the component strokes, a recognition module tries to recognize a full character. An experiment has been conducted with a total of 495 classes on 20,873 data samples and 10 people as data contributors yielding 97.33% recognition rate with 2.18% misrecognition rate and 0.5% rejection rate.

Authors in [47] have designed a novel feature vector by considering the direction of writing, curvature, curliness, inclination, standard deviation of x and y coordinates at stroke level. In this paper, all constituent strokes have been firstly divided into 9 local zones, from where the mentioned feature vector is generated. Classification scheme is performed by using supprort vector machine (SVM) classifier. The achieved character recognition accuracy was 87.48% for Bangla script and 84.10% for Devanagari script on 4900 and 5000 test samples respectively

Authors in [73], have manually grouped constituent strokes forming characters into 54 classes based on the similarity of shapes in the graphemes. For stroke recognition, they have constructed one HMM for each stroke class. Characters are finally recognized by the classification of stroke results using previously made 50 look up tables. The classification accuracy at the stroke level is 84.6% on the test set. The character recognition accuracy obtained by the proposed scheme is 87.7% on the test set.

In [82], authors have extracted the component strokes from the characters. Next, the sequential and dynamic information extracted at stroke level (by tracking the pen movements on the writing pad) are considered as features. These features are then fed to MLP classifier for stroke recognition. Characters are then constructed from the recognized strokes by matching the stroke sequences stored in the database. The system has been tested on 21372 Bangla character data and obtained 88.23% accuracy (92.92% accuracy when top 5 choices are considered).

2.2 Word Recognition

Since last two decades, the trend of recognizing handwritten word samples has seen a significant growth irrespective of the languages in which those are written. Various paradigms for the recognition of handwritten words have also been introduced in the pattern recognition and machine learning literature. Recognition of words can be done in a holistic fashion or in stroke/sub-stroke based approach. In the holistic approach, a word sample is recognized as a whole unit, irrespective of the number of strokes and/or syllables in the sample. Many researchers [9, 12, 15, 24–26, 30, 44, 51, 53, 75, 80, 95, 96, 112, 113, 127–129] have investigated for the recognition of off-line printed words/scripts written Bangla, English, etc. In this section, some of the off-line works are mentioned below.

Holistic Word Recognition is one of the new modalities for handwritten word identification. The holistic paradigm in handwritten word recognition treats the word as a single, indivisible entity and attempts to recognize words from their overall shape, as opposed to recognize the individual characters comprising the word. In the work mentioned in [9] reports a longest-run based holistic feature, that has been used to classify word images belonging to different classes, using a neural network based classifier. To evaluate the technique, a few words from the handwritten documents of the CMATERdb 1.2.1 dataset have been used. Frequently occurring English words are manually extracted from the handwritten pages and the accuracy of the technique is evaluated using a three-fold cross-validation method. The best-case and average-case performances of the technique to the said data set are 89.9% and 83.24% respectively.

An algorithm for segmenting unconstrained printed and cursive words has been proposed in [25]. The algorithm initially over-segments handwritten word images (for training and testing) using heuristics and feature detection. An artificial neural network (ANN) is then trained with global features extracted from segmentation points found in words designated for training. Segmentation points located in ”test” word images are subsequently extracted and verified using the trained ANN. Two major sets of experiments have been conducted, resulting in segmentation accuracies of 75.06% and 76.52%.

Segmentation of cursive handwritten Bangla script is considered by the authors mentioned in [15]. Unlike English, Bengali handwritten characters and its components often encircle the main character, making the conventional segmentation methodologies inapplicable. Experimental results, using the proposed segmentation technique, on sample cursive handwritten data containing 218 ideal segmentation points show a success rate of 97.7%.

In the work mentioned in [95], authors have designed a novel two-stage approach for segmentation of isolated Bangla word images. In the first stage, a feature based approach is designed to classify the connected word segments into either of the two classes, namely, ‘Segment further’ and ‘Do not Segment’ using a MLP based classifier. In the second stage, fuzzy segmentation features are designed to identify the ‘Matra’ region and the potential segmentation points on the ‘Matra’ of the connected word segments that belong to ‘Segment further’ class. Using the current technique, the overall successful segmentation accuracy achieved after two stages is 95.87%.

Authors in [24] have introduced a stroke based lexicon reduction technique in order to reduce the search space for recognition of handwritten words. The principle of this technique involves mainly two aspects of a word image to constitute a feature vector: one is word-length and the other is shape of the word. The length of the word image is represented by the number of specific vertical strokes present in the word image and, on the other hand, the shape of a word image is realized with the combination of both horizontal and vertical strokes. The experiment has been carried out with a database of 35,700 off-line handwritten Bangla word images. According to them, though their proposed lexicon reduction technique has been developed for recognition of Bangla handwritten words, its generalization property can easily be exploited for recognition of handwriting in other scripts also.

In [112] authors have described postal address interpretation is the task of assigning to letter mail pieces a delivery point encoding, e.g., ZIP+4 Code. The encoding is determined from images of destination addresses on mail piece faces; addresses that are handwritten, are of poor-quality machine printing, are incomplete or incorrect. They have described several recognition algorithms used in the interpretation of handwritten and machine-printed address text (digits / symbols / alphabets / words).

Authors in [26] deal with several component processes of a recognition system for isolated off-line cursive script words. The approach described here transforms a word image through a hierarchy of representation levels: points, contours, features, letters, and words. A unique feature representation is generated bottom-up from the image using statistical dependencies between letters and features. Ratings for partially formed words are computed using a stack algorithm and a lexicon represented as a trie. Several new techniques for low- and intermediate-level processing for cursive script are described, including heuristics for reference line finding, letter segmentation based on detecting local minima along the lower contour and areas with low vertical profiles, simultaneous encoding of contours and their topological relationships, extracting features (e.g., middle loop, upper zone stroke), and finding shape-oriented events.

Segmentation of cursive words into letters has been one of the major problems in handwriting recognition. Authors in [127] have introduced a new segmentation algorithm, guided in part by the global characteristics of the handwriting. They have found the successive segmentation points by evaluating a cost function at each point along the baseline. The cost of segmenting at a point is a weighted sum of four feature values at that point. The weights of the features are determined using linear programming. Authors have tested with 750 words written by 10 writers, 97% of the letter boundaries were correctly located.

The purpose of the survey mentioned in [75] is to provide a comprehensive overview of the application of Markov models in the research field of offline handwriting recognition, covering both the widely used HMM and the less complex Markov-chain or n-gram models. Firstly authors have introduced the typical architecture of a Markov-model-based offline handwriting recognition system and make the reader familiar with the essential theoretical concepts behind Markovian models. Then, authors have given a thorough review of the solutions proposed in the literature for the open problems how to apply Markov-model-based approaches to automatic offline handwriting recognition.

Keorich et al. in [12] have presented a hybrid recognition system that integrates HMM with NN in a probabilistic framework. The input data are processed first by a lexicon–driven word recognizer based on HMMs to generate a list of the candidate N–best–scoring word hypotheses as well as the segmentation of such word hypotheses into characters. An NN classifier is used to generate a score for each segmented character and in the end, the scores from the HMM and the NN classifiers are combined to optimize performance. Experimental results show that for an 80,000–word vocabulary, the hybrid HMM/NN system improves by about 10% the word recognition rate over the HMM system alone.

Ray et al. in [80] have presented a Deep bidirectional long short-term memory (BLSTM) based recurrent neural network (RNN) architecture for text recognition. This architecture used CTC for training to learn the labels of an un-segmented sequence with unknown alignment. This work is motivated by the results of deep neural networks for isolated numeral recognition and improved speech recognition using deep BLSTM based approaches. Deep BLSTM architecture has been chosen due to its ability to access long range context, learn sequence alignment and work without the need of segmented data. Due to the use of CTC and forward backward algorithms for alignment of output labels, there are no Unicode re-ordering issues, thus no need of lexicon or post-processing schemes. This is a script independent and segmentation free approach. This system has been implemented for the recognition of un-segmented words of printed Oriya text. This system achieves 4.18% character level error and 12.11% word error rate on printed Oriya text.

Few online word recognition strategies have been mentioned in [18, 30, 76, 85, 86, 110, 122, 122, 130] for various languages like English, Gurumukhi, Devanagari etc.

Authors in [130] have proposed a cascade connection HMM (CCHMM) method for online English word recognition. This model extends the way of HMM pattern description of handwriting English words by allowing state transition, skip and duration. According to the statistic probabilities, the behaviour of handwriting curve may be depicted more precisely. Viterbi algorithm for the cascade connection model may be applied after the whole sample series of a word is input. Compared with the method of creating models for each word in lexicon, this method gives a faster recognition speed. Experiments show that CCHMM approach could obtain 89.26% recognition rate for the first candidate, while the combination of character and ligature HMM method's first candidate produces 82.34% recognition rate.

The classification of online handwritten word samples can be effectively addressed by a granular computing approach as mentioned in [122]. In fact, handwriting can be viewed as a sequence of information granules consisting in single strokes. In this paper, an automatic handwriting recognition system is proposed. An oriented sequence of nodes, as a particular directed labelled graph, is used to represent each handwritten pattern. Each node of the graph stores the feature vector describing a single stroke, while the edge connecting each node to the succeeding one stores information about the pen displacement between the two strokes (usually referred as virtual stroke). Once the handwritten patterns have been represented by labelled graphs, a general technique for automatic graph classification is used to perform different recognition tasks. The tackled tasks include word recognition, writer recognition and character set recognition. The tests have been carried out using real world data.

In [85], authors have proposed a novel method to segment the Gurumukhi text based on pressure, pen down status and time together. In case of some characters getting over segmented due to the shape of character and user's style of writing, a method to merge the sub strokes is presented. The proposed algorithms have been applied on a set of 2150 words and have given very good segmentation results.

Sachan et al. in [86] have considered user's handwriting as a sequence of packets captured through the movement of stylus or pen on the surface of the tablet. The packet consists of x, y position of the stylus, button (tip of stylus), pressure of the stylus and the time of each packet. The user's writing is pre-processed and is segmented into meaniningful shapes. The segmented shapes are processed to extract directional features. The feature data is fed to the recognition engine which is a NN Classifier. The average word recognition accuracy is 76% approximately.

A hybrid neural network model is developed and applied to online handwritten word recognition in [30]. The word recognition system uses a module that assigns character class confidence values to segments of images of handwritten words. The module accurately represent ambiguities between character classes and assign low confidence values to a wide variety of non-character segments resulting from erroneous segmentation. The proposed hybrid neural model is a cascaded system. The first stage is a self-organizing feature map algorithm (SOFM). The second stage maps distances into allograph membership values using a gradient descent learning algorithm. The third stage is a multilayer feed forward network (MLFN). The new system performs better than the baseline system. Experiments have been performed on a standard test set from the SUNY/USPS database and achieved 87.37% word recognition accuracy.

For online English word recognition, authors in [110] have proposed a Conditional Random Field (CRF) driven beam search strategy that can be applied on a combined segmentation and recognition framework. To do so, firstly, authors have built a candidate segmentation lattice using over-segmented primitives of the word patterns and then recognition has been performed by synchronously matching lexicon words with nodes of the lattice. Probable search paths are evaluated by integrating character recognition scores with geometric and spatial characteristics of the handwritten segments into the CRF model. To make computation efficient, authors have used beam search to prune the set of likely search paths. Authors have achieved 78.72% and 92% word recognition accuracies over IBM_UB_1 (5000 training and 1795 test samples) and UNIPEN (2127 training and 2127 test samples) database respectively.

Authors in [122] have proposed a system that encodes a word pattern as a sequence of strokes along with some inter stroke information, in a special kind of directed labelled graph. Each node of the graph keeps feature information of a single stroke and edge stores the information about displacement between two associated strokes. Then graph classification system uses a special mining algorithm to perform different recognition tasks.

In another work mentioned in [18], authors have used HMM for recognition of online Tamil word samples. Here, the HMM is used to find the highest weighted path of the sequence involved in the word formation using Viterbi algorithm depending on a back-end database defined in the course of the project. The highest weighted path sequence is the output word of the input handwritten sample.

In [76], authors have proposed a new method of creating character-level representation of text to reduce the computational costs associated with training a DCNN. They have demonstrated that their method of character embedding greatly reduces training time and memory use, while significantly improving classification performance. In their experiment authors have achieved 84.5% word recognition accuracy.

In case of online Bangla OHR, limited research articles are available in the literature. Even, most of the works only describe how to extract sub-stroke level features from word samples and recognition of the same. In the following section few available works obtained from the literature are mentioned.

Approach mentioned in [40], tries to recognize cursively written Bangla word samples by extracting sub-stroke level features and proposed a writing model based on HMM for the Bangla script. Authors have achieved approximately 93% recognition accuracy over 6516 online test Bangla word samples.

In [67], authors have introduced a hybrid classification strategy to the handwriting recognition community. In this strategy, an MLP architecture is used for feature extraction purpose and SVM classifier is used for final classification. Sub-stroke level feature representation of the input word sample is fed to the MLP trained by back propagation (BP) algorithm. The values computed at the hidden nodes of the MLP are used as a transformed feature vector and fed to the SVM as its input. This strategy of using an MLP for feature extraction and a SVM for classification purpose, improves the recognition accuracy from their individual level. In the best situations, we obtained 88.79% and 87.20% recognition accuracies respectively on the test sets of two databases corresponding to 50 and 110 city names. In each case, these accuracy figures correspond to reduced feature vector (90 dimensional) provided by MLP and SVM as the classifier.

The method reported in [32] describes online Bangla word recognition by using a Weighted Finite-State Transducer (WFST) based LM for improving the recognition accuracy. Both the recognition hypothesis (i.e. the segmentation lattice) and the lexicon are modelled as two WFSTs. Concatenation of these two FSTs accept a valid word(s) which is (are) present in the recognition lattice. A third FST called error FST is also introduced to retrieve certain words which were missing in the previous concatenation operation. The proposed model has been tested for online Bangla handwritten word recognition. Experiment on a part of ISI-Bangla handwriting database shows that while the present classifiers (without using any LM) can recognize about 73% word, use of recognition and lexicon FSTs improve this result by about 9% giving an average word-level accuracy of 82%. Introduction of error FST further improves this accuracy to 93%.

Authors in [23] have proposed an analytical recognition approach, which involves segmentation of the input Bangla word. Modified quadratic discriminant function classifier is applied for recognition of segmented strokes using a chain code histogram based feature vector. Finally, an input word is recognized by a verification module, which uses a set of rules for construction of characters from strokes. A total of 10000 handwritten online word samples provided by 50 native Bengali writers have been used in their experiment. Recognition error at stroke level and at character level are 1.22% and 1.96% respectively. Overall word level recognition accuracy on the test set is 82.34

In OHR domain, accuracy of segmentation based word recognition method heavily depends on proper segmentation of word samples to obtain valid component strokes. Authors have tried to collect stroke level information from target word by estimating busy zone over it and then applying Down->Up->Down (DUD) approach for segmentation [19, 20, 46]. In these experiments, texts are firstly segmented into primitives which are recognized in the next step. They have proposed a segmentation technique by applying combination of both offline and online information and developing some rules. The segmented primitives have a chance of being either basic/compound character or a part of a basic/compound character. Directional features have been used towards recognition of those primitives. Some rules have also been discovered by examining the different combination patterns formed by Bangla characters.

Chowdhury et al. in [31] have segmented the online word sample into several segments. In the next step, they have used fuzzy linguistic rules to construct the word sample from segmented strokes. They have achieved 77% recognition accuracy over 500 online handwritten Bangla word samples.

As online handwriting recognition depends on the order of writing, words written with different stroke-order are treated as different words to the online recognizer, Bhattacharya et al. in [21] have proposed a stroke-order normalization method for Bangla online word recognition using offline and online information. In this technique, firstly, based on the offline information, sub-strokes in a word are ordered according to their relative positions. This results in similar stroke-order among the different instances of the same word. Next, online information of each ordered sub-stroke is used for feature extraction. They have tested their method on a dataset of 6000 words and obtained 74.65% and 90.53% word recognition accuracies, respectively, before and after stroke-order normalization.

Mukherjee et al. in [69] have proposed a hybrid layered architecture comprising of three networks CNN, RNN and CTC for OHR system without use of any specific lexicon. Feature extraction and classification are two major modules of such a recognizer. Deep architectures of CNN models have been found to be efficient in extraction of useful features from raw signal. On the other hand, a RNN along with CTC has been shown to be able to label un-segmented sequence data. They have achieved 84.47% word recognition accuracy over 15277 test Bangla samples.

In [41], authors have proposed an unsupervised feature generation approach based on dissimilarity space embedding (DSE) of local neighbourhoods around the points along the pen trajectory. In this work, they have also proved that DSE has the ability to produce discriminative feature representation and thereby is beneficial for classification.

Srimany et al. in [114] have observed that simple concatenation of individual feature vector does not help much in improving the overall word recognition accuracy. Hence in their work, they have trained distinct SVM classifiers with different feature vectors and combine their outputs at the final stage to improve overall performance. Proposed model exhibits 89.92% success rate when applied on 27789 online test Bangla word samples.

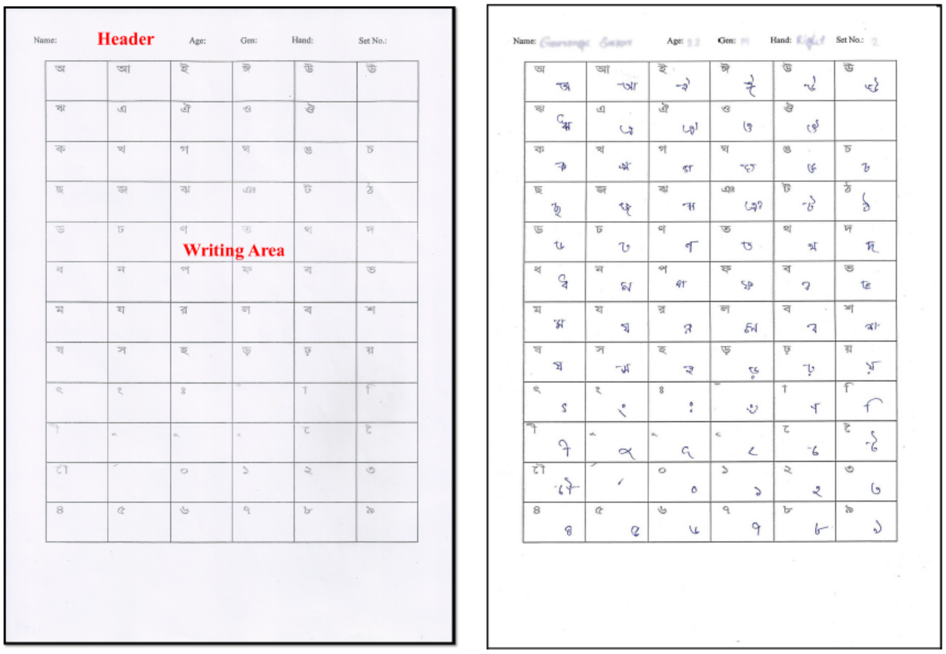
3
Database Development

The advancement of online handwriting recognition systems for Indic scripts like Bangla, Devanagari is still in the preliminary stage because of inadequate database. Hence, preparing a standard database is one of the most important tasks for OHR systems for different regional languages. This is a laborious and time consuming process that not only takes care of every details (like shape/size variations, speed variations of the handwriting to be incorporate in the database etc.), but also to ensure that the data contains all the information which can be useful for the future researchers. For example, the database should contain variations in terms of shape/size of different strokes, variation of number of pixels to form the same stroke due to different writing speeds, variation in number and order of strokes to represent the same information, variation of different writing styles of the people belonging to different age groups, genders, educational background etc. Again, the information of contributors must also be recorded because; online character or word recognition can be further extended to develop writer identification or verification system. Therefore, appropriate type of data with supporting information need to be incorporated in the database after rigorous evaluation of the requirement of the data collection system. Efficient design of data collection forms and planning of appropriate data collection strategies are essential as it should be taken into consideration that replicating the same scenario again is loss of precious time and efforts. Even if all these factors are considered, still all these scenarios, still erroneous samples exist in the preliminary collected database due to natural variations and human habits. Therefore, to finalize the database before make it available to the research community, manual checking of the whole database is essential. The goal is to obtain a database which will have the natural variation that can be encountered in real life situations.

3.1 Design of data collection form

The primary stage of database development process, is the design of data collection form(s). During design of this form various factors are considered. Different aspects of the populous may influence handwriting traits like: gender, age, educational qualification, religion, etc. Also, writing at different intervals can affect the handwriting style of different individuals. To capture these variations and to simplify the problem, data collection forms are designed that contain characters of Bangla script that are mostly used for writing. Isolated character data collection form(s) and handwritten text data collection form(s) are designed seperately. Isolated characters include all the basic characters of the alphabet available in Bangla along with numerals and vowel modifiers. The data collection form is divided into two zones where in the header zone writer information are collected and in the writing zone writers need to write the suggestive characters in their own handwriting. Fig. 3.1.1 (a) shows a blank data collection form of isolated characters where all the fields are marked for easy understanding, whereas 3.1.1 (b) shows corresponding filled in form of (a). During design of isolated form, all vowels and alphabets that are used in writing Bangla texts are considered along with vowel modifiers. Only one vowel modifier '◌ো'is not incorporated as this can be generated by combining vowel modifier '◌ে' and vowel modifier '◌া'. Total 11 vowels, 40 consonants, 10 vowel modifiers and 10 numerals are considered to design the isolated character data collection form.

In Bangla writing system, unlike Roman, upper case and lower case characters are absent, but the script is more complex due to the presence of modified complex and compound characters formed by combinations of simple characters [35]. Therefore, to incorporate all possible simple and complex characters in our collected data, the data collection form is divided into two parts: simple text sample forms and complex text sample forms. Simple form contains words formed of basic characters only, complex form contains mostly complex characters formed by using basic characters. Each form is divided into three major zones. Header zone is the first zone where a writer can put information like: name, age, gender etc. In the second zone i.e. Body-Upper, here machine printed text is provided that required to be copied by the writers. This text is created with help of the faculty members of Bengali department of West Bengal State University, West Bengal, India, to ensure the presence of most of the characters (basic, complex and compound) of Bangla. Fig. 3.1.2 shows a blank form (left) and a filled-in (right) data collection form, where in (a), all the fields are marked for better understanding. Fig. 3.1.3 (a-b) highlights the blank and filled document level forms.



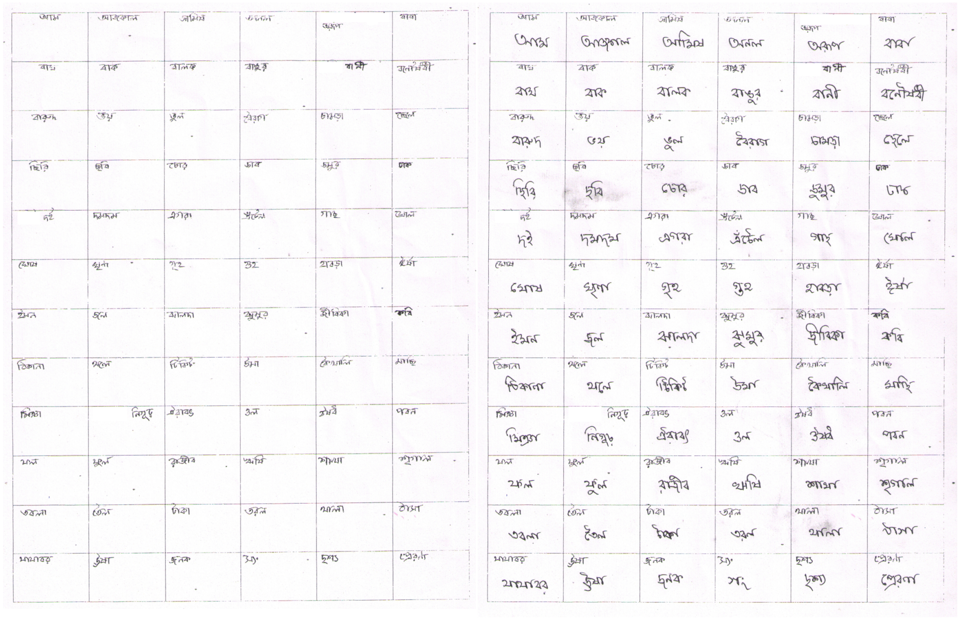
Name: Header Age: Gen: Hand: Set No.:

অ	আ	ই	ঈ	উ	ঊ
ঋ	এ	ঐ	ও	ঔ	
ক	খ	গ	ঘ	ঙ	চ
ছ	জ	ঝ	ঞ	ট	ঠ
ড	ঢ	ণ	ত	থ	দ
ধ	ন	প	ফ	ব	ভ
ম	য	র	ল	ব	শ
ষ	স	হ	ড়	ঢ়	য়
ৎ	ং	ঃ	ঁ	া	ি
ী	ু	ূ	ৃ	ে	ৈ
ো	্	০	১	২	৩
৪	৫	৬	৭	৮	৯

Writing Area

Name: Age: Gen: Hand: Set No.:

Figure 3.1.1: Example of (a) blank data collection form to collect Bangla isolated characters, (b) filled in form consists of Bangla isolated characters



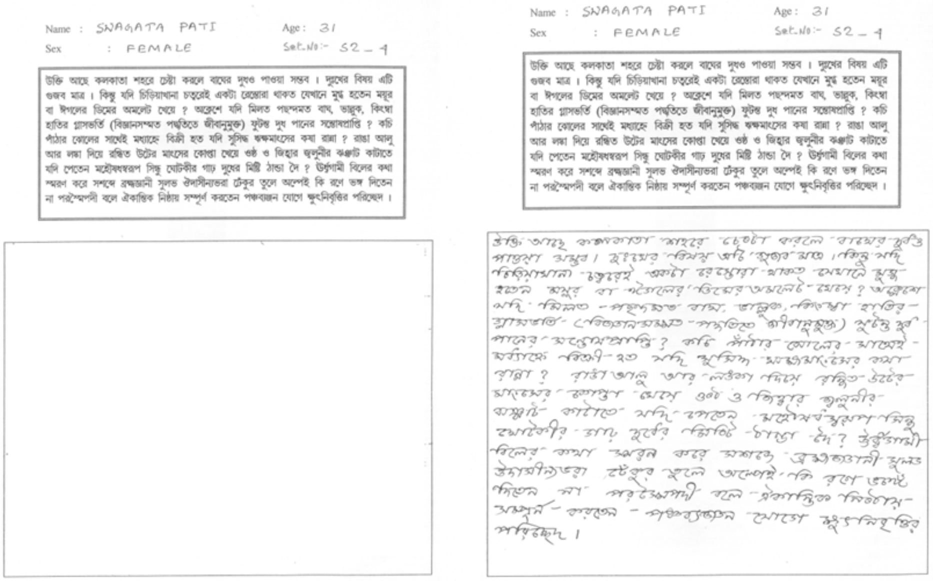
আম	আফজাল	আমিষ	অনল	অরূপ	বাবা
বাঘ	বাক	বালক	বাদুর	বাঁশী	বনৌষধী
বারুদ	ভয়	ভুল	বৈরাগ্য	চামড়া	ছেলে
ছিঁড়ি	ছবি	চোর	ডাব	ডুমুর	ঢাক
দই	দমদম	এগারো	এঁটেল	গাছ	ঘোল
ঘোষ	ঘৃণা	গৃহ	গুহ	হাবড়া	ঈর্ষা
ইমন	জল	ঝালদা	ঝুমুর	জীবিকা	কবি
ঠিকানা	থলে	টিকিট	ঊষা	কৈখালি	মাছি
মিত্রা	নিষ্ঠুর	ঐরাবত	উল	উর্বশী	পবন
ফল	ফুল	সুগ্রীব	ঋষি	শ্যামা	শৃগাল
তবলা	তৈল	টাকা	তরল	থালা	তামা
যাযাবর	ভূষা	জনক	সং	দৃশ্য	প্রেরণা

(a)

আম আম	আফজাল আফজাল	আমিষ আমিষ	অনল অনল	অরূপ অরূপ	বাবা বাবা
বাঘ বাঘ	বাক বাক	বালক বালক	বাদুর বাদুর	বাঁশী বাঁশী	বনৌষধী বনৌষধী
বারুদ বারুদ	ভয় ভয়	ভুল ভুল	বৈরাগ্য বৈরাগ্য	চামড়া চামড়া	ছেলে ছেলে
ছিঁড়ি ছিঁড়ি	ছবি ছবি	চোর চোর	ডাব ডাব	ডুমুর ডুমুর	ঢাক ঢাক
দই দই	দমদম দমদম	এগারো এগারো	এঁটেল এঁটেল	গাছ গাছ	ঘোল ঘোল
ঘোষ ঘোষ	ঘৃণা ঘৃণা	গৃহ গৃহ	গুহ গুহ	হাবড়া হাবড়া	ঈর্ষা ঈর্ষা
ইমন ইমন	জল জল	ঝালদা ঝালদা	ঝুমুর ঝুমুর	জীবিকা জীবিকা	কবি কবি
ঠিকানা ঠিকানা	থলে থলে	টিকিট টিকিট	ঊষা ঊষা	কৈখালি কৈখালি	মাছি মাছি
মিত্রা মিত্রা	নিষ্ঠুর নিষ্ঠুর	ঐরাবত ঐরাবত	উল উল	উর্বশী উর্বশী	পবন পবন
ফল ফল	ফুল ফুল	সুগ্রীব সুগ্রীব	ঋষি ঋষি	শ্যামা শ্যামা	শৃগাল শৃগাল
তবলা তবলা	তৈল তৈল	টাকা টাকা	তরল তরল	থালা থালা	তামা তামা
যাযাবর যাযাবর	ভূষা ভূষা	জনক জনক	সং সং	দৃশ্য দৃশ্য	প্রেরণা প্রেরণা

(b)

Figure 3.1.2: Example of (a) blank Bangla text data collection form, (b) filled in form consists of Bangla text



Name : SNAGATA PATI Age : 31
Sex : FEMALE Set.No:- S2_4

উক্তি আছে কলকাতা শহরে চেষ্টা করলে বাঘের দুধও পাওয়া সম্ভব । দুঃখের বিষয় এটি গুজব মাত্র । কিন্তু যদি চিড়িয়াখানা চত্বরেই একটা রেস্তোরা থাকত যেখানে মুগ্ধ হতেন ময়ূর বা ঈগলের ডিমের অমলেট খেয়ে ? অক্লেশে যদি মিলত পছন্দমত বাঘ, ভাল্লুক, কিংবা হাতির গ্লাসভর্তি (বিজ্ঞানসম্মত পদ্ধতিতে জীবানুমুক্ত) ফুটন্ত দুধ পানের সন্তোষপ্রাপ্তি ? কচি পাঁঠার ঝোলের সাথেই মধ্যাহ্নে বিক্রী হত যদি সুসিদ্ধ ঋষ্যমাংসের কষা রান্না ? রাঙা আলু আর লঙ্কা দিয়ে রন্ধিত উটের মাংসের কোপ্তা খেয়ে ওষ্ঠ ও জিহ্বার জ্বলুনীর ঝঞ্ঝাট কাটাতে যদি পেতেন মহৌষধস্বরূপ সিন্ধু ঘোটকীর গাঢ় দুধের মিষ্টি ঠান্ডা দৈ ? ঊর্ধ্বগামী বিলের কথা স্মরণ করে সশব্দে ব্রহ্মজ্ঞানী সুলভ ঔদাসীন্যভরা ঢেঁকুর তুলে অল্পেই কি রণে ভঙ্গ দিতেন না পরস্মৈপদী বলে ঐকান্তিক নিষ্ঠায় সম্পূর্ণ করতেন পঞ্চব্যঞ্জন যোগে ক্ষুৎনিবৃত্তির পরিচ্ছেদ ।

Name : SNAGATA PATI Age : 31
Sex : FEMALE Set.No:- S2_4

উক্তি আছে কলকাতা শহরে চেষ্টা করলে বাঘের দুধও পাওয়া সম্ভব । দুঃখের বিষয় এটি গুজব মাত্র । কিন্তু যদি চিড়িয়াখানা চত্বরেই একটা রেস্তোরা থাকত যেখানে মুগ্ধ হতেন ময়ূর বা ঈগলের ডিমের অমলেট খেয়ে ? অক্লেশে যদি মিলত পছন্দমত বাঘ, ভাল্লুক, কিংবা হাতির গ্লাসভর্তি (বিজ্ঞানসম্মত পদ্ধতিতে জীবানুমুক্ত) ফুটন্ত দুধ পানের সন্তোষপ্রাপ্তি ? কচি পাঁঠার ঝোলের সাথেই মধ্যাহ্নে বিক্রী হত যদি সুসিদ্ধ ঋষ্যমাংসের কষা রান্না ? রাঙা আলু আর লঙ্কা দিয়ে রন্ধিত উটের মাংসের কোপ্তা খেয়ে ওষ্ঠ ও জিহ্বার জ্বলুনীর ঝঞ্ঝাট কাটাতে যদি পেতেন মহৌষধস্বরূপ সিন্ধু ঘোটকীর গাঢ় দুধের মিষ্টি ঠান্ডা দৈ ? ঊর্ধ্বগামী বিলের কথা স্মরণ করে সশব্দে ব্রহ্মজ্ঞানী সুলভ ঔদাসীন্যভরা ঢেঁকুর তুলে অল্পেই কি রণে ভঙ্গ দিতেন না পরস্মৈপদী বলে ঐকান্তিক নিষ্ঠায় সম্পূর্ণ করতেন পঞ্চব্যঞ্জন যোগে ক্ষুৎনিবৃত্তির পরিচ্ছেদ ।

Figure 3.1.3: Example of (a) blank Bangla text data collection form, (b) filled in form consists of Bangla text

3.2 Data collection process

During data collection, individuals are asked to write the sample isolated characters and sample printed text within the provided area in the forms in their natural speed. Also the writers are given each data collection form (blank) in different days, so that the intra-writing variations along with inter-writer variations can be captured. Individuals of different gender, culture, age groups and educational qualification are considered during data collection. Individuals in the age group of 8-70 years are considered during data collection that include school going children, retired persons, graduate, post graduate students and people with different professionals. These natural variations are incorporated in the database to create standard isolated character and text document databases to be helpful in assessing whether a character/text recognition algorithm is robust in nature or not.

3.3 Database preparation

3.3.1 Isolated character extraction

For character level database, characters are written within the boxes as shown in Fig. 3.1.1. In the next step, the coordinates of boundary of every box have been measured and then character level information (x, y coordinates along with pen status) from all the boxes are stored.

3.3.2 Text extraction

For preparing the word level database, forms as shown in Fig. 3.1.2 and 3.1.3 are used. For the form shown in Fig. 3.1.2, the coordinates of boundary of each of the boxes have been measured and then word level information (x, y coordinates along with pen status) from all the boxes are stored. To extract texts from the forms containing text document as shown in Fig. 3.1.3, a GUI has been designed which is displayed in Fig. 3.3.1. An scheme has been applied to segment text lines from the document. Then words are segmented from the individual text line. The annotated form has been reflected in Fig. 3.3.2. The detailed procedure for text line and word extraction has been reported in the following sub-sections.

3.3.2.1 TEXT LINE EXTRACTION

In order to extract text lines from the document, the procedure finds the horizontal distance between two consecutive strokes, This distance is measured by taking the difference of the x-coordinate values of the i-th stroke and (i+1)-th stroke. If this distance is larger than an experimentally set threshold, then the strokes are considered to be part of different text lines (i.e. if the i-th stroke is the last stroke of j-th line then (i+1)-th stroke is the first stroke of the (j+1)-th line). This fact is shown in Fig. 3.3.3. The experimentally set threshold value is termed as the HORIZONTAL CONSTRAINT.

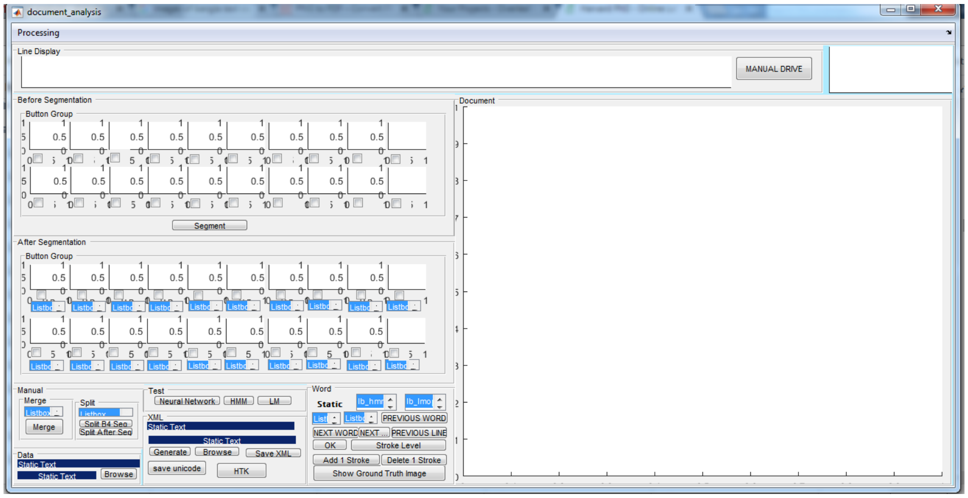
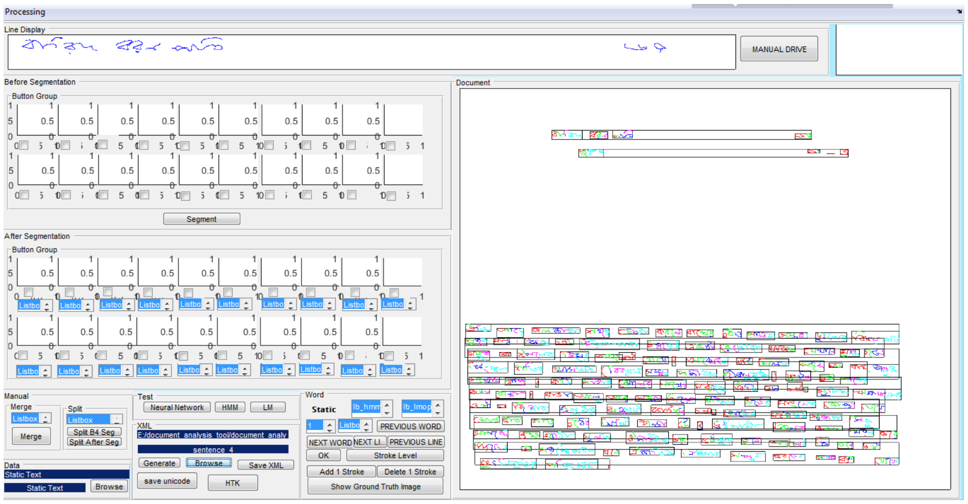


Figure 3.3.1: GUI for the document processing



E:/document_analysis_tool/document_analy

sentence 4

Figure 3.3.2: Example of an annotated document corresponding to image shown in Fig. 3.1.3

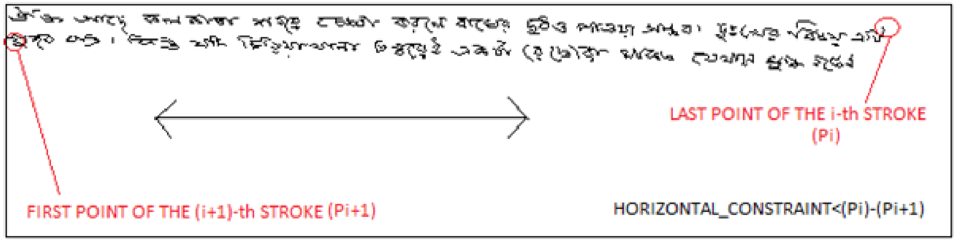

LAST POINT OF THE i-th STROKE
(Pi)
FIRST POINT OF THE (i+1)-th STROKE (Pi+1)
HORIZONTAL_CONSTRAINT<(Pi)-(Pi+1)

Figure 3.3.3: Text line segmentation using HORIZONTAL_CONSTRAINT

The procedure adopted (using HORIZONTAL_CONSTRAINT value) for text line segmentation works well in most of the cases. Exception occurs if a text line starts and the previous text line contains very few strokes as reflected in Fig. 3.3.4. In such cases, the concept of HORIZONTAL_CONSTRAINT fails. To encounter this problem, the VERTICAL_CONSTRAINT concept has been developed. In this case, instead of x-coordinate, the value of y-coordinate is used to differentiate whether two strokes belong to the same text line or not. Fig.3.3.4. illustrates the concept of VERTICAL_CONSTRAINT.

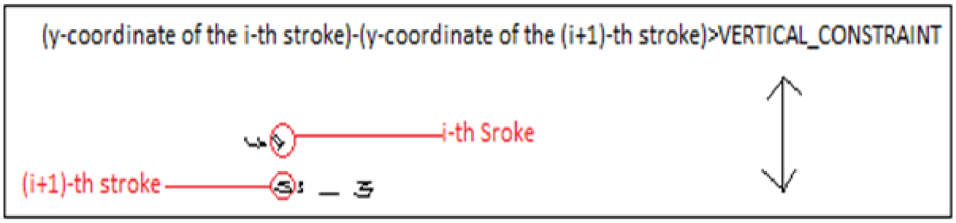

(y-coordinate of the i-th stroke)-(y-coordinate of the (i+1)-th stroke)>VERTICAL_CONSTRAINT
i-th Sroke
(i+1)-th stroke

Figure 3.3.4: Text line segmentation using VERTICAL_CONSTRAINT

Therefore, to extract the text lines from handwritten document, HORIZONTAL and VERTICAL_CONSTRAINT are measured and checked with corresponding threshold values. The algorithm for text line extraction is reported in Algorithm 1, where HORIZONTAL_THRESHOLD and VERTICAL_THRESHOLD are set as 1300 and 50 respectively.

Algorithm 1 : Text line extraction algorithm

```
STEP 1: n=1;//count of text line
STEP 2: j=0;//count of the stroke; even for x-coordinate and odd for y-coordinate
STEP 3: for every $i^{th}$ stroke in the document:
STEP 4: if(stroke=first stroke):
line(n,j)=x;
line(n,j+1)=y;
j=j+2;
STEP 5: else:
STEP 6: if(pixel = start of a stroke):
STEP 7: HORIZONTAL_CONSTRAINT = (x-coordinate of the start of $ith$ stroke)-(x-coordinate of the end of $(i-1)^{th}$ stroke) VERTICAL_CONSTRAINT = (y-coordinate of the start of $ith$ stroke)-(y-coordinate of the end of $(i-1)^{th}$ stroke)
STEP 8: if(HORIZONTAL_CONSTRAINT>1300 || (VERTICAL_CONSTRAINT>50)): //End of a line
STEP 10: line(n,j)=-2;//-2 is the marker for end of a line
STEP 11: line(n,j+1)=-2;//-2 is the marker for end of a line
STEP 12: n=n+1;
STEP 13: line(n,j)=x;
STEP 14:line(n,j+1)=y;
STEP 15: j=j+2
STEP 16: else:
STEP 17: line(n,j)=-1;//-1 is the marker for end of a sub-stroke
STEP 18: line(n,j+1)=-1;//-1 is the marker for end of a sub-stroke
STEP 19: j=j+2;
STEP 20: line(n,j)=x;
STEP 21: line(n,j+1)=y;
STEP 22: j=j+2
STEP 23: else:
STEP 24: line(n,j)=x;
STEP 25: line(n,j+1)=y;
STEP 26: j=j+2
```

3.3.2.2 WORD EXTRACTION

In the next step, words are extracted from each text line by considering the distance between two consecutive strokes. If this distance is greater than HORIZONTAL_CONSTRAINT, then the two strokes belong to separate but consecutive words. The detailed word extraction procedure is shown in Algorithm. 2.

Algorithm 2: Word extraction algorithm

```
STEP 1: for each text line:
STEP 2: nob=0;//number of blanks per text line
STEP 3: word_count=0;
STEP 4: for k1 in range(minX,maxX):
STEP 5: c=0;j=0;
STEP 6: while(line(i,j)!=-2)://till the end of a text line
STEP 7: if(line(i,j)==k1):
STEP 8: c=c+1;
STEP 9: if(j==0 || line(i,j-1)==-1): //start of stroke
STEP 10: if(nob>17):
STEP 11: word(i,word_count,m)=-2;
STEP 12: word(i,word_count,m+1)=-2;
STEP 13: word_count=word_count+1;
STEP 14: nob=0;
STEP 15: m=0;
STEP 16: j1=j;
STEP 17: while(line(i,j1)!=-1):
STEP 18: word(i,word_count,m)=copy(i,j1);
STEP 19: j1++;
STEP 20: m++;
STEP 21: word(i,word_count,m=m+1)=-1;
STEP 22: word(i,word_count,m=m+1)=-1;
STEP 23: nob=0;
STEP 24: j=j+2;
STEP 25: if(c==0):
STEP 26: nob=nob+1;
```

Fig. 3.3.5 illustrates the extracted words for a single text line (words are shown within boxes).

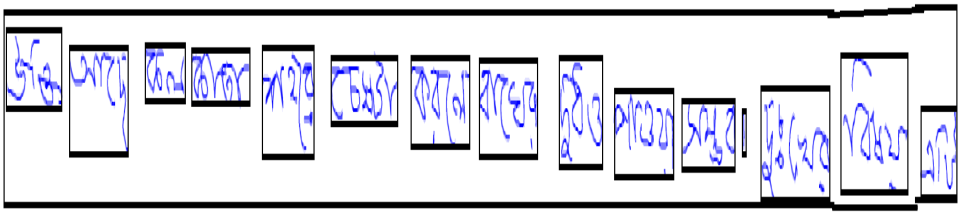


Figure 3.3.5: Extracted words from a text line written in Bangla script

It is to be noted that after applying the automated word extraction technique, there are some words which may not be collected correctly. The experiment analysis reveals that there may be three possible reasons that are mentioned below.

Case 1:

A meaningful word is separated into more than one segment as shown in Fig. 3.3.6 due to putting more gap between strokes of the word.

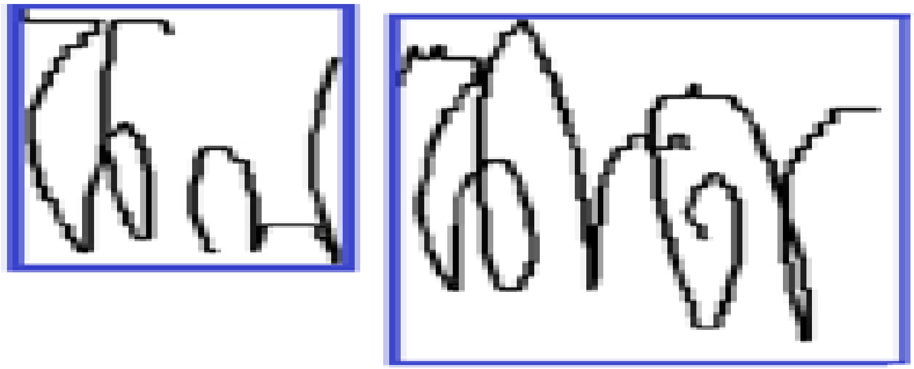


Figure 3.3.6: Single word treated as two different words

Case 2: Two separate words are treated as a single word unit as shown in Fig. 3.3.7 due to putting very small gap between strokes of two different words.

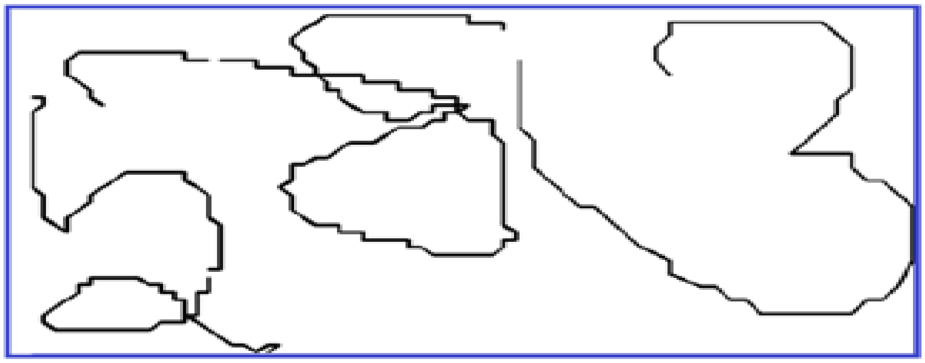


Figure 3.3.7: Two different words treated as a single word

Case 3: Fig 3.3.8 illustrates the third case, where word (here part of a word) within the first box is incomplete and the second box contains the multiple word segments that include the remaining part of the first word and other two valid words.

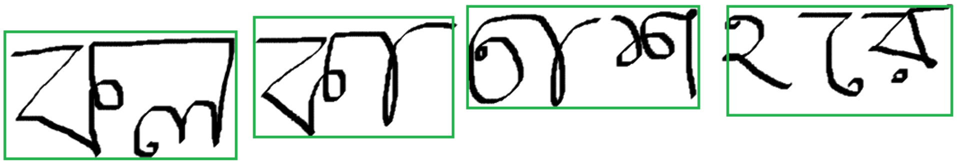


Figure 3.3.8: Example of an error case generated by the proposed word extraction method as reported in Algorithm 2

The above mentioned three cases are handled very carefully. Case 1 is resolved by applying manual merging procedure, in which one particular stroke is concatenated to its immediate previous stroke (as shown in Fig. 3.3.9).

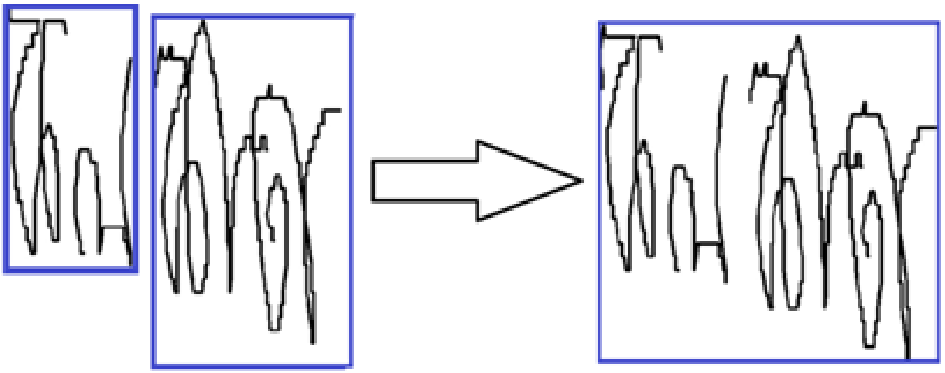


Figure 3.3.9: Manual merge operation performed on two consecutive strokes

Case 2 is managed by implementing manual splitting operation, in which the strokes can be isolated one by one as reflected in Fig. 3.3.10.

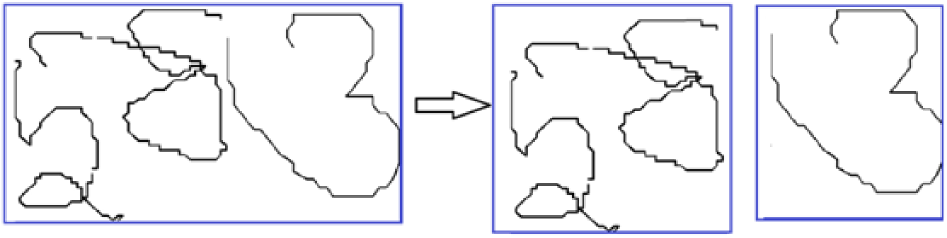


Figure 3.3.10: Manual split operation to form separate words

The step-wise mechanism to handle the third case is mentioned in Fig. 3.3.8.

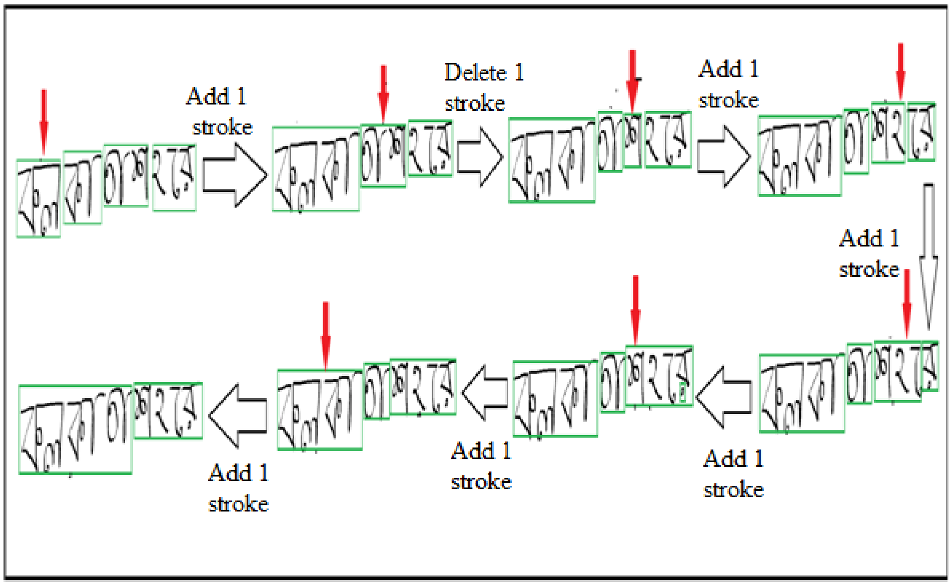


Figure 3.3.11: Steps to handle third case as shown in Fig. 3.3.8

After performing the different merging and splitting procedures, as shown above, the words in the first text line of the document looks like as reflected in Fig. 3.3.12.

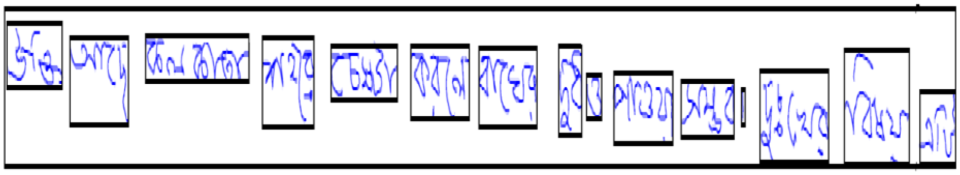


Figure 3.3.12: The words in a text line after performing the different merging and splitting operations

3.4 Ground truth generation for document

This section describes the ground truth generation for the online handwritten documents considered here. The ground truth for a particular document is stored in a XML file format containing the information like: name of the document, number of text lines of the document along with the width of each text line by including start and end coordinates information, number of words in each text line along with the width of each word by including start and end coordinates information, number of strokes in every word along with pixel and class label information of every strokes for a word. The structure of the XML file is depicted in Fig. 3.4.1. The tag information mentioned in XML file (see Fig. 2.18) include following:

- Document

  It is the document for which the ground truth is being made. It contains the information about the name of the document, the dimensions of the document and the number of text lines the document contains.

- Text line

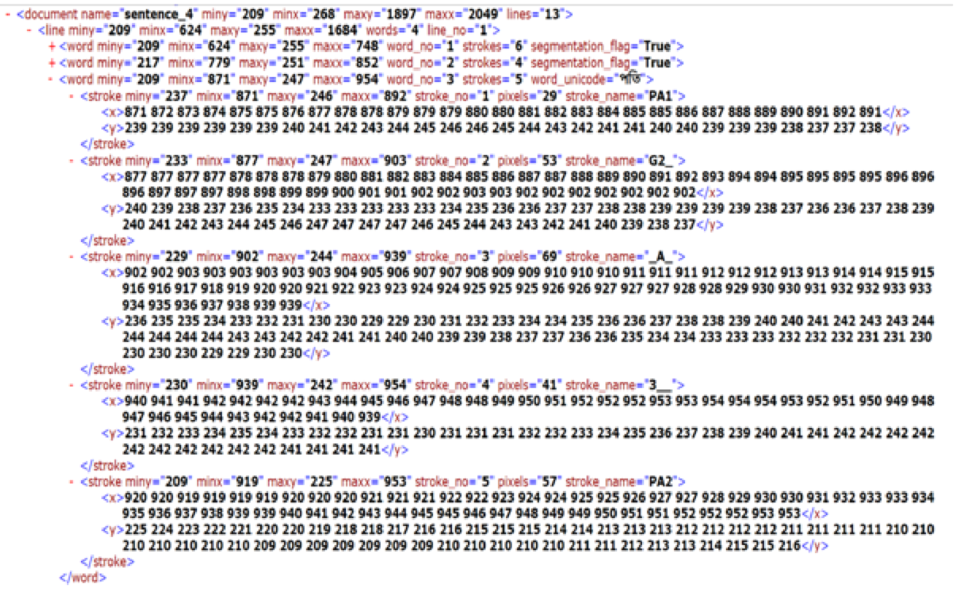
  The text line tag contains the information regarding the dimension of the text lines, text line index, number of words the text line contains.

- Word

  Under each text line tag there are many word tags. The number of word tags is defined in the text line tag by using the attribute "number_of_words". The word tag contains attributes which defines the dimension of the word, stroke information and the Unicode representation of the word.

- Stroke

  Under a Word tag, there are many Stroke tags. The number of stroke tags is defined by the attribute "number_of_strokes" in the Word tag. Stroke tags contain the dimension of the strokes, number of points the stroke contains, stroke name and x and y tags that contain the x-coordinates and the y-coordinates.



```
- <document name="sentence_4" miny="209" minx="268" maxy="1897" maxx="2049" lines="13">
  - <line miny="209" minx="624" maxy="255" maxx="1684" words="4" line_no="1">
    + <word miny="209" minx="624" maxy="255" maxx="748" word_no="1" strokes="6" segmentation_flag="True">
    + <word miny="217" minx="779" maxy="251" maxx="852" word_no="2" strokes="4" segmentation_flag="True">
    - <word miny="209" minx="871" maxy="247" maxx="954" word_no="3" strokes="5" word_unicode="পতি">
      - <stroke miny="237" minx="871" maxy="246" maxx="892" stroke_no="1" pixels="29" stroke_name="PA1">
          <x>871 872 873 874 875 875 876 877 878 878 879 879 879 880 880 881 882 883 884 885 885 886 887 888 889 890 891 892 891</x>
          <y>239 239 239 239 239 239 240 241 242 243 244 245 246 246 245 244 243 242 241 241 240 240 239 239 239 238 237 237 238</y>
        </stroke>
      - <stroke miny="233" minx="877" maxy="247" maxx="903" stroke_no="2" pixels="53" stroke_name="G2_">
          <x>877 877 877 877 878 878 878 879 880 881 882 883 884 885 886 887 887 888 889 890 891 892 893 894 894 895 895 895 895 896 896
            896 897 897 897 898 898 899 899 900 901 901 902 902 903 903 902 902 902 902 902 902 902</x>
          <y>240 239 238 237 236 235 234 233 233 233 233 233 234 235 236 236 237 237 238 238 239 239 239 239 238 237 236 236 237 238 239
            240 241 242 243 244 245 246 247 247 247 247 246 245 244 243 243 242 241 240 239 238 237</y>
        </stroke>
      - <stroke miny="229" minx="902" maxy="244" maxx="939" stroke_no="3" pixels="69" stroke_name="_A_">
          <x>902 902 903 903 903 903 903 903 904 905 906 907 907 908 909 909 910 910 910 911 911 911 912 912 912 913 913 914 914 915 915
            916 916 917 918 919 920 920 921 922 923 923 924 924 925 925 925 926 926 927 927 927 928 928 929 930 930 931 932 932 933 933
            934 935 936 937 938 939 939</x>
          <y>236 235 235 234 233 232 231 230 230 229 229 230 231 232 233 234 234 235 236 236 237 238 238 239 240 240 241 242 243 243 244
            244 244 244 244 243 243 242 242 241 241 240 240 239 239 238 237 237 236 236 235 234 234 233 233 233 232 232 232 231 231 230
            230 230 230 229 229 230 230</y>
        </stroke>
      - <stroke miny="230" minx="939" maxy="242" maxx="954" stroke_no="4" pixels="41" stroke_name="3__">
          <x>940 941 941 942 942 942 942 943 944 945 946 947 948 948 949 950 951 952 952 952 953 953 954 954 954 953 952 951 950 949 948
            947 946 945 944 943 942 942 941 940 939</x>
          <y>231 232 233 234 235 234 233 232 232 231 231 230 231 231 231 232 232 233 234 235 236 237 238 239 240 241 241 242 242 242 242
            242 242 242 242 242 242 241 241 241 241</y>
        </stroke>
      - <stroke miny="209" minx="919" maxy="225" maxx="953" stroke_no="5" pixels="57" stroke_name="PA2">
          <x>920 920 919 919 919 919 920 920 920 921 921 921 922 922 923 924 924 925 925 926 927 927 928 929 930 930 931 932 933 933 934
            935 936 937 938 939 939 940 941 942 943 944 945 945 946 947 948 949 949 950 951 951 952 952 952 953 953</x>
          <y>225 224 223 222 221 220 220 219 218 218 217 216 216 215 215 215 214 214 213 213 213 212 212 212 212 211 211 211 211 210 210
            210 210 210 210 210 209 209 209 209 209 209 209 210 210 210 210 210 211 211 212 213 213 214 215 215 216</y>
        </stroke>
      </word>
```

Figure 3.4.1: Sample XML structure representing ground truth of an online handwritten document

After applying text line and word extraction methodologies, extracted words undergo through automated followed by a manual segmentation procedure to generate valid strokes. These strokes are then labelled by class names termed as "stroke_name" attribute as reflected in Fig. 3.4.1. Then a procedure has been applied to compute the value of "word_unicode" by using "stroke_name" information. The following sub-sections describe the techniques in details.

3.4.1 Stroke extraction from words

For each word in a text line, strokes are extracted through automated procedure by implementing word segmentation technique which is described later in section 6.1.1 of Chapter 6 in this thesis. After automated segmentation, there may be situations where the target word is not segmented accurately as reflected in Fig. 3.4.2 (3^{rd} stroke is under-segmented).

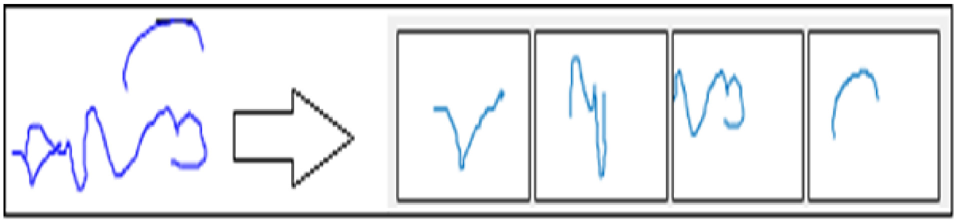


Figure 3.4.2: Under segmented word sample obtained by using automatic segmentation procedure

Manual splitting procedure is applied to handle this case by clicking the mouse at the desired position on the under-segmented stroke. The stroke will be split based on the clicking point as shown in Fig. 3.4.3.

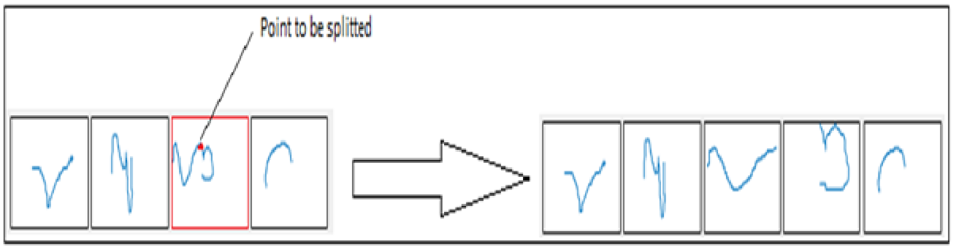


Figure 3.4.3: Example of manual Splitting operation

After collecting different type of stroke patterns, it has been found that in Bangla script without having complex and compound characters, there exists a total of 59 unique stroke patterns/classes as mentioned in Fig. 6.1.1 of chapter 6.

3.4.2 Stroke Selection and Unicode generation of words

In this step, all the constituent strokes of a particular word are assigned with appropriate stroke-class labels as shown in Fig. 3.4.4. The selected label is then stored as the "stroke_name" attribute value in the stroke tag as depicted in Fig. 3.4.1.

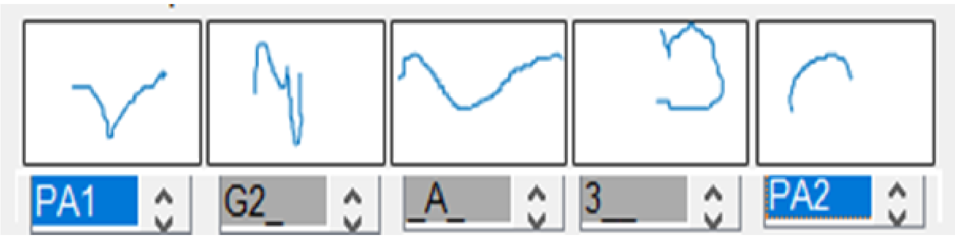


Figure 3.4.4: Selection of stroke -class labels for a particular word sample

After selection of class labels for all the constituent strokes of a word, a procedure has been adopted to generate Unicode from the sequence of stroke-class label representations. The problem arises because stroke-class labels do not have any Unicode representation. Hence, a procedure is required to form characters and modifiers from the sequence of class label representations. A look up table has been designed in this regard that contains all possible different stroke combinations for every character and modifier. Character and modifier symbols are then formed by applying longest mach procedure mentioned in Algorithm 3. Finally, the Unicode of a word sample is formed by considering the Unicode of constituent characters and modifier symbols.

The constituent strokes of the word are firstly rearranged and then the longest match procedure is implemented as mentioned in Algorithm 3 to find the Unicode of corresponding characters and modifier symbols. The rearrangement of strokes are based on their positional information. This is how the delayed strokes are placed in their appropriate positions. The longest match includes the process of combining strokes from left to right till a valid character/modifier is formed. The process breaks at the point where no further valid character/modifier can be formed. This process repeats till there is no more stroke in the sequence.

For example, consider the sequence of stroke-class labels "PA1 G2 _A_ PA2 3__" for a word. Fig. 3.4.5 illustrates the scenario pictorially.

PA1=Invalid

PA1 G2=Valid

PA1 G2 _A_=Invalid

Algorithm 3: Formation of Unicode of online handwritten Bangla words

```
STEP 1: i=1;
STEP 2: track=1;
STEP 3: seq=[]
STEP 4: unicode=[]
STEP 5: While i<=(length of the sequence)
STEP 6: seq.push(i);
STEP 7: if(pattern(seq)exist)
STEP 8: unicode[track]=getunicode(seq);
STEP 9: else
STEP 10: track++;
STEP 11: seq.removeAll();
STEP 12: end if
STEP 13: i++;
STEP 14: End While
```

_A_=Invalid

_A_ PA2=Valid

_A_ PA2 3__=Invalid

3__=Valid

So the stroke-class labels are grouped into three components: (PA1 G2)(_A_ PA2) (3__)

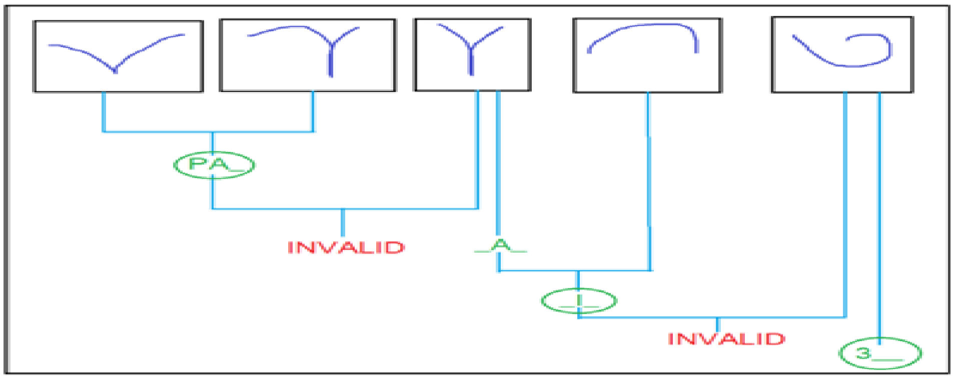


Figure 3.4.5: Character and modifier symbol formation by using Algorithm 3

3.4.3 Unicode

Unicode originally implied that the encoding was UCS-2 and it initially did not make any provisions for characters outside the BMP (U+0000 to U+FFFF). When it became clear that more than 64K characters would be needed for certain special applications (historic alphabet and ideographs, mathematical and musical typesetting, etc.), Unicode was turned into a sort of 21-bit character set with possible code points in the range U-00000000 to U-0010FFFF. The 2×1024 surrogate characters (U+D800 to U+DFFF) were introduced into the BMP to allow 1024×1024 non-BMP characters to be represented as a sequence of two 16-bit surrogate characters. This is how UTF-16 was born, which represents the extended "21-bit" Unicode in a way backwards compatible with UCS-2.

3.4.4 Unicode Formation

3.4.4.1 UTF-8

In addition to all that, UTF-8 was introduced to provide an ISCII backwards compatible multi-byte encoding. The definitions of UTF-8 in UCS and Unicode differed originally slightly, because in UCS, up to 6-byte long UTF-8 sequences were possible to represent characters up to U-7FFFFFFF, while in Unicode only up to 4-byte long UTF-8 sequences are defined to represent characters up to U-0010FFFF. (The difference was in essence the same as between UCS-4 and UTF-32.)

No endianess is implied by the encoding names UCS-2, UCS-4, UTF-16, and UTF-32, though ISO 10646-1 says that Big-endian should be preferred unless otherwise agreed. It has become customary to append the letters "BE" (Big-endian, high-byte first) and "LE" (Little-endian, low-byte first) to the encoding names in order to explicitly specify a byte order. UTF-8 has the following properties:

1. UCS characters U+0000 to U+007F (ASCII) are encoded simply as bytes 0x00 to 0x7F (ASCII compatibility). This means that files and strings which contain only 7-bit ASCII characters have the same encoding under both ASCII and UTF-8.

2. All UCS characters >U+007F are encoded as a sequence of several bytes,

each of which has the most significant bit set. Therefore, no ASCII byte (0x00-0x7F) can appear as part of any other character.

3. The first byte of a multi-byte sequence that represents a non-ASCII character is always in the range 0xC0 to 0xFD and it indicates how many bytes followed for this character. All further bytes in a multi-byte sequence are in the range 0x80 to 0xBF. This allows easy resynchronization and makes the encoding stateless and robust against missing bytes.

4. All possible 231 UCS codes can be encoded.

UTF-8 encoded characters may theoretically be up to 6 bytes long; however, 16-bit BMP characters are only up to three bytes long.

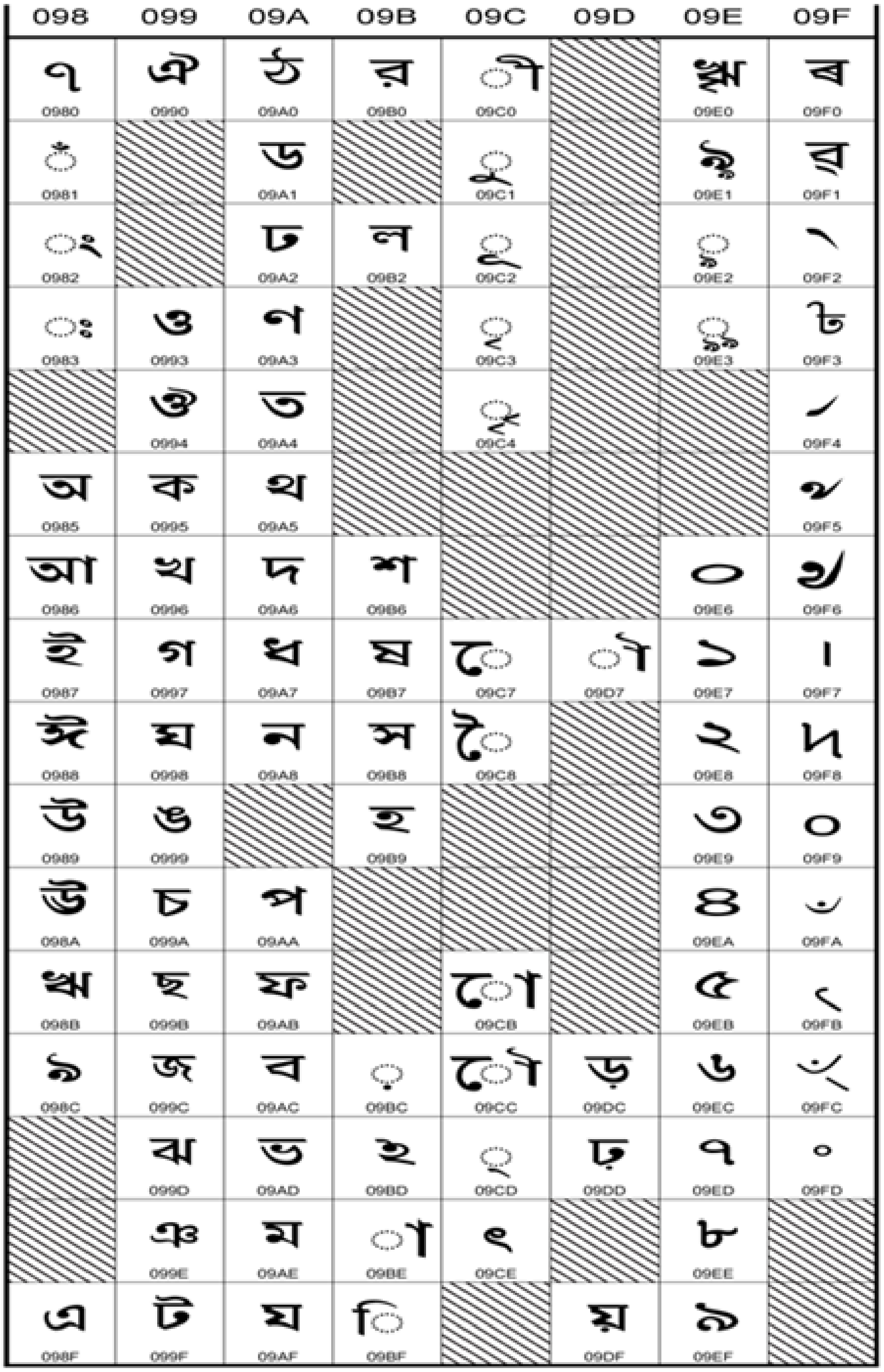
Unicode (and UCS-4) to UTF-8 Mapping table is shown in Table 3.4.1.

Table 3.4.1: Unicode (and UCS-4) to UTF-8 Mapping

U-00000000 – U-0000007F:	0xxxxxxx
U-00000080 – U-000007FF:	110xxxxx 10xxxxxx
U-00000800 – U-0000FFFF:	1110xxxx 10xxxxxx 10xxxxxx
U-00010000 – U-001FFFFF:	11110xxx 10xxxxxx 10xxxxxx 10xxxxxx
U-00200000 – U-03FFFFFF:	111110xx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx
U-04000000 – U-7FFFFFFF:	1111110x 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx 10xxxxxx

3.4.5 UNICODE FOR BANGLA ALPHABET

In case of Bangla alphabet, Unicode is in the range of U-0980 to U-09FF. Fig. 3.4.6 gives the Unicode for each letter in Bangla alphabet:



	098	099	09A	09B	09C	09D	09E	09F
0	ঀ 0980	ঐ 0990	ঠ 09A0	র 09B0	◌ী 09C0	▨	ৠ 09E0	ৰ 09F0
1	◌ঁ 0981	▨	ড 09A1	▨	◌ু 09C1	▨	ৡ 09E1	ৱ 09F1
2	◌ং 0982	▨	ঢ 09A2	ল 09B2	◌ূ 09C2	▨	◌ৢ 09E2	৲ 09F2
3	◌ঃ 0983	ও 0993	ণ 09A3	▨	◌ৃ 09C3	▨	◌ৣ 09E3	৳ 09F3
4	▨	ঔ 0994	ত 09A4	▨	◌ৄ 09C4	▨	▨	৴ 09F4
5	অ 0985	ক 0995	থ 09A5	▨	▨	▨	▨	৵ 09F5
6	আ 0986	খ 0996	দ 09A6	শ 09B6	▨	▨	০ 09E6	৶ 09F6
7	ই 0987	গ 0997	ধ 09A7	ষ 09B7	◌ে 09C7	◌ৗ 09D7	১ 09E7	৷ 09F7
8	ঈ 0988	ঘ 0998	ন 09A8	স 09B8	◌ৈ 09C8	▨	২ 09E8	৸ 09F8
9	উ 0989	ঙ 0999	▨	হ 09B9	▨	▨	৩ 09E9	৹ 09F9
A	ঊ 098A	চ 099A	প 09AA	▨	▨	▨	৪ 09EA	৺ 09FA
B	ঋ 098B	ছ 099B	ফ 09AB	▨	◌ো 09CB	▨	৫ 09EB	৻ 09FB
C	ঌ 098C	জ 099C	ব 09AC	◌় 09BC	◌ৌ 09CC	ড় 09DC	৬ 09EC	ৼ 09FC
D	▨	ঝ 099D	ভ 09AD	ঽ 09BD	◌্ 09CD	ঢ় 09DD	৭ 09ED	৽ 09FD
E	▨	ঞ 099E	ম 09AE	◌া 09BE	ৎ 09CE	▨	৮ 09EE	▨
F	এ 098F	ট 099F	য 09AF	◌ি 09BF	▨	য় 09DF	৯ 09EF	▨

Figure 3.4.6: Unicode for Bangla alphabet

3.4.6 Mapping from UNICODE to UTF-8 for Bangla Alphabet

As, Unicode range for Bangla alphabet is from U-0980 to U-09FF, to convert Unicode of a Bangla letter to its UTF-8 format, 3 bytes (i.e. 24 bits) are required. The UTF-8 format for all the Bangla letters/symbols are:

Table 3.4.2: UTF-8 format for all the Bangla letters/symbols

অ	224 166 133	ঙ	224 166 153	প	224 166 170	◌ি	224 166 191
আ	224 166 134	চ	224 166 154	ফ	224 166 171	◌ী	224 167 128
ই	224 166 135	ছ	224 166 155	ব	224 166 172	◌ু	224 167 129
ঈ	224 166 136	জ	224 166 156	ভ	224 166 173	◌ূ	224 167 130
উ	224 166 137	ঝ	224 166 157	ম	224 166 174	◌ৃ	224 167 131
ঊ	224 166 138	ঞ	224 166 158	য	224 166 175	◌ে	224 167 135
ঋ	224 166 139	ট	224 166 159	র	224 166 176	◌ৈ	224 167 136
ঌ	224 166 140	ঠ	224 166 160	ল	224 166 178	◌ো	224 167 139
এ	224 166 143	ড	224 166 161	শ	224 166 182	◌ৌ	224 167 140
ঐ	224 166 144	ঢ	224 166 162	ষ	224 166 183	◌্	224 167 141
ও	224 166 147	ণ	224 166 163	স	224 166 184	ৎ	224 167 142
ঔ	224 166 148	ত	224 166 164	হ	224 166 185	ড়	224 167 156
ক	224 166 149	থ	224 166 165	◌ঁ	224 166 129	ঢ়	224 167 157
খ	224 166 150	দ	224 166 166	◌ং	224 166 130	য়	224 167 159
গ	224 166 151	ধ	224 166 167	◌ঃ	224 166 131		
ঘ	224 166 152	ন	224 166 168	◌া	224 166 190		

3.5 Statistical information about collected Data

This section describes the detailed statistics of two different types of databases developed here. Towards designing the isolated character database, 100 volunteers have contributed, and from each individual three sets of characters of Bangla alphabet have been collected. Though initially 135 volunteers agreed to contribute for development of this character database, but only 100 volunteers have completed the required three sets. The database contains approximately 12,000 alphabet, 3,300 vowels, 3,000 vowel modifiers and 3,000 numerals. Each collected dataset contains ordered pixel points having x and y coordinates along with pen up/down information saved as .dat format with a particular naming convention: $< 4\ \mathit{digit\ serial\ number}\ (\mathit{Writer\_id}) > \_ < 2\ \mathit{digit\ serial\ number}\ (\mathit{Set\_id}) >$. For example, the first isolated digitized form would be named as '0000_01', where '0000' stands for writer id of the first writer, '01' stands for the first set of writings by that writer and these two fields are separated by the '_' sign.

In the present work, information about the online Bangla words, collected here, is reported by using word collection form and document form as mentioned in Fig. 3.1.2-3.1.3 respectively. 130 people have donated texts in the form given to them (one sample is shown in Fig. 3.1.2). As this form contains 50 different words, hence a total of 6500 word samples are collected.

The text database using form shown in Fig. 3.1.3 is built by taking the data from 50 writers with variable number of datasets per writer. Emphasis was given on variable number of dataset for each writer due to the fact that in real world scenarios it is quite hard to get same number of handwriting samples from every writer. Therefore, one of the main challenges in real world situation is to identify writer of a sample when the number of samples per writer is uneven. This is the main reason for collecting random samples from random number of writers in order to create the database. The database contains 5 samples from 26 writers where 3 samples are of simple text and 2 samples are of complex text. Another 13 writers have written 2 simple texts each. Out of remaining 11 writers, 7 writers have written 4 samples of simple and complex text, 2 writers have written 7 samples of simple and complex text and last 2 writers have written 8 samples for both the categories. The document containing simple and complex texts, have 76 and 80 words respectively. Hence, in this procedure a total of 162 (26 X 3 + 13 X 2 + 7 X 4 + 2 X 7 + 2 X 8) sets of simple and 110 (26 X 2 + 7 X 4 + 2 X 7 + 2 X 8) set of complex text are collected. Here also each dataset is saved in .dat format and stored using the same naming convention as for isolated datasets. Table 3.5.1 shows distribution of datasets with approximate number of total words available in the database.

Table 3.5.1: Detailed report about the distribution of datasets for writers

Number of Writers	**Basic Samples (1 set) (maximum 8 copies per writer)**	**Complex Samples (1 set) (maximum 8 copies per writer)**	**Total Samples (simple + complex)**	**Words Present (approx.)**
26	78	52	130	10088
13	26	-	26	1976
7	28	28	56	4368
2	14	14	28	2184
2	16	16	32	2496
Total=50	Total=162	Total=110	Total=272	Total=21112

4
Feature Extraction

Feature extraction is the process of generating input to the pattern recognition system. Many methods have been developed over the years to generate powerful features to classify the patterns under consideration. The level at which these features are extracted determines the amount of necessary pre-processing required and may influence the amount of error introduced into the feature extracted. Features many be represented as continuous, discrete, or discrete binary variables. During the features extraction phase patterns are measured according to the procedure adopted. A measurement is the value of some quantifiable property of a pattern. A feature is a function of one or more measurements, computed so that it quantifies some significant characteristic of this pattern. This process generates a set of features that, together forms the feature vector. Mostly, all the supervised learning system pursue three basic methodologies: (i) suitable feature extraction for classification, (ii) selection of suitable classification method and (iii) evaluation of performance of the classification system. Thus, for any automatic text recognition system, some good feature extraction methodologies are needed that can be efficiently used for the recognition of texts. In general, there are two categories of features: application dependent and generalized. Usually, application dependent features are widely used than generalized features, as most of the time application dependent features achieve higher accuracies than generalized features. From the works mentioned in [22, 41, 82, 98, 103], it is found that combination of different features (dependent/general) produce better accuracies. In this chapter, three categories of handcrafted (structural, topological and stroke based) feature techniques along with deep learning based approach are discussed. Sometimes raw data needs to be pre-processed before using feature extraction technique. Therefore, some pre-processing procedures are also described here. This chapter furthermore describes classification schemes reported in this thesis.

4.1 Pre-processing

For online handwriting recognition, data samples are represented as a collection of pen positions p_t , where t ranges from 1 to M. pt describes the position of the pen having x and y coordinates (x_t and y_t) with pen-up or pen-down status. Here, M represents total number of pen positions for each of the character samples. After a deep analysis on the collected data, three observations are found; presence of duplicate points, variance in number of pixels for the same pattern due to writing speed variation of individuals and size variations of same writing pattern. Speed variation implies people write with less speed contribute more number of pixels for the same pattern information. Again, people can write the same information with different sizes. During the pre-processing step, duplicate or repeated points must be removed to avoid redundancy in all the cases. Sometimes data are pre-processed to cope up with speed and size variations by applying point and size normalization techniques.

- Duplicate points removal:

If two consecutive pen points are represented by $p_j$and p_k respectively, then j^{th} point p_j is retained with respect to k^{th} point p_k if

$$x^2 + y^2 > m^2$$

Where, x = x_j - x_k and y = y_j - y_k . The value of m is set to zero in order to remove all redundant points.

- Point normalization:

  To cope with speed variation this technique is used. In this procedure firstly, a new sequence of points is generated (using 4-connected Bresenham algorithm) where the points are unit distance apart. Then, this sequence is normalized by λ number of points that retain the structure of the character. The value of λ may vary for different feature extraction schemes. Fig. 4.1.1 (a-c) highlights the original sample, sample where the points are rearranged with unit distance apart and finally sample with normalized n points.

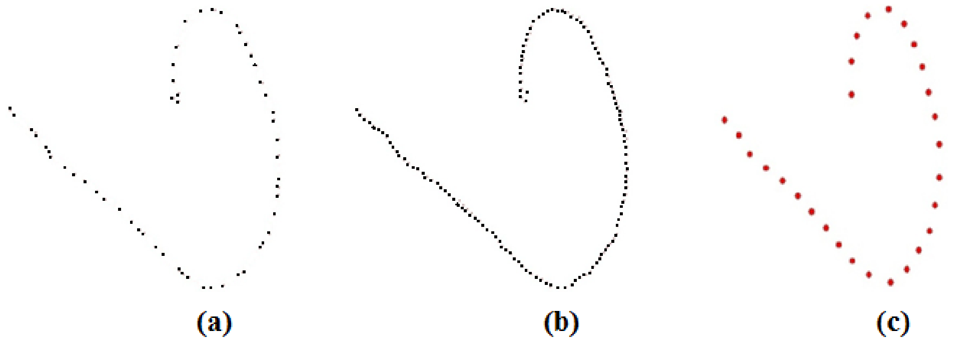


Figure 4.1.1: (a) Original character sample, (b) rearranged with unit distance apart and (c) finally sample with normalized n points

- Point normalization:

  To handle size variation this technique scales the normalized points within a fixed window of size M x N. The values of M and N may vary for different feature extraction techniques. Fig. 3.2 (a-c) shows two sample characters normalized with n points (a-b) and corresponding scaled image (c).

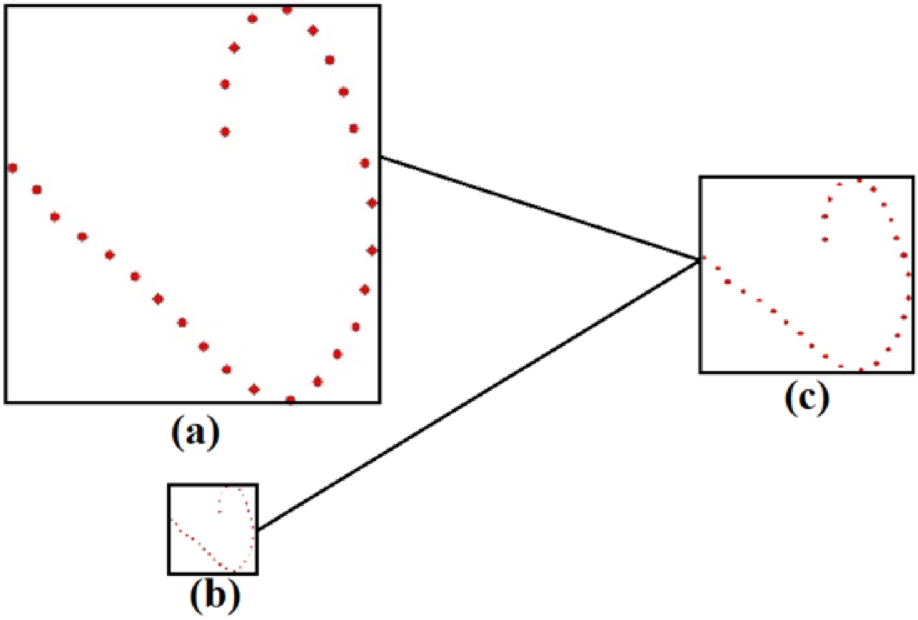


Figure 4.1.2: Images with different sizes scaled into M x N window

4.2 Feature extraction methods

4.2.1 Feature extraction for character recognition (holistically)

Feature is one of the prime aspects used to distinguish patterns. Extraction techniques of these features are dependent on the objective of the system. In the following subsections few well established feature extraction methods are discussed in detail that are used in various recognition works on isolated Bangla characters [98–100, 102, 103] and strokes [101, 104, 105].

4.2.1.1 Distance based feature:

Closer look on the Bangla character set reveals that there are some characters which are structurally almost similar. To cope up with this, we need some local information in certain regions of the character image which would ultimately produce the strong discriminating features for recognition of such characters. A different approach towards Distance based feature for Uyghur character is presented in [99]. This technique is customized here for the recognition of Bangla character. In this feature extraction technique, characters are considered as a single entity irrespective to the number of constituent strokes and divided into N hypothetical segments. As to represent a single segment two points are required and the character shape is divided into N segments, hence, N+1 sample pen points pi are required, where i varies from 1 to N+1. The distance from each point to every other points are computed and these distance values are considered as the features for character recognition purpose. As it a quite difficult to decide the optimal number of segmentation needed for best recognition score, experiment has been performed with different possible values of N (6,8,10,16,32,48 and 55 respectively) to get optimum number of segments. Higher value of N indicates the more the number of segments i.e. it would provide more detailed/close view about the character sample. The steps of the distance based feature are given in Algorithm 4.

Algorithm 4: Algorithm used to compute distance based features

```
STEP 1: START
STEP 2: i=1;
STEP 3: j=i+1;
STEP 4: compute the distance d between $p_i$ and $p_j$;
STEP 5: j=j+1;
STEP 6: if (j<=N+1) then goto STEP 4, otherwise goto STEP 7;
STEP 7: i=i+1;
STEP 8: if (i<N+1) goto STEP 3,otherwise goto STEP 9;
STEP 9: END.
```

If 8 (i.e. N=8) segmented character is considered, then total number of pen points is 9, from p1 to p9, which is shown in Fig. 3.3(a-c). From the Algorithm mentioned in Algorithm 3.1, it is clear that there will be total 8 iterations. In first iteration, distances are computed from pen point p1 to p2, p3, p4, p5, p6, p7, p8 and p9 respectively. Therefore, 8 feature values are generated in this iteration. In the second iteration, distances are calculated from pen point p2 to all other pen points except p1, because it is already covered in the first iteration. Thus it produces 7 features values in the second iteration. Following the similar approach, in the last iteration only one distance value is computed from pen point p8 to p9. Thus, a total of 36 (i.e. 8+7+6+5+4+3+2+1) feature values are generated for an 8 segmented Bangla character sample. In general, for an N segmented character shape, N (N+1)/2 number of feature values are produced by using the distance based feature extraction procedure. In Fig. 4.2.1(a-c), red points indicate pen points and black lines denote the distances between pen points.

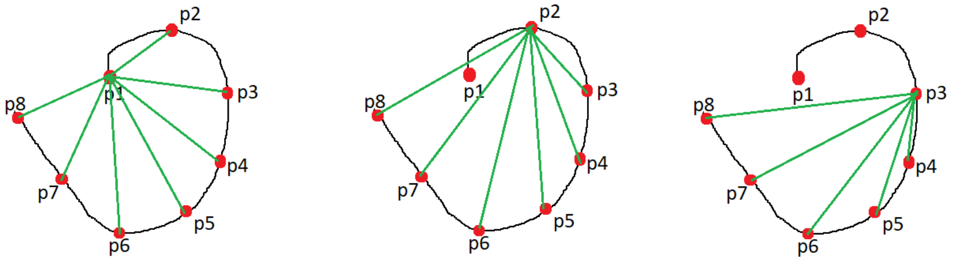


Figure 4.2.1: (a-c) Distance calculation from one pen point to the rest considering 8 points in the sample handwritten character

4.2.1.2 Hausdorff Distance based Feature[102]

4.2.1.2.1 Hausdorff Distance (HD) and Directed Hausdorff Distance (DHD)

In this section, an expedient shape based feature extraction approach based on HD has been described for the recognition of online handwritten Bangla basic characters. Generally forward HD h(X, Y), from set X to set Y is computed by using maxmin function as defined in Eq 4.1.

$$h(X, Y) = max_{x \varepsilon X} \left\{ min_{y \varepsilon Y} \left\{ d(x, y) \right\} \right\} \tag{4.1}$$

Where a and b represent the points of set X and set Y respectively and d(x, y) is any metric in between the points. In the current work, d(x, y) is taken as the Euclidian distance between x and y. The same procedure is applied to calculate the distance h(Y, X) from set Y to X, which is known as backward HD. The calculation of forward or backward HDs between any two sets X and Y having different number of points is shown in Algorithm 5.

Algorithm 5: Calculation of Forward/backward HD between two sets of points

```
STEP 1: k = 0
STEP 2: for every point $x_i$ of X,
STEP 2.1: dist = Infinity ;
STEP 2.2: for every point $y_j$ of Y
        $d_{ij}$ = d ($x_i$ , $y_j$)
        if $d_{ij}$< dist then
        dist = $d_{ij}$
STEP 2.3: if dist > k then
        k = dist
```

For HD based feature calculation, computation of HD values between zones reflects the asymmetric property as introduced by maxmin function in Eq. 4.1. Due to this asymmetry, values of h(X, Y) and h(Y, X) may not be same always. HD measurement for set X and Y, takes maximum value from h(X, Y) and h(Y, X) as shown in Eq. 4.2. Algorithm 6 describes the way of finding HD values among different zones when sample character is divided into N rectangular zones. If we closely observe the Algorithm 6, then it can be seen that this HD based procedure basically tries to match the points of two zones/sets, or more specifically, it attempts to find the similarities of shapes present in different zones. In the present work, as described in Algorithm 6, the sample character is divided into N rectangular zones (see Fig. 3.4. where N=16, and black coloured points describe the contributed pixels in the respective zones). It can be observed that some of the zones may not have any data pixel; it generally depends on shape of the sample character. Please note that for this distance calculation strategy, only maximum of h(X, Y) and h(Y, X) acts as feature component. From Algorithm 5, this can be noticed that for the entire character sample N*(N-1)/2 [i.e. (N-1)+(N-2)+....+1] number of distance values have been produced as features. Therefore, for HD based feature computation lengths of 6, 36, 120 and 300 element feature vectors have been produced when a character image is divided into 4, 9, 16 and 25 rectangular zones respectively.

$$h(X,Y) = max\left\{h(X,Y), h(Y,X))\right\} \tag{4.2}$$

Algorithm 6: HD calculation for any character sample segmented into N number of rectangular zones

```
STEP 1: Divide the character sample into N rectangular zones
STEP 2: for p=1 to N-1 do
STEP 2.1: for q=p to N do
        if p!=q do
        find HD between $zone_p$ and $zone_q$
        end for
        end for
```

For DHD based feature calculation, the character sample is divided into N rectangular zones same as described in HD based technique. Then DHD values are computed from every zone to all other zones. Algorithm 7 specifies the DHD based feature calculation procedure when any character sample is divided into N number of zones. It is worth mentioning that the distance values of h(X, Y) and h(Y, X) of Eq. 4.1, also known as forward and backward HDs respectively, are not

Algorithm 7: DHD calculation for any character sample segmented into N rectangular zones

```
STEP 1: Divide the character sample into N rectangular zones
STEP 2: for p=1 to N-1 do
STEP 2.1: for q=1 to N do
        if p!=q do
        find DHD between $zone_p$ and $zone_q$
        end for
        end for
```

equal most of the time. Hence, in this work DHD values have been computed from each zone to all other zones to get the exclusivity of the directional distances. As spread of data pixels for different characters are dissimilar in different zones because of their shape structures, this feature extraction approach can be assumed to work well to identify the shapes which are similar/dissimilar in nature. While considering a particular zone and following Algorithm 7, N-1 number of DHD measurements have been generated. As a result a total of N*(N-1) number of DHD values have been produced considering the entire character image which serve as feature values for the present work. In the current experiment, different values of N such as 4, 9, 16 and 25 are taken to profoundly achieve the discriminatory local information. Hence, lengths of 12, 72, 240, 600 element feature vectors have been produced corresponding to the values of N which is of double length compared to HD based feature calculation. Thus features produced in HD based feature calculation are basically a subset of the features produced when DHD based procedure is applied.

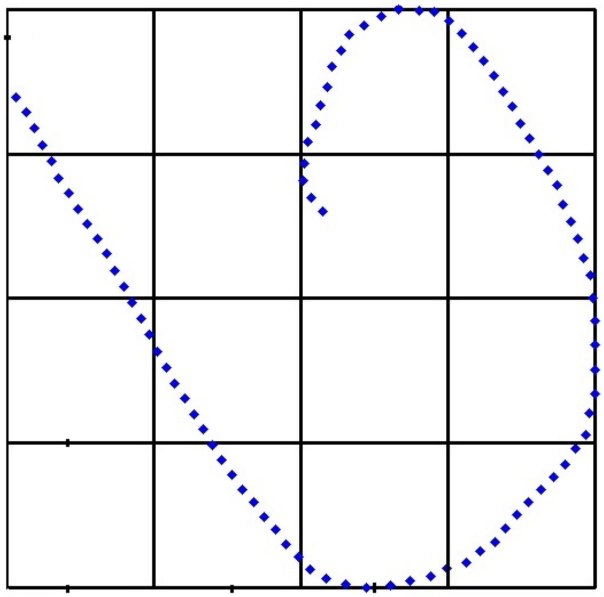


Figure 4.2.2: Sample character when segmented into 16 rectangular zones

4.2.1.3 CG -based Circle Feature

In this approach, three different feature vectors are computed by using global information, local information and CQMD information extraction strategies. In all the three cases, a CG based circle is drawn over the character sample and different feature values are estimated from therein. Details of these feature computation processes are described in the following sub-sections.

4.2.1.3.1 Global Information

In this feature extraction approach firstly, the CG of each character sample is computed. Thereafter the distance of farthest data pixel of the character sample from the CG is determined. This distance is then considered as radius of a circle that completely encloses the entire character sample. After forming the circle, the ratios of the radius of the circle and the distance between the CG of the character sample to the data point p_k are computed, where k=1,2,....,64 (as in this procedure all character images are normalized into 64 points). Estimated 64 ratios are considered as features for the recognition of basic Bangla characters. Close observation reveals that these feature values have the power to explain the shape information of a character meticulously. Fig. 4.2.3 (a-b) shows the formation of the CG-based circle and the computation of the one such ratio. Here G is the CG, K is the farthest data pixel from CG, R is the radius and p_1 is the arbitrary pixel point of that character. The ratio of distances p_1G and R is calculated. The detailed steps are mentioned in Algorithm 8.

Algorithm 8: Algorithm to compute global information from the CG-based circle

```
STEP 1: BEGIN
STEP 2: Calculate the CG of the character sample.
STEP 3: Find the distance R from CG to farthest distant pixel from the CG.
STEP 4: Consider this distance R as radius and form the circle.
STEP 5: For each pixel point, calculate the ratio of the distance from the pixel under consideration to CG and radius.
STEP 6: END
```

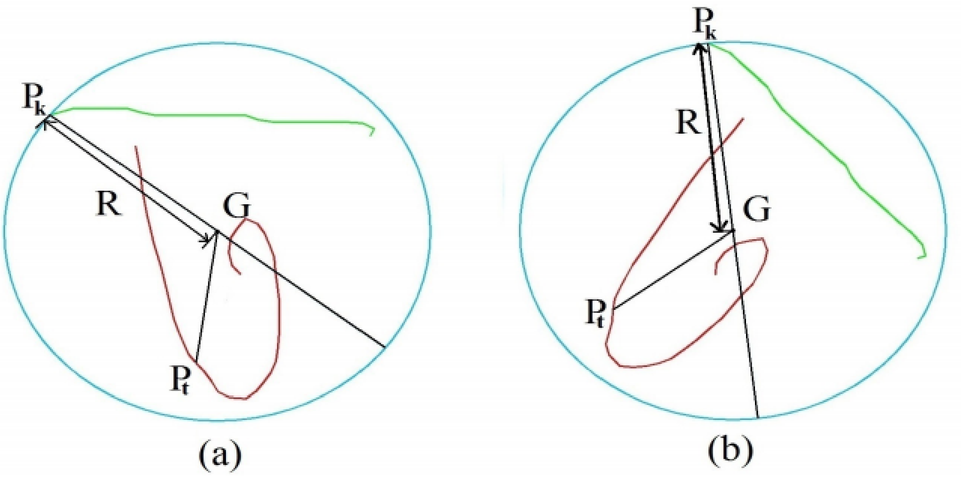


Figure 4.2.3: CG-based circle generation enclosing the character ত for the estimating global feature at (a) 0° and (b) 45° rotations

The character is also rotated by 90° clockwise direction (see Fig. 4.2.3 (b)) and same procedure is repeated. As a result 128 (i.e. 2x64) features are obtained for each character sample.

4.2.1.3.2 Local Information

This feature extraction technique starts with the CG-based circle formation as described in the previous section. Then the circular region is divided into four subregions based on CG. Fig. 4.2.4(a) and 4.2.4(b) describe the situations for 0° and 90° clockwise rotational effect. This method basically keeps information about pixel distribution of the character sample in each sub-region. In Fig. 4.2.4(a) and 4.2.4(b), yellow, red, green and blue coloured portions describe the mass distribution of the character sample in each sub region. In each region, the ratio of the number of coloured pixels to the background pixels are calculated. These ratios are considered as feature values for character recognition purpose. Algorithm 9 highlights the steps needed to compute local features.

Algorithm 9: Algorithm to compute the CG-based local feature

```
STEP 1: BEGIN
STEP 2: Calculate CG of a character sample.
STEP 3: Find the distance R from CG to the farthest data pixel point from
the CG.
STEP 4: Consider this distance R as radius and form a circle.
STEP 5: Circular region is divided into four sub-regions based on CG.
STEP 6: For each sub region, find the ratio of the number of data pixels to
background pixels.
STEP 7: END
```

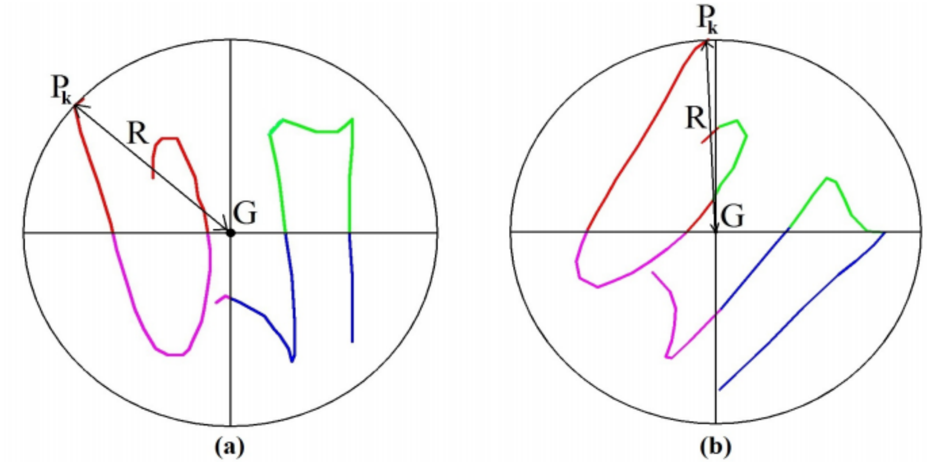


Figure 4.2.4: Segmentation of an image into four sub-regions to estimate local features at (a) 0° and (b) 45°(clockwise rotation)

From this procedure, four feature values are computed considering four sub-regions. The computed values describe about the mass distribution in the said four sub-regions. The character sample is rotated by 45° in clockwise direction and Algorithm 9 is applied in similar fashion. Hence, a total of 8 (i.e. 2 * 4) feature values are measured as local information.

4.2.1.3.3 Circular Quadrant Mass Distribution (CQMD)

In this procedure also, a CG based circle is drawn over a character sample and then the sample is divided into four sub-regions (as mentioned in local information extraction technique). Let R is the radius of the circle. Two more concentric circles are drawn with radii R/3 and 2*R/3 respectively, as shown in Fig. 4.2.5. As an effect, each sub-region is again divided into three small sub-parts marked as A_i, B_i and C_i, where i=1,2,3,4 as mentioned in Fig. 4.2.5. In each sub-part of each sub-region, the ratio of foreground to background pixels is measured and the same is served as feature. Although the basic idea of CQMD is the same as that of local information feature extraction approach, but this technique can estimate shape detail more minutely. As the writing pattern of different Bangla characters are different and cursive in nature, the pixel distributions inside the sub-parts- A_i, B_i and C_i vary significantly. This difference enhances the ability to distinguish different character patterns effectively and thereby augments the efficiency of the recognition process. In this approach, for each sub-region, three ratios (describing number of data pixels to back ground pixels) are produced (as there are three sub-parts in each sub-region). Hence, a total of 12(=4 * 3) feature values are computed by considering all four sub-regions of the corresponding character sample. The character sample is then rotated by 45° in a clockwise direction and the same steps are repeated. Finally, the CQMD approach produces 24(=2*12) element feature vector from each character sample.

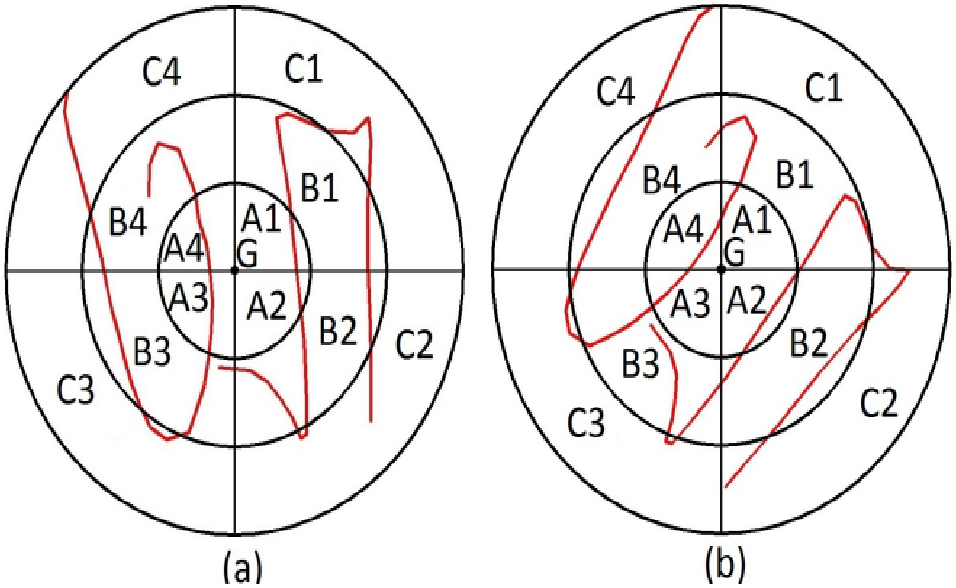


Figure 4.2.5: Illustration of CQMD feature extraction technique when the character আ is rotated at (a) 0°and (b) 45°clockwise directions

4.2.1.4 Area and Local features[98]

In this section, quad-tree based image segmentation approach is followed to design three different feature extraction techniques for the recognition of online handwritten Bangla characters. At the first step, the sample character is divided into four (2x2) rectangular blocks. Then one area based feature using composite Simpson's rule and two other local features namely mass distribution and chord length are computed from therein. In each iteration, the level of quad-tree is increased by one, which can be achieved by dividing each rectangular block into four sub-blocks to get a structure of 4x4 blocks at depth two. The said features are again estimated from each block to witness the outcome with a closer view than before. In the current experiment, the observations are recorded by segmenting the image up to 64 blocks (i.e. quad-tree of depth four). Fig.4.2.6 reflects quad-tree based segmentation of the same character sample varying the depth from one to four. In this way 4, 16, 64 and 256 feature values are produced at quad-tree of depth one, two, three and four respectively. The details of feature extraction methodologies are described in following subsections:

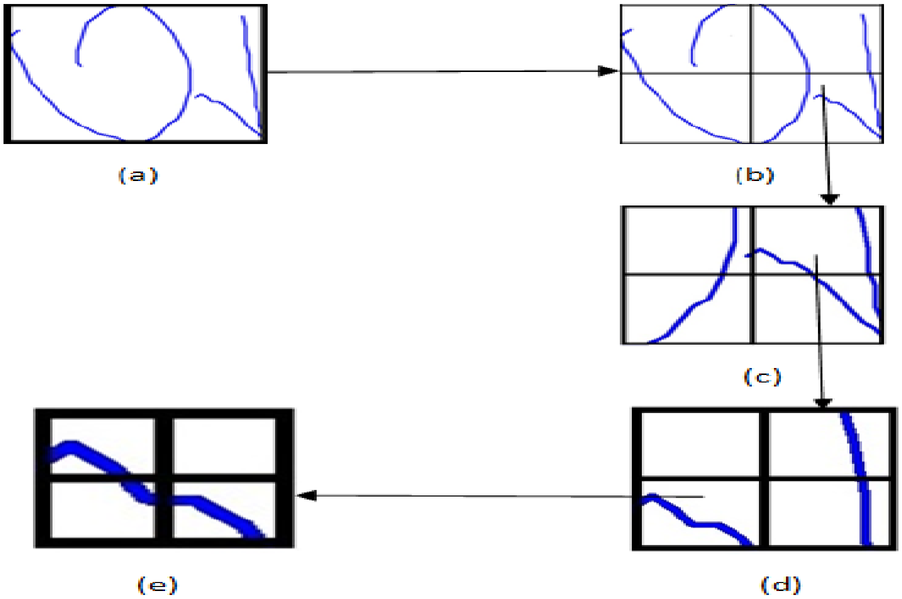


Figure 4.2.6: Illustration of image segmentation using quad-tree based approach (a) Original image. (b-e) segmented images produced at different depths of the tree

4.2.1.4.1 Area feature

As the structural patterns of the alphabet in Bangla are supposedly unalike from each other, hence, it can be assumed that for different types of characters, pixel patterns constituting the samples must be dissimilar at different blocks. There may be some blocks where no pixels are found for any particular character sample. Therefore, when block-wise area under the curve is calculated then these calculations become distinct for different character patterns. The truthfulness of this statement can be easily observed from Fig. 4.2.7 by looking at the positional information of selected blocks for character samples ই, ক. Thereby, it is inferred that these values could be useful for classification of the characters. The working principle for the computation of area feature is presented in Algorithm 10. Eq. 4.3 is known as composite Simpson's rule which is used here to find area under a curve constituting a set of points.

Algorithm 10: Area feature using composite Simpson's rule

```
STEP 1: BEGIN
STEP 2: for i = 1 to N do,
STEP 3: for j = 1 to N do,
STEP 4: Find area covered by the curve in $block_{i,j}$ using Eq.4.3
STEP 5: End for
STEP 6: End for
STEP 7: End
```

$$Area = \sum_{k=1}^{n-1} ((h/3) * (y_k + 4 * y_{middle} + y_{k+1})) \tag{4.3}$$

Where, each rectangular block contains varying number of pixels n, starting from y_1 to y_n. (y_k, y_{k+1}) denotes the measurement of y coordinate values between consecutive pixel points in that block. Values of h and y_{middle} have been calculated as follows:

$$Area = \sum_{k=1}^{n-1} ((h/3) * (y_k + 4 * y_{middle} + y_{k+1}))$$

$$y_{middle} = \frac{(y_k + y_{k+1})}{2}$$

After execution of Algorithm 10, a total of N*N number of area values will be generated as an outcome when the character image is divided into NxN rectangular blocks. These measurements are taken as feature values. This has been assumed that the blocks which do not have any pixel return zero as area value.

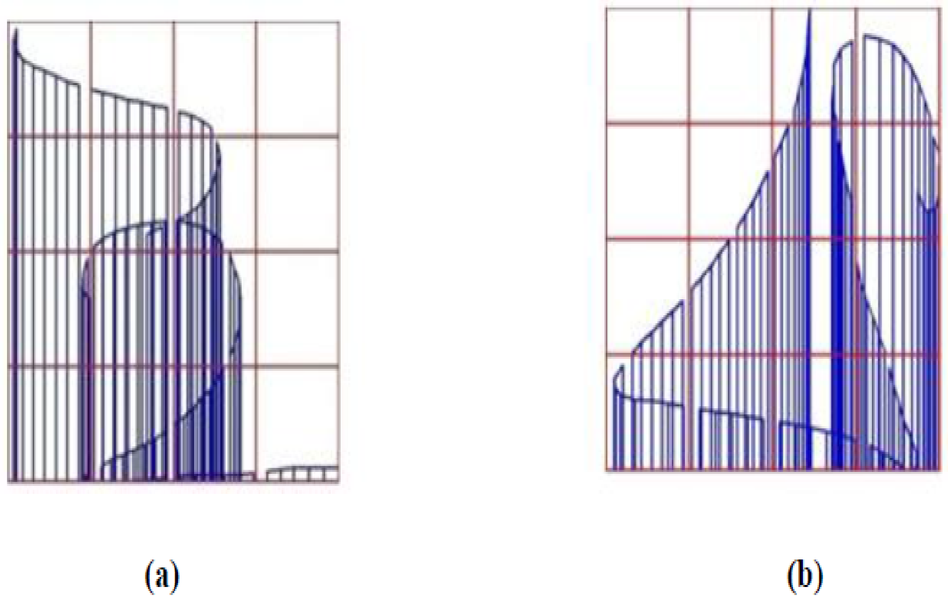


Figure 4.2.7: Calculation of area under curve in the respective blocks for two different character samples using composite Simpson's rule in a quad-tree based image segmentation at depth two

4.2.1.4.2 Local Feature

- Mass Distribution

Depending on structural pattern of the character, certain rectangular blocks are densely populated by the data pixels. Therefore, block-wise mass distribution information may carry an important role to distinguish different character patterns efficiently. Here mass distribution describes the pixel counts inside a block, produced by quad-tree image segmentation approach. Fig.4.2.8 shows mass distribution of the character samples ই, ক when the images are segmented into 16 blocks. Here, blue points are the pixels in the respective blocks. From these figures this can be easily understood that for different character patterns, a particular block has varied data pixels, which in turn produces discriminative feature towards online Bangla handwritten character recognition.

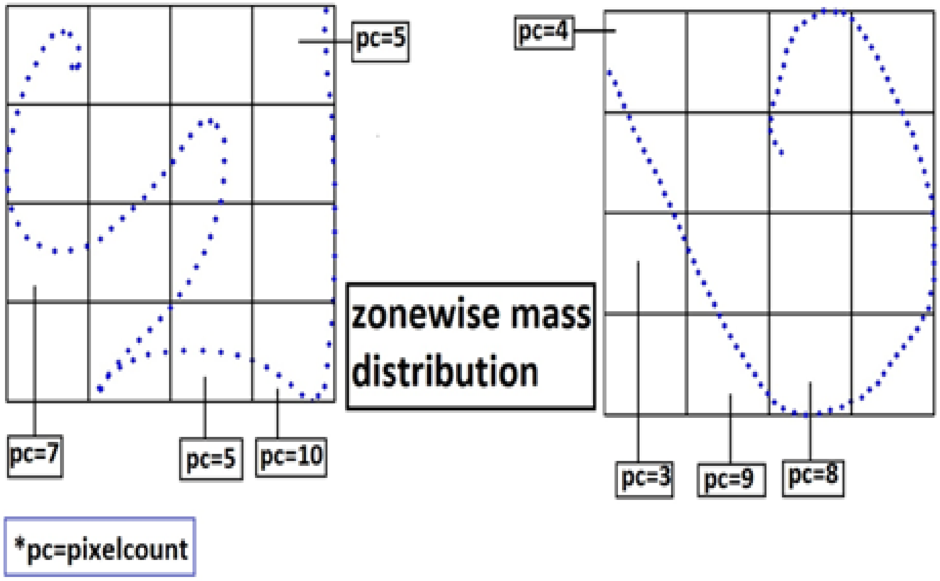


Figure 4.2.8: Mass distribution of two character samples খ and ত in a quad-tree based image segmentation at depth two

- Chord Length

  As compared to mass distribution, in this approach, the length of the contributed chord in each block has been considered. Dividing the character sample into a number of small chords/segments and storing block-wise chord length information as feature values play a vital role in this pattern classification problem. This is because these lengths vary significantly for different character patterns. Algorithm 11 describes the steps to compute block-wise chord length feature.

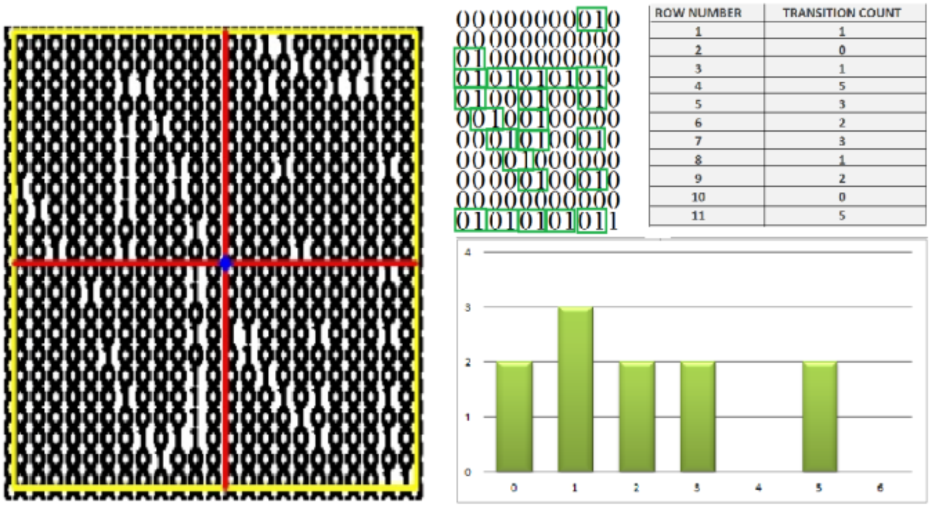
Algorithm 11: Block-wise chord length feature extraction procedure

```
STEP 1: BEGIN
STEP 2: for i = 1 to N do,
STEP 3: for j = 1 to N do,
STEP 4: $Chord\_Length_{i,j} = \sum_{n=1}^{k-1} \sqrt{(x_n - x_{n+1})^2 - (y_n - y_{n+1})^2}$
STEP 5: End for
STEP 6: End for
STEP 7: End
```

  Let us assume that m number of varying pixels starting from (x_1,y_1) to (x_m,y_m) is there in each rectangular block as mentioned in Fig. 4.2.8. To find the chord length of $block_{i,j}$, summation of Euclidian distances between the consecutive pair of pixels have been measured in the said block as indicated in STEP 4 of Algorithm 11. Fig. 4.2.8 clearly demonstrates the fact that length of the chord remarkably changes for different characters in a particular block which enables the classifier to differentiate handwritten online Bangla characters successfully.

4.2.1.5 Transition Count Feature

This approach scales a character to a window of size 64 X 64 and binarized. A pixel is set to '1' if it is a data point (foreground pixel), and '0' otherwise (background pixel). Based on CG, the character image is then divided into four subregions. In each of these sub-regions, the transition counts from background to foreground pixel and vice verse are computed along four directions: row, column and along two major diagonals (i.e. at four different angles $\theta = 0^\circ, 90^\circ, 45^\circ, 135^\circ$). The physical interpretation of traversing the four directions is to estimate the data distribution along different directions. It is to be noted that the size of these four sub-regions depends on the position of the CG. Fig. 4.2.9 (a) illustrates one such scenario. It has been observed that maximum value of the transition count for a particular row/column/diagonal within a sub-image of a character varies from 0 to 6. Therefore, to get an idea about how different values of transition count vary over the entire image, Then, the frequencies of these counts are measured and use them as feature values. For each sub-image 28 (i.e. 7x4) feature values are generated for '0' to '1' transition and another 28 for '1' to '0'. Thereby, total 56 (i.e. 28x2) feature values are generated for each sub-image. Finally, a total of 224 (i.e. 56x4) feature values are produced for the whole image. For easy reference, feature calculation (here, '0' to '1' transition) on a hypothetical image is described in Fig. 4.2.9(b).



```
00000000010
00000000000
01000000000
01010101010
01000100010
00100100000
00010100010
00001000000
00000100010
00000000000
01010101011
```

ROW NUMBER	TRANSITION COUNT
1	1
2	0
3	1
4	5
5	3
6	2
7	3
8	1
9	2
10	0
11	5

(a) (b)

Figure 4.2.9: Transition count feature calculation for character আ

4.2.1.6 Topological Feature

- Crossing point

  There are a few character pairs in the Bangla alphabet which exhibit strong similarities. For example, 'ম' and 'য' are quite similar in structure except 'ম' has a crossing point in the lower left side due to formation of a loop. Hence, the existence of crossing point acts as a good feature to differentiate such character pairs. To compute this feature, a minor modification has been made with the sample character. All the missing pixels between two consecutive data points are estimated. This feature monitors whether a character sample contains a loop or not. For example, consider Fig. 4.2.10 where the existence of loop can easily be observed. In this figure, the crossing point is marked green. It is clear that only those pixel points of the sample character are crossing points that have more than three adjacent neighbours and thus help to form a loop. This situation is clearly depicted in Fig. 4.2.10 where blue points are the neighbours of corresponding crossing point. Total of five feature values (the number of crossing points, the coordinates of the first crossing point, and the coordinates of closest neighbour of the first crossing point) are computed using this procedure. This technique becomes effective to distinguish the shapes having loops. Again, as this method yields only 5 features, hence, this features are combined with CG based circle features. when all three CG based circle approaches are combined, then the size of the feature vector becomes 160 (i.e. 128 + 8 + 24) element feature vector. For topological features, another five feature values are produced. Therefore, a total of 165 (160+5) features are generated from each character sample.

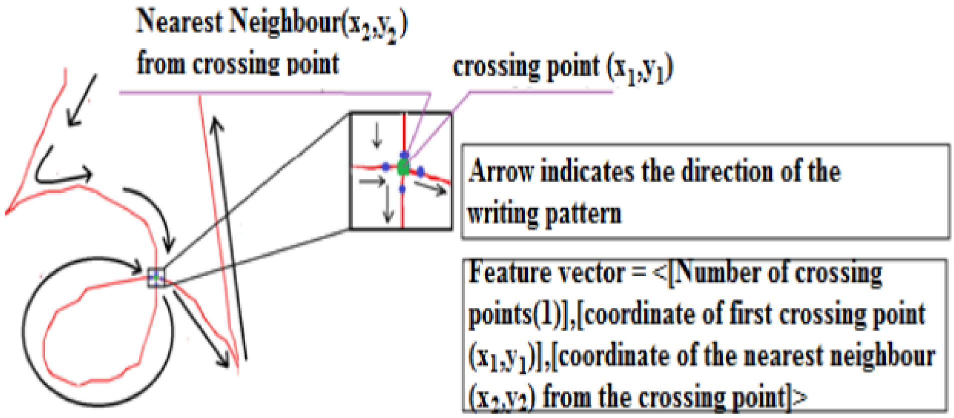

Nearest Neighbour(x2,y2)
from crossing point
crossing point (x1,y1)
Arrow indicates the direction of the writing pattern
Feature vector = <[Number of crossing points(1)],[coordinate of first crossing point (x1,y1)],[coordinate of the nearest neighbour (x2,y2) from the crossing point]>

Figure 4.2.10: Identification of a crossing point in character sample 'ম'

4.2.2 Feature extraction methods for stroke recognition

Strokes cannot be collected separately but they are extracted from collected character or text information. Strokes at character level can be stored easily but the same is not true for text level. People generally write characters in single or by using multiple strokes and these are treated as basic strokes. After analyzing all the strokes collected from the character database, it has been found that there are 52 different stroke symbols as shown in Fig. 5.1.5 (in Chapter 5). In case of text, it is found that there exist joining between basic strokes. Hence, such joint strokes need to be segmented for any stroke-based word recognition system and section 6.1.1 (in Chapter 6) describes the word segmentation algorithm in detailed for the same. Analyzing the collected strokes it has been found that there are 59 different stroke symbols as reflected in Fig. 6.1.11 (in Chapter 6). After extracting the strokes from character/text, following feature extraction techniques are experimented for stroke recognition purpose.

4.2.2.1 Zone-based Path Traversal (ZPT) Feature

In this procedure, a stroke image is divided into 8x8 rectangular blocks. According to the writing pattern of the stroke, these blocks may have varied number of pixel points. For the extraction of ZPT feature, each block is marked with a distinct index value as shown in Fig. 4.2.11. Then the pixels are traversed according to the writing pattern to find out the longest segments present within different blocks (e.g. AB, BC, CD etc.). Please note that index of a block is set to track the positional information of the segment and a segment represents the continuous portion/contribution of a stroke sample in a block. From the figure, it can be noticed that a particular block can have more than one such segments (due to inherent shape characteristic of a stroke). The contributions of all the segments for each block are measured by considering the facts: if any segment does not contain end point of a stroke and the pattern passes through say, block 1 to block 2, then, the contribution of the segment lies in block 1 is set to ((n - 1) * index_value(block1)) + index_value(block2). Otherwise if this is the last segment within block 1, then the contribution is set as (n *index_value(block1)). Here n denotes number of pixels within that segment. It is assumed that a pixel p_i contributes the index value of the block it resides iff the next pixel p_j lies within the same block. Otherwise, if the next pixel p_j lies in different block (say 2), then pi contributes the index value of the block 2. Zero valued block implies that it does not contain any pixel. As the index value of blocks are known, hence, contributed value of any arbitrary block not only reveals the pixel density of that block along with the pixels that contribute in the neighbouring blocks but also this feature extraction technique helps to identify the path (sequence of blocks) through which stroke pattern passes. The transition between blocks is depicted in Fig. 4.2.11 (pixel colour depends on the index value of corresponding blocks). For a writing pattern, block-wise contributions of all the segments served as feature values in this method. As a stroke is divided into 8x8 number of blocks, this procedure yields a 64-element feature vector. It is to be noted that this procedure tries to acquire the shape information of the stroke.

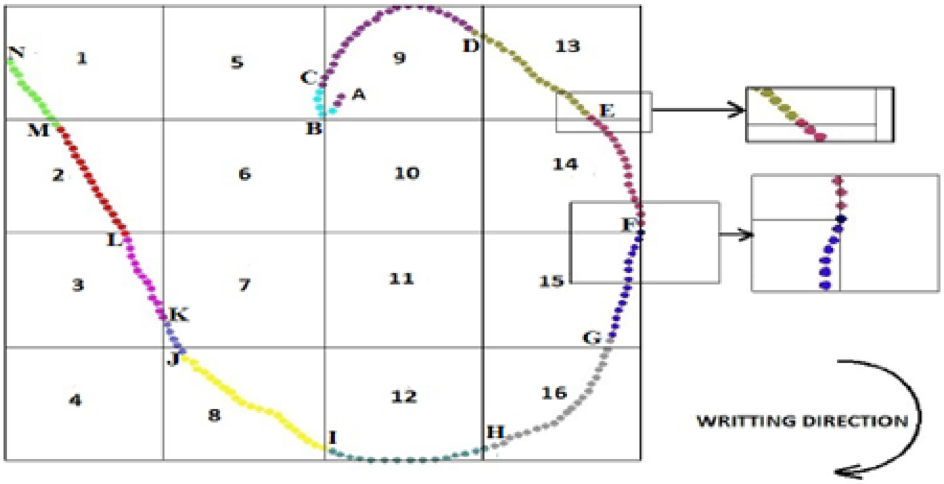


Figure 4.2.11: Pictorial description of ZPT feature computation

4.2.2.2 Distance based feature

Distance based features are useful for both stroke and character recognition as mentioned in [99, 101, 104]. When this is used for stroke recognition, at first, each stroke is segmented to certain number of parts (say, M). As each part is represented by a pair of points, the same is divided into M+1 almost equidistant points p_k (k ranges from 1 to M+1). Feature values are estimated by the distances measured from each point to every other points as shown in Fig. 4.2.12. Existence of some nearly similar shape structures has been observed during statistical analysis of the stroke database considered here. In order to differentiate such similar shape patterns, some local information are estimated to produce discriminating features for the recognition purpose. As selecting the optimal number of segmentation of the stroke is always challenging, to avoid the complexity, we have used the work [99, 101, 104] where authors have divided each stroke into 16 segments. or details of Distance based feature extraction procedure see [101]. For a stroke pattern with M number of segments, this feature extraction method produces M*(M+1)/2 number of features (i.e. the distance values). Thereby a feature vector of size 136 is generated because here strokes are segmented into 16 parts.

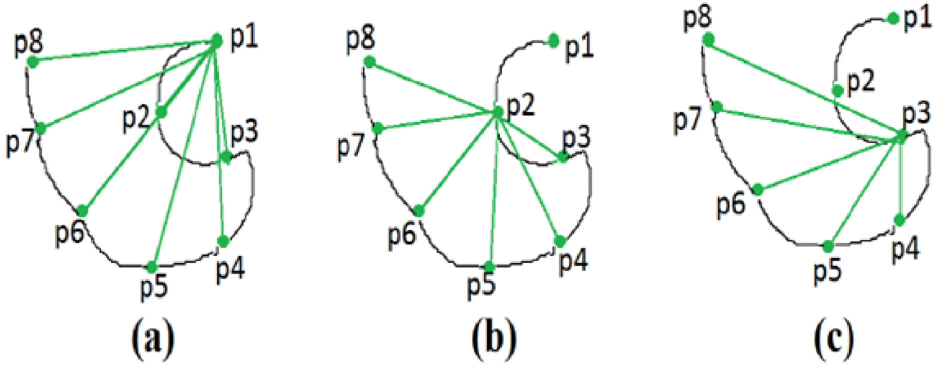


Figure 4.2.12: Calculation of distances between segmented points

4.2.2.3 Point based feature

In the first step of point based feature calculation, a stroke sample is normalized into 64 points. The steps for point based feature extraction technique are described in Algorithm 12 [81, 82]. The dimension of the produced feature vector is 64x3=192.

Algorithm 12: Point based feature computation

```
STEP 1: For each pixel i
STEP 2: Calculate the normalized horizontal co-ordinates
STEP 3: $t_{i1} = (x_i - \mu_x)/\sigma_y$
STEP 4: Calculate the normalized vertical co-ordinates
STEP 5: $t_{i2} = (y_i - \mu_y)/\sigma_y$
STEP 6:$\mu_i = (\frac{1}{N} \sum_{i=1}^{N} p_i)$
STEP 7: $\sigma_y = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} (\mu_y - y_i)^2}$
STEP 8: Calculate the slope angle ($t_{i2}$)
STEP 9: $t_{i2} = arg((x_{i+1} - x_{i-1}) + j * (x_{i+1} - x_{i-1}))$
where $j^2$ = -1 and ”arg ” is an approximation of the tangent slope angle at point i
STEP 10: End
```

4.2.2.4 Curvature based feature

This procedure follows the quad-tree based image segmentation for extracting some local features. It is to be noted that the variation of writing patterns in different zones in terms of shape and structure plays an important role in distinguishing the different character symbols. Here, if the writing pattern in a particular zone is considered as a curve, then the nature of the curve may be informative for classifying the strokes. To calculate this feature, firstly, the position of the first pixel of the stroke in the zone is examined and then sequentially scanned the points till it lies within the same zone. The set of pixel points together is considered as a segment. The first and last points of such a segment are shown in Fig. 4.2.13 (marked in red and green respectively). The very next pixel is considered as the first point for the next zone. In this way, all the segments are computed for the stroke sample. A straight line L is drawn joining those end points. Then the perpendicular distances from each of the pixel points in a zone to the straight line L are calculated using Eq. 6.6. The sum of the perpendicular distances is taken as the measurement of total amount of curviness of the stroke segment in that zone. If any zone contains no data pixel then by default this value is set to zero. The algorithm for the extraction of the curvature feature is given in Algorithm 13.

Algorithm 13: Curvature based feature computation

```
STEP 1: Perform a quad-tree based image segmentation over the stroke sample.
STEP 2: For each zone of the image, draw a straight line L from the starting pixel to the end pixel of that zone.
STEP 3: Calculate the sum of the perpendicular distances using equation 6.6 from the pixel $p_j$ to L, where j ranges from 1 to N. N represents the total number of pixels within that zone.
STEP 4: Repeat Steps 2-3 for all the zones.
```

$$Perpendiculardistance, D = \frac{|(y_2 - y_1)x_0 - (x_2 - x_1)y_0 - x_2y_1 - y_2x_1|}{\sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2}} \tag{4.4}$$

Where L passes through two points, say P_1=(x_1, y_1) and P_2=(x_2, y_2). The distance measurement from (x_0, y_0) to L is D. For a particular stroke sample, this procedure generates 64 features from 64 zones.

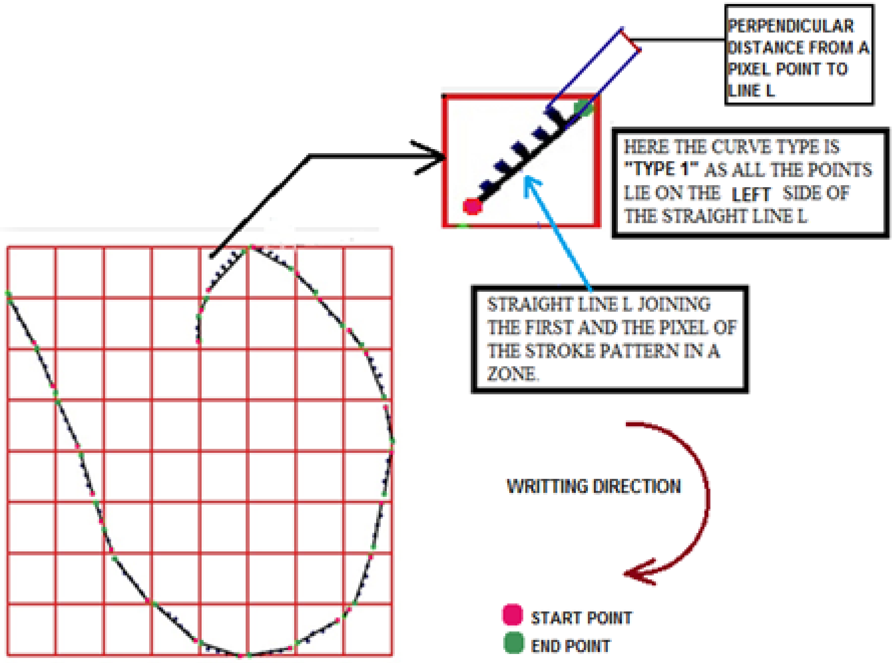

PERPENDICULAR DISTANCE FROM A PIXEL POINT TO LINE L
HERE THE CURVE TYPE IS "TYPE 1" AS ALL THE POINTS LIE ON THE LEFT SIDE OF THE STRAIGHT LINE L
STRAIGHT LINE L JOINING THE FIRST AND THE PIXEL OF THE STROKE PATTERN IN A ZONE.
WRITTING DIRECTION
START POINT
END POINT

Figure 4.2.13: Illustration of curvature based feature extraction technique applied on a stroke sample 3___ when segmented at quad-tree of depth 3

4.2.3 Deep learning based approach

The experiment mentioned in [106], firstly, the offline image is generated from the online information and then resize it to fit into a window of size 28x28. CNN, a deep learning architecture uses convolutional layers that involves arbitrary number of learnable filters to move along the width and height of the image to produce feature map. A filter can be considered as an array of numbers where the numbers are called weights or parameters. Convolution operation with kernel of spatial dimension 5 (i.e. size of 5x5) converts 28 spatial dimension to 24 (i.e., 28-5+1) spatial dimension [79]. After sliding the filter over all the locations, a 24x24 array of numbers are achieved, which is known as activation map or feature map. Hence, size of the first level feature maps becomes 24x24.

4.3 Classification schemes

After feature set creation, the next task is classification of characters or strokes. Classification analyzes the numerical properties (extracted as feature values) of various stroke/character and organizing them into different categories. All the classification algorithms typically consider two phases of processing: training and testing/validation. Training phase is typically assumed as "gold standard" data, where the model is trained by pairing our input with the expected output. During the test phase, how well the model is trained is estimated. The parameters like: overall accuracy rate, classification error, model building time etc have been used for performance measurement.

This study includes state-of-the-art of classifiers. These classifiers are mainly categorized into four groups: Bayesian, Functional, Rule based and Tree based. BayesNet is one of the popular Bayesian classifier which is used under Bayesian category. Under Functional classifier we have used four popular classifiers namely: MLP, SVM, Simple Logistic, BayesNet and NaiveBayes. Among the classifiers mentioned above, MLP is widely used in our experiment. The following section briefly describes about the above mentioned classifiers.

4.3.1 MLP

According to Rumelhart et al. [83], neural network is a massively parallel distributed system consisting of a large number of interconnected simple processors (neurons) which has the ability for storing experimental knowledge and making it available for future use. The local information processing in the brain takes place in cells or neurons which form a large number of parallel networks in the cortex of the brain. The acquired knowledge or memory is known to be stored in the form of connection strengths between neurons. Some key features of neural networks are:

- Each neuron has a number of inputs and a single output. Neuron processes information locally by performing a simple mathematical function, also called activation function, on its outputs.
- Neurons operate in parallel and are connected into a network through weights (also called synaptic weights) indicating the connection strength.
- Network acquires knowledge from the training data in a process called learning and acquired knowledge is stored in the form of synaptic weights.

Artificial neural networks (ANN) are a programming paradigm that seek to emulate the microstructure of the brain, and are used extensively in artificial intelligence problems from simple pattern-recognition tasks, to advanced symbolic manipulation. Ideas for ANNs have evolved from neurobiology and ANNs are basically different models that attempt to simulate some of the basic information processing found in the brain. The MLP is the most well-known ANN model used for the non-linear prediction and classification tasks. According to connection topology, an MLP is a feed-forward neural network with at least one hidden layer. Fig 4.3.1 shows the structure of an MLP network. Topology of a neural network refers to its framework as well as its interconnection scheme. The framework is often specified by the number of layers and the number of nodes per payer. The types of layer include:

Input layer: In input layer, dummy neurons transmit the input data to the hidden neurons through the input–hidden layer links. The inputs are multiplied by the weights on the corresponding links before they reach the inputs of the hidden layer neurons.

Hidden layer: Neurons in the hidden layer are not directly observable and hence called hidden. Neurons in the hidden layer accumulate those weighted inputs received by them and process those accumulated values using the activation function.

Output layer: The neurons in it are called output units, which encode possible concepts (or values) to assign to the instance under consideration. For example, each output unit represents a class of objects.

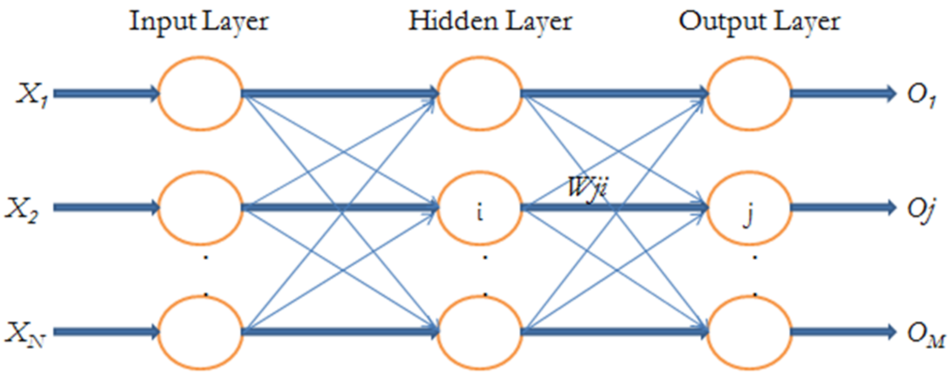


Figure 4.3.1: Architecture of MLP

Learning or training in MLP is supervised i.e., the network utilizes the information on the class membership of each training instance. Learning/training involves presenting the input vectors, one at a time, at the input layer and allowing the input passing to hidden layer and then to output layer producing a final output. At this point, the network output is compared with the desired output and their difference, also called error, is computed. This error is back propagated and is utilized for adjusting the weights of the input hidden-layer and hidden-output layer links using an appropriate learning procedure to minimize the error in a repeated processing of the inputs. This algorithm is known as Error Back Propagation Algorithm or simply Back Propagation (BP) algorithm [83] which is given in 14:

This weight adjustment may be done after presentation of each input vector or a batch of all input vectors (epoch). In batch learning mode, error gradient is

Algorithm 14: Back Propagation algorithm

```
STEP 1: Set all synaptic weights all links with random values.

STEP 2: Let the neuron j receives input values which are the outputs of the
  neurons in the previous layer then the accumulated input for the neuron j
  is computed as $I_j = \sum_{W_{ij}} O_i$ where,$O_i$ is the output of a neuron i in the
  previous layer and $W_{ij}$ is the synaptic weight of the connection between
  the neurons i and j.

STEP 3: Output of the neuron j is given as $O_j = F(I_j)$, where F(.) is the
  activation function.

STEP 4: On presenting an input vector, the corresponding error at the
  output layer for any neuron p is given as
  $E = E_p - D_p$, where $O_p$ is the output of the pth neuron in the output
  layer and $D_p$ is the corresponding desired output.

STEP 5: Adjustment of the synaptic weights starts at the output unit and
  works backward to the hidden layers using the following formula:
  $W_{ij}(t+1) = W_{ij}(t) + \Delta W_{ij}$, where $W_{ij}(t)$ is the synaptic weight of the
  connection between of a $neuron_i$ to a $neuron_j$ at the $t^{th}$ iteration and $\Delta W_{ij}$
  is the weight adjustment which is computed as
  $\Delta W_{ij} = \eta \delta_j O_i$, where $\eta$ is the learning rate (0< $\eta$ <1) and $\delta_j$ is the error
  gradient at $neuron_j$. Sometimes, a momentum term is added to weight
  adjustment formula for conservation of the past trends in weight
  adjustment as given below:
  $W_{ij}(t+1) = W_{ij}(t) \eta \delta_j O_i + \alpha(W_{ij}(t) - W_{ij}(t-1))$,0< $\alpha$ <1. The error
  gradient $\delta_j$ for different layers is defined below: For neurons in the output
  layer
  $\delta_j = O_j(1 - O_j)(D_j - O_j)$. For neurons in hidden layer
  $\delta_j = O_j(1 - O_j) \sum_k \delta_k W_{jk}$ where $\delta_k$ is the error gradient at $neuron_k$
  connection with neuron j in the hidden layer.

STEP 6: Repeat STEP 4 and 5 until convergence is reached.
```

summed over all the input training vectors to produce a weight change.

Activation function plays a key role to design an MLP. Generally, a non-linear, continuous and bounded function is chosen as an activation function. The sigmoid functions are a family of S-shaped functions satisfies the above said characteristics. A popular form of sigmoid function shown in Eq. 6.7, which has a lower bound 0 and an upper bound 1, and commonly used in MLP is given as:

$$F(x) = \frac{1}{1 + e^{-x}} \tag{4.5}$$

MLP differs from the linear classifiers like SVM in a way that it can form complex decision space rather than forming a hyperplane. Each neuron in the first hidden layer (just above the input layer) creates a hyperplane. Each neuron in the next layer combines those hyperplanes to form a convex region. Combination of these convex regions results into concave regions. Thus, it is possible to form any arbitrary regions with appropriate number of neurons in each of them.

For an M class classification problem, the number of output neurons will be M. If the input vector has dimension of N, then number of input neurons will be N. During learning/training process number of hidden layers and number of neurons in each layer are to be determined.

4.3.2 SVM

SVM [29], a well-known pattern dichotomizer and linear regression tool, has popularly been used for different applications such as character recognition, object detection, content-based image retrieval, face detection, text classification, and medical applications, bioinformatics just to name a few.

As a pattern dichotomizer, SVM tries to construct a hyperplane as a decision surface such that the margin of separation between the positive and negative samples is maximized. It is equivalent to perform SRM, rooted in Statistical Learning theory, to achieve good generalization.

Unlike MLP, SVM relies on pre-processing the input vector, representing the input pattern in higher dimension. With an appropriate nonlinear mapping function $\varphi(.)$ mapping the input vector to higher dimension, it is possible to find a separating hyperplane between the two classes which may not be possible in the original dimension of the input vector.

The goal of SVM training is to find a hyperplane with largest margin. Here margin is defined as the distance between the hyperplane and the nearest training patterns (also called the support vectors) in the higher dimensional space. Thus the support vectors are the training patterns which define the hyperplane and they are the most informative patterns. Fig 4.3.2 shows an example of separating hyperplane.

In general, SVMs are trained to perform binary classification by solving the following optimization problem. For a given training data set $\mathbb{T}D$ consisting of pairs x_i,y_i, where x_i denotes input feature vector for i^{th} sample, $x_i \varepsilon \mathbb{R}n$ and y_i denotes the corresponding target value $y_i \varepsilon 1, 1$, SVM tries to solve the following problem

$$min_{w,b,\zeta} \frac{1}{2} w.w + C \sum_{i=1}^{N} \zeta_i \tag{4.6}$$

subject to constraints $y_i(w.x_i + b) \geq 1 - \zeta_i$ for $(x_i, y_i) = 1, 2, ...N$

C is a user specified positive parameter and $\zeta_i \geq 0$ for all i. w is a weight vector and b is the bias.

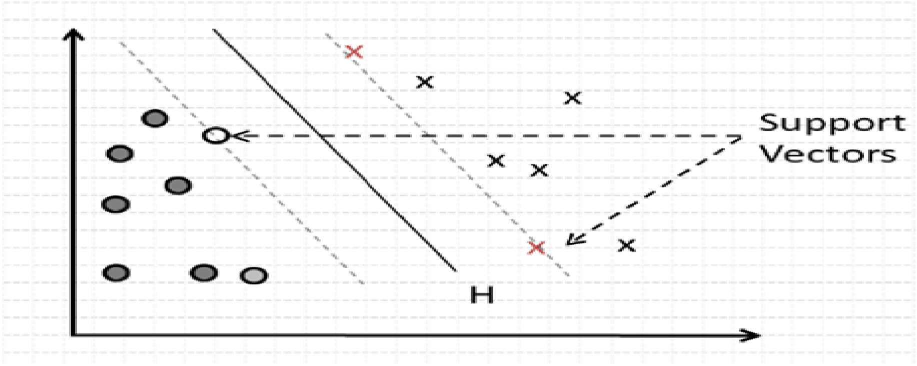


Figure 4.3.2: An illustration of separating hyperplane (H)

In case of linearly inseparable classes, a kernel function $K(x, x_i)$ is used to map the input feature vector x_i into higher dimensional feature space to make them linearly separable. Thus the solution determines the SVM classifier.

$$f(x) = sign(\sum_{i=1}^{N} y_i a_i K(x, x_i) + b) \tag{4.7}$$

where, $sign(u) = \begin{cases} 1 & foru > 0 \\ 0 & u < 0 \end{cases}$

Here, b is the bias, a_i is the Langrage multiplier and $K(x, x_i)$ is the kernel function.

There are several types of kernels used in SVM models. Some of the popularly used kernel functions are:

- Gaussian (Radial Basis Function) kernel:

  $K(x, x_i) = exp(-\gamma * \|x - x_i\|^2)$

  where, $\gamma = \frac{1}{2\sigma^2}$ and ”σis the standard deviation of the x_i values.

- Polynomial kernel: $K(x, x_i) = (”\gamma x^T x_i + ”\alpha)^d$, where d is the degree of polynomial. d,γ>0 and a is a real number.

- Linear kernel: $K(x, x_i) = x^T x_i$

Out of these kernels, Gaussian kernel that results into realization of the RBF network is an important one. The rationale behind the RBF network specially targets non-linearly separable patterns, which may be the most likely scenario for practical applications. Studies have shown that RBF networks designed through support vector method can produce better recognition performances compared to those designed with traditional methodology for the same data set.

Though SVM is primarily designed for 2-class pattern classification, but multiclass pattern classification is also possible by combining several binary SVM classifiers. Among different regulations available for combining a number of binary

SVM classifiers "One Versus One" (OVO) and "One Versus All" (OVA) are widely used among pattern recognition community. To solve k class problem, $k(k-1)/2$ numbers of binary SVM classifiers is constructed in OVO methodology. On the other hand OVA method requires k number of binary SVM classifiers to solve the same problem. None of the methods is completely superior to each other.

4.3.3 SIMPLE LOGISTIC

It is a classifier for building linear logistic regression model [116]. Here LogitBoost is used with simple regression functions as base learner for fitting the logistic model. The optimal number of LogitBoost iterations to perform is cross-validated here, which helps for the selection of automatic attribute.

4.3.4 Bayesian network

Fig.4.3.3 depicts the possible structure of a Bayesian network used for classification. The dotted lines denote potential links, and the blue box is used to indicate that additional nodes and links can be added to the model, usually between the input and output nodes.

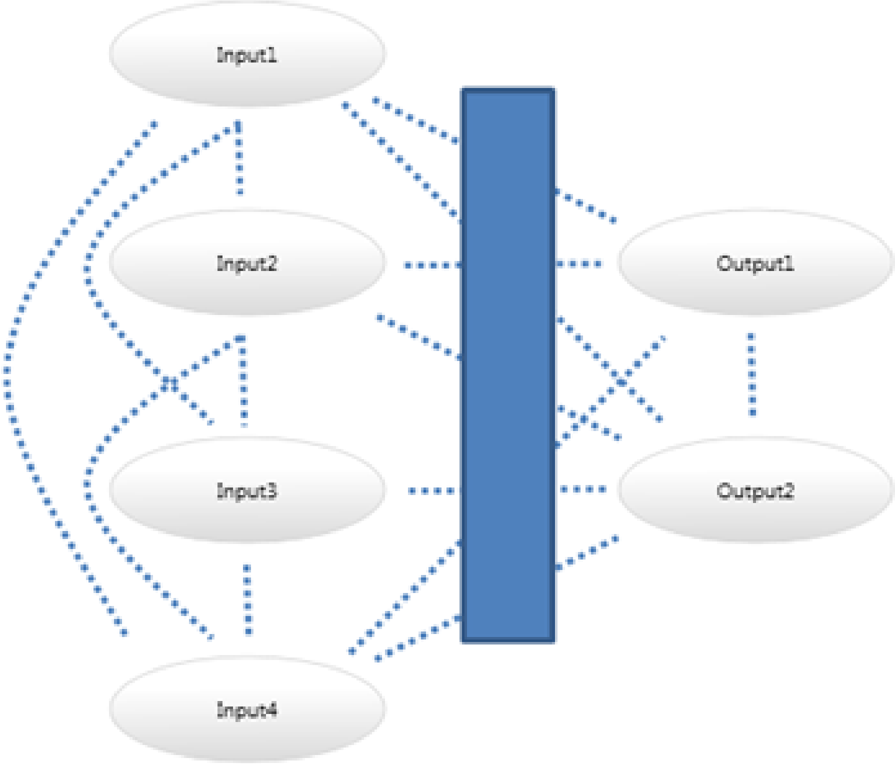


Figure 4.3.3: Generic structure of a Bayesian classifier

In order to perform classification with a Bayesian network such as the one depicted in Fig.4.3.3, first evidence must be set on the input nodes, and then the output nodes can be queried using standard Bayesian network inference. The result will be a distribution for each output node, so that one can not only determine the most probable state for each output, but also see the probability assigned to each output state.

5
Character Recognition

After extracting features at character and stroke levels, various experimentation have been performed in order to recognize the characters. As mentioned in Chapter 4, for character recognition purpose few features are extracted from the character holistically. These feature are then fed to the different classifiers for recognition. It is to be noted that, when strokes are considered in feature extraction phase, the recognition results of the same are passed through post-processing module in order to form the characters from the underlying recognized strokes. This chapter describes character recognition results by following holistic feature extraction procedures mentioned in [98–100, 102, 103]. This chapter also describes the rule based [101][99] and DFA based approaches [105] that are adopted for the formation of characters from the recognized strokes in stroke based character recognition procedure. In rule based approach, different stroke orders for all the characters are stored in a database and an unknown character is recognized by matching the stroke order sequence stored in the database. In DFA based approach, after classification, it recognizes the major stroke and then matches with the probable different paths for the construction of different characters from the major stroke. Following sections elaborately describes recognition results for both the procedures applied.

5.1 Experimental Results

Experimental results of holistic and stroke based character recognition approaches are described in detailed in the following sub-sections (4.2.1-4.2.2). The effectiveness of features used for holistic character recognition is mentioned in sub-section 4.2.1, whereas, 4.2.2 describes the results for stroke based character recognition approach.

5.1.1 Character recognition by holistic approach

This following sub-sections elaborately describes character recognition results by following holistic feature extraction procedures mentioned in [98–100, 102, 103].

5.1.1.1 Recognition by Distance based feature [99]

In the present work, 10, 000 character samples are used for the evaluation of Distance based feature. Some standard classifiers like MLP, Simple Logistic, BayesNet, NaiveBayes and SVM (as mentioned in the earlier section) are used for recognition of the characters. 5-fold cross-validation scheme has been applied on total dataset and Table 5.1.1 reflects the corresponding success rates of different classifiers. This has been noticed that if the process continue dividing the character sample by more number of segments, up to a certain level recognition rate increases gradually but after that it starts falling downwards. This is because, as the number of segments gets increased, length of the structural units of the character sample also gets decreased, indicating the detailed view of the character sample. Extracted features from such structural units then show better performance towards recognition of the character sample. It can be observed easily from Table 5.1.1 that, initially success rate is increasing steadily with increasing number of segments. Table 5.1.1 reveals the maximum recognition rate when the character sample is segmented into 40 segments for MLP and Simple Logistic classifiers whereas all the other three classifiers such as BayesNet, SVM and NaiveBayes show maximum success rates for 16 segments.

Again, when the character sample is segmented beyond these levels, then the size of those structural units becomes less informative, so extracted features from such small units fail to describe the character sample properly. As a result recognition rate starts to fall slowly. Gray cells in the 5.1.1 represent maximum success rates for a particular classifier. Fig. 5.1.1 graphically describes the behaviour of different classifiers for different number of segmentations. Black curve represents recognition result of MLP, whereas pink, green, violet and blue curves describe the recognition rates of Simple Logistic, SVM, NaiveBayes and BayesNet respectively. From this figure, it is observed that SVM recognizes the Bangla character samples with highest recognition accuracy of 98.20 % when $N = 16$. For the conducting the present experiment, we have chosen C-SVC type SVM from LibSVM library, whose kernel type is using radial basis function: $exp(-gamma * |u - v|^2) witheps = 0.001, gamma = 0.0$.

Table 5.1.1: Success rates of different classifiers for the recognition of online isolated basic Bangla characters

Number of segmentation (N)	Success rates for different Classifiers (in %)				
	MLP	Simple Logistic	BayesNet	SVM	NaiveBayes
6	91.05	94.03	90.36	97.79	87.71
8	93.68	96.46	93.16	98.01	89.69
10	95.16	97.28	94.40	98.10	91.46
16	95.79	97.57	95.50	98.20	92.75
32	96.36	97.79	93.53	95.42	92.45
40	96.77	97.80	93.28	90.64	91.68
48	96.67	97.79	93.16	83.65	91.28
52	96.58	97.69	92.88	79.70	91.14
55	96.22	97.58	92.51	75.41	91.04

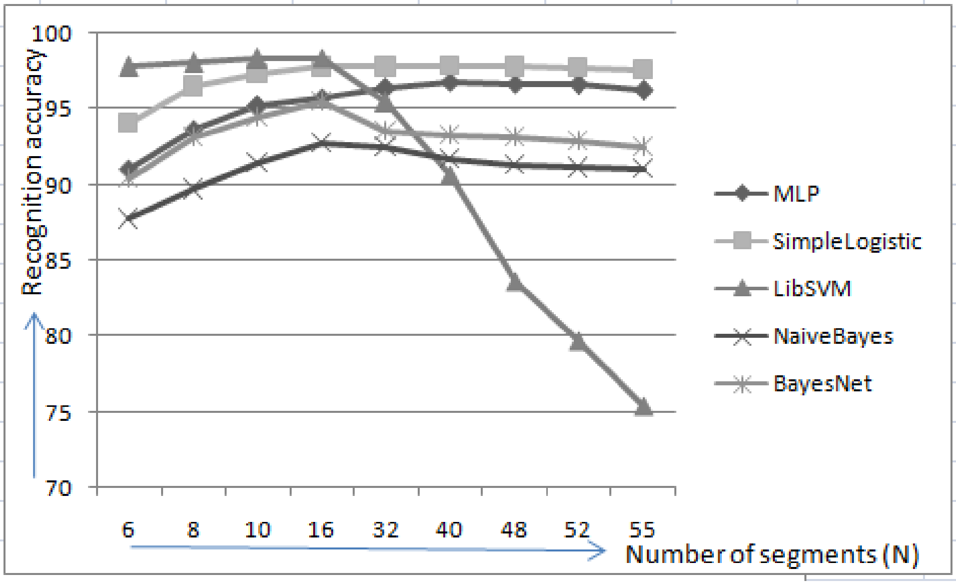


Figure 5.1.1: Graphical description of performance of the different classifiers while recognizing the online handwritten Bangla basic characters with varying number of segments

5.1.1.2 Recognition by Hausdorff Distance based Feature [102]

To test the effectiveness of the feature vectors produced by HD and DHD based procedures for the recognition of handwritten Bangla characters, some well-known classifiers such as MLP, SVM, BayesNet, Simple Logistic have been applied. 5-fold cross validation scheme on the total dataset (10000) has been applied. Table 5.1.2 reflects the recognition rates of the said classifiers for both the feature extraction strategies when a sample character is divided into 4, 9, 16, 25 rectangular zones. By analyzing the data as recoded in Table 5.1.2, it can be said that for all the classifiers success rate increases as division of character sample increases from 4 to 16 for both the feature extraction procedures. Dividing the character into more number of zones means more number of components constituting the character have been produced; thus more discriminative feature set can be obtained for the recognition of the character under consideration. In contrast, when specimen sample is divided beyond 16 zones then success rate starts falling, as observed from Table 5.1.2. This happens because when character is divided into say, 25 rectangular zones then number of components gets increased and thus size of the components gets decreased as well and it becomes less informative; as a result overall recognition accuracy declines. Here, MLP produces best recognition of 95.57% when DHD based features are used and sample character is divided into 16 rectangular zones. It has also been observed that irrespective of the zoning scheme and the classifiers applied, DHD based feature extraction procedure outperforms HD based technique. This is because feature set produced by the DHD procedure can be considered as the superset of the feature set produced by HD technique; as the former one considers both forward and backward distances as feature component whereas, HD based procedure considers only maximum of forward and backward HD as feature value. The graphical behaviour of all the classifiers, used here, for both the feature extraction approaches has been shown in Fig. 5.1.2. Here, blue, red, green, and violet lines represent the nature of classifiers when sample character is divided into 4, 9, 16 and 25 rectangular zones respectively. From the Fig. 5.1.2, it can be easily observed that green line representing 16- rectangular zones, always lies on top of all the other lines irrespective of the feature extraction approaches. It has also been observed from this graph that recognition rates of all the said classifiers are higher for DHD based feature calculation than HD based computation.

Table 5.1.2: Success rates of different classifiers for both DHD and HD based feature estimation procedures considering different zoning schemes (bold styles data indicate maximum accuracy for a particular zoning scheme and shaded cell indicates maximum accuracy achieved by the present technique irrespective of the zoning scheme or feature extraction procedure)

Classifier	HD based features				DHD based features			
	Number of Zones				Number of Zones			
	4	9	16	25	4	9	16	25
MLP	55.9	83.67	91.97	91.75	73.97	89.59	95.57	93.8
BayesNet	46.09	79.78	88.31	88.1	67.15	82.25	88.61	88.24
SVM	52.29	92.16	92.8	92.4	85.7	93.64	94.43	93.16
Simple Logistic	41.18	83.19	91.97	91.88	66.99	90.81	95.36	94.85
NaiveBayes	36.32	49.68	56.89	71.79	59.21	59.5	59.98	72.45

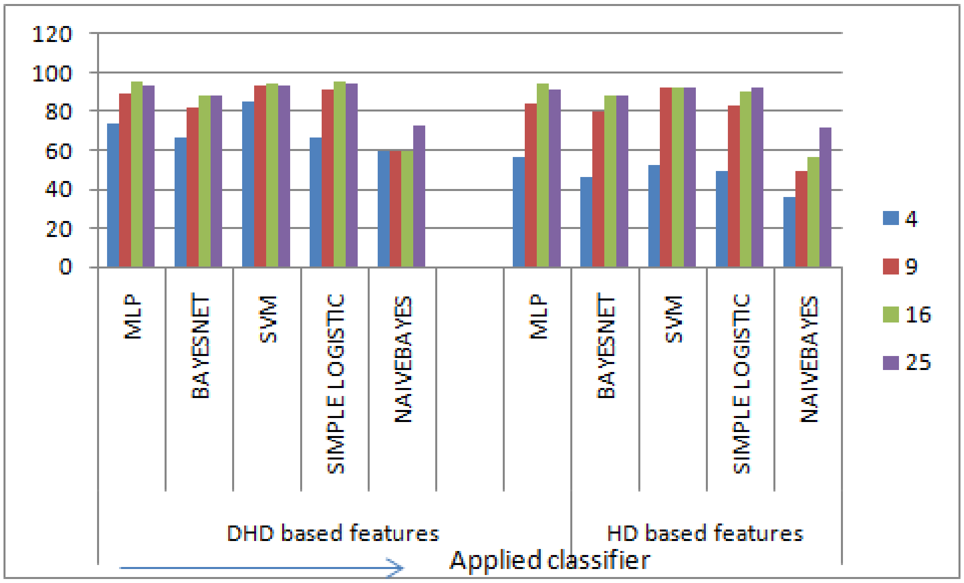


Figure 5.1.2: Graphical behaviour of the classifiers for DHD and HD based feature estimation procedures considering different zoning schemes

5.1.1.3 Recognition by Transition count features combined with CG-base circular and Topological Features [103]

This section presents the effectiveness of transition count and combination of CG based circular feature extraction methods for online Bangla handwritten character recognition. Transition count feature describes the transition from background to foreground pixels and vice versa, called transition count feature. The second method combines topological features and CG based circular features where global information, local information and CQMD information have been extracted. The impact of each feature vector along with their combination has also been analyzed.

Table 5.1.3 reflects the different combination of the features used for the experiment and the dimension of all the feature vectors. Table 5.1.4 reports the individual and combined impact of global and local features in terms of recognition rates observed by the classifiers [100]. Table 5.1.5 describes the strength of individual and the combined feature vectors; transition count feature and CG-based circular (global, local and CQMD) feature with topological feature [103]. The different approaches are applied on total of 10000 isolated online Bangla character samples for the evaluation. These feature vectors are fed to 5 well-known classifiers viz., SVM, MLP, BayesNet, Simple Logistic and NaiveBayes. A 5-fold cross validation scheme is applied on the character dataset.

Table 5.1.3: Illustration of feature set and their combination applied for online Bangla character recognition

sl#	**Features used**	**Feature length**	
1	Transition feature	224	
2	Global + local features	136	
3	Global + local + CQMD Features	160	
	Topological feature	5	
4	Transition + Global + local + CQMD + topological features	Before PCA	389
		After PCA	124

It is evident from Table 5.1.4 that the performances of the mentioned classifiers are better for the combination of global and local features (98.26%) than considering only global (96.56%) or local features (50.68%). This is because the combined feature vector rightly estimates the shape information of the specified character.

From Table 5.1.5, it can be observed that in case of the transition count based feature, the SVM produces an accuracy of 91.36% which is the best score among all the classifiers. Note that the combination of CG based circular features with topological features produces feature vector of length 165 (160+5). To reduce the feature dimension, PCA is applied. The resultant feature vector is then evaluated on the said classifiers. It has been observed that in this case also SVM produces the best accuracy of 98.48% . For CG-based circular feature estimation, the character image is rotated of by 45° to get some valuable information that can be used

Table 5.1.4: Accuracies of different classifiers in recognizing online handwritten Bangla characters

Classifiers	Success Rate (%)		
	Global feature	Local feature	Combined feature
SVM	96.56	50.68	98.26
MLP	86.50	47.68	91.50
BayesNet	82.89	37.40	88.24
Simple Logistic	80.32	39.02	90.00
NaiveBayes	82.30	41.42	84.59

to recognize skewed or slanted character samples. To validate this, the feature vector generated by combining topological features and CG based circular features without any rotation is tested. In this situation, the SVM produces the highest recognition accuracy of 95.84%. Clearly, this accuracy is less than the accuracy obtained when topological features are combined with CG-based circular feature considering rotation of the images by 45°.

When the transition count based features are added with the combination of the topological features with CG based circular features, total 389 (224 + 165) features are produced. Again, PCA is carried out and the resultant feature vector (124-element) is fed to the classifiers. At this stage also SVM produces the best result with 98.7% success rate. C-SVC type SVM from LibSVM library has been selected and its kernel type is polynomial: $(gamma * u * v + coefo)^{degree}$ having eps=0.001, coefo=0 and gamma=0.0 (default value). Here, the values of gamma and coefo are tuned to observe their impact on the system. Changing the gamma value does not make significant changes in the final outcome but when tuned the value of coefo (0, 1, 2, 3) the performance of the system increases gradually. If the value of coefo is further increased beyond 3, then the performance of the system degrades. The tuned value of coefo and the corresponding outcomes are shown in Table 5.1.6. From this table we have observe that the recognition accuracy increases up to 99.48% when the value of coefo is set to 3. Fig. 5.1.3 highlights the graphical representation of success rates achieved for transition count based features, a combination of topological features with CG based circle approaches, and the combination of transition count and CG based circle features with topological features. If Fig. 5.1.3 is closely observed then it can be seen that the individual impact of transition count based feature and combination of topological feature with CG based circle approaches is far less than the combination of these two. This indicates that the feature set produced by combining the transition count and CG based circular approaches along with topological feature has the ability to recognize the online Bangla character sample with highest level of accuracy.

Table 5.1.5: Accuracies of different classifiers in recognizing online handwritten Bangla characters

Classifier	**Success rates (%)**		
	Transition Count Feature	**Global Information + Local Information + CQMD + Crossing Point**	**Transition Count Feature + Global Information + Local Information + CQMD + Crossing Point**
SVM	91.36	98.48	**98.7**
MLP	73.15	89.26	96.14
BayesNet	74.37	86.91	91.42
Simple Logistic	87.29	90.58	93.81
NaiveBayes	60.74	85.34	88.75

Table 5.1.6: Accuracy obtained by tuning the parameter gamma of SVM classifier

Value of gamma	**Value of coefo**	**Database size**	**Accuracy (in %)**
0	1	10000	99.28
	2		99.46
	3		**99.48**
	4		99.41

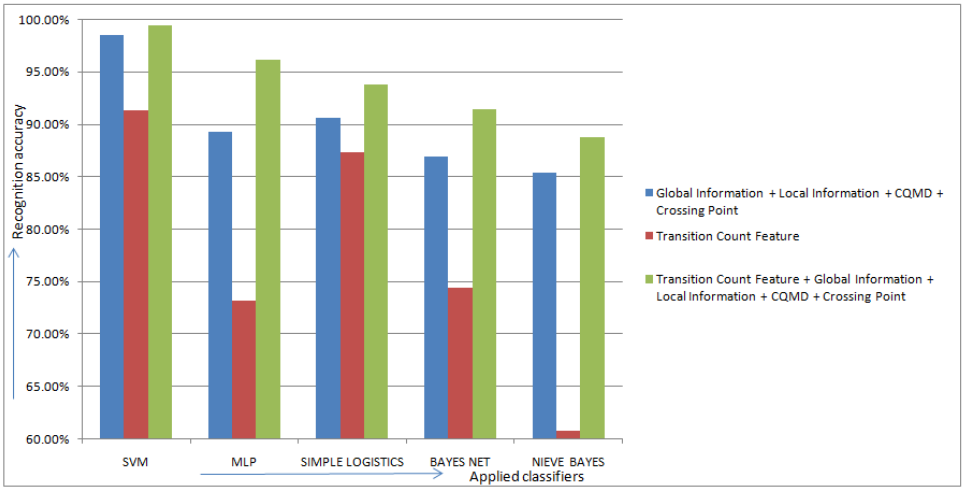


Figure 5.1.3: Graphical representation of the success rates for different classifiers in recognizing online handwritten Bangla basic characters

5.1.1.4 Recognition by Area and Local Features [98]

After applying quad-tree based image segmentation scheme, the individual and combined strengths of area feature, chord length and mass distribution mentioned in section 4.2.1.4 are described here for the recognition of handwritten online Bangla characters. Table 5.1.7 reflects the number of feature counts for all possible combinations that are fed to the classifiers at varied depths of the tree.

Table 5.1.8 (a-b) highlights the recognition accuracies, achieved through some well-known classifiers like MLP, Simple logistic, BayesNet, SVM and NaiveBayes when different quad-tree depths are followed for segmenting the images. Here, 5-fold cross validation scheme has been applied on the total 10000 dataset. From Table 5.1.8, it is noticed that irrespective of applied feature set and classifiers, success rates gradually increase as depth of the quad-tree structure increases for segmenting the image. This statement is true for depth up to three. When depth of the tree is increased beyond this level then performances of the said classifiers start decreasing.

Analyzing Fig. 4.2.6 in section 4.2.1.4 [98], it can be stated that with increas-

Table 5.1.7: Feature counts for all possible combinations at various quad-tree depths

Features used	Quad-tree depth			
	One (2X2)	Two (4X4)	Three (8X8)	Four (16X16)
Composite Simpson's Feature (1)	4	16	64	256
Mass Distribution (2)	4	16	64	256
Chord Length (3)	4	16	64	256
(1+2)	8	32	128	512
(1+3)	8	32	128	512
(2+3)	8	32	128	512
(1+2+3)	12	48	192	768

ing quad-tree depth, closer view of the character sample is obtained which in turn decreases the size of component parts in the respective blocks. Up to depth level three, features estimated on obtained segments of the character sample, relatively smaller in size, become informative enough to classify them. When character is further divided into more number of blocks, by increasing depth of the tree, blockwise components are becoming so small in size. As a result, features estimated from these segments become less informative and thus fail to identify the character samples properly.

Gray cells, in Table 5.1.8, show their best performances for different feature combinations. Bold styled values specify the name of feature extraction procedure that reflects top recognition at a particular depth. From Table 5.1.8, this is observed that, in the present experiment, SVM outperforms all other classifiers to yield the recognition accuracy of 98.5% (marked as bold styled and coloured in red) when all three feature sets are combined and depth of the quad-tree is three. Fig. 5.1.4 graphically describes the outcomes of the different experimentations performed under the current work. Blue, red, green, and violet lines reflect the nature of the classifiers when features are extracted from the images segmented by quad-tree based approach at depth one, two, three and four respectively. It is clearly seen from Fig. 5.1.3 that green line is always at top position and the out-

Table 5.1.8: (a-b) Performance of different classifiers for recognition of online handwritten Bangla characters when individual and various combinations of estimated feature sets are applied at level three of quad-tree based image segmentation

(a)

Feature sets applied	Classifier applied at different quad-tree depth									
	2 X 2					4 X 4				
	MLP	Simple Logistic	BayesNet	SVM	Naive Bayes	MLP	Simple Logistic	BayesNet	SVM	Naive Bayes
C1	58.6	46.83	50.5	56.3	50.5	87.9	86.36	88.4	93.7	80.5
C2	55.9	45.32	47.1	47.7	53.93	89.2	89.09	87.9	96.9	82.9
C3	60.5	44.73	52.9	58.62	46.4	90.7	89.36	89.9	95.97	80.9
C1 + C2	84.3	78.43	67.9	78.32	68.5	96.1	96.98	91.7	93.75	86.3
C1 + C3	82.1	75.09	62.6	70.91	57.7	93.6	93.24	91.0	93.76	82.4
C2 + C3	86.1	79.54	66.0	75.38	62.4	97.3	97.43	92.2	98.22	85.8
C1 + C2 + C3	92.6	91.18	71.3	83.71	69.2	97.6	97.23	92.4	93.77	85.9

(b)

Feature sets applied	Classifier applied at different quad-tree depth									
	8 X 8					16 X 16				
	MLP	Simple Logistic	BayesNet	SVM	Naive Bayes	MLP	Simple Logistic	BayesNet	SVM	Naive Bayes
C1	88.5	94.22	92.1	96.0	84.2	87.2	93.15	92.1	90.35	81.6
C2	90.1	96.03	91.0	98.02	85.3	88.4	95.02	95.38	95.02	82.5
C3	91.4	95.21	93.1	97.43	85.5	89.2	94.09	93.1	92.84	83.9
C1 + C2	98.1	97.33	92.6	98.15	86.4	96.0	96.28	92.6	94.71	85.9
C1 + C3	96.8	96.5	93.3	97.5	85.6	94.3	94.82	93.3	93.15	84.3
C2 + C3	97.9	97.4	93.2	98.24	86.6	94.5	96.24	93.2	95.22	85.1
C1 + C2 + C3	98.2	97.5	93.5	98.5	86.8	96.4	96.26	93.5	94.88	86.1

come for SVM classifier is best at this depth.

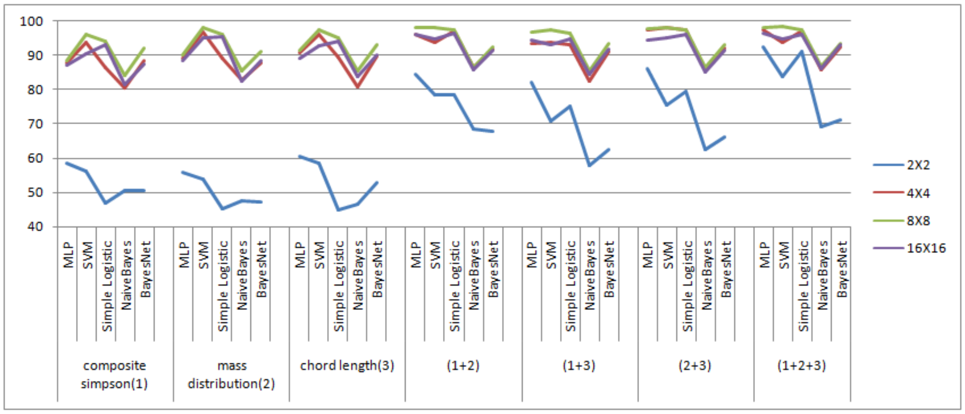
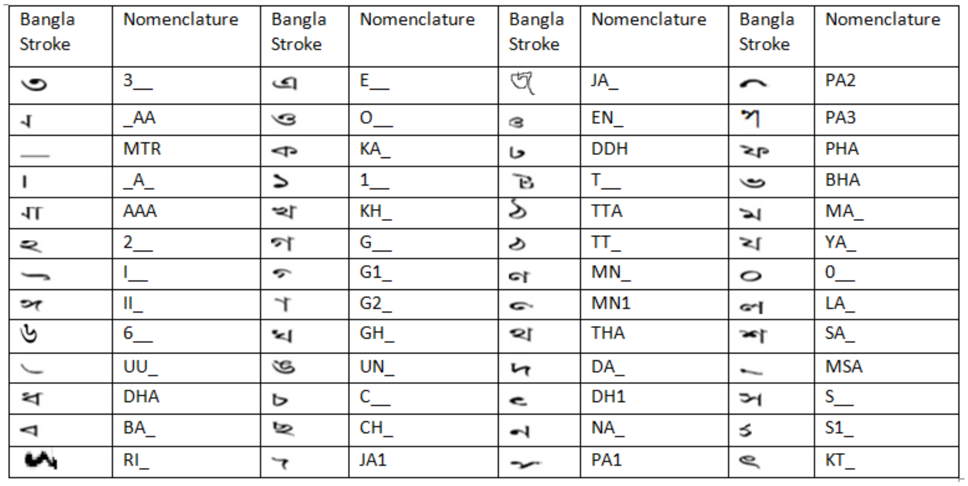


Figure 5.1.4: Graphical behaviour of the classifiers for all combinations of feature estimation procedures considering different depths of the quad-tree

In this Chapter, different types of features (distance based, hausdorff distance based, CG based circular with topological features and few local features in terms of area, chord length, mass distribution) have been discussed for character recognition holistically. In spite of the strength of the features discussed in earlier sections to recognize characters efficiently, few characters are misrecognized as some other symbols. Analyzing the misrecognized characters for different feature extraction strategies, some common misclassified character pairs are noticed like: (চ, ঢ), (ম, স), (ত, ভ), (ই, ঈ). Looking into the symbols, this can be said that the sole reason for misclassification is due to strong structural similarity between symbols.

5.1.2 Character recognition by stroke based approach

This section describes stroke based character recognition methods adopted in this thesis. In this procedure, strokes are considered in feature extraction phase, the recognition results of the same are passed through post-processing module in order to form the characters from the underlying recognized strokes. Strokes cannot be collected separately but they are extracted from collected character or text information. Strokes at character level can be stored easily but the same is not true for text level. People generally write characters in single or by using multiple strokes and these are treated as basic strokes. After analyzing all the strokes from the character database considered, it has been found that there are 52 different stroke symbols as shown in Fig. 5.1.5. This section elaborately describes stroke recognition accuracies achieved by applying features mentioned in section 4.2.2 on strokes. The character recognition accuracies by adopting rule based [101] and DFA based approaches [105] for the formation of characters is also describes in this section.



Bangla Stroke	Nomenclature	Bangla Stroke	Nomenclature	Bangla Stroke	Nomenclature	Bangla Stroke	Nomenclature
	3__		E__		JA_		PA2
	_AA		O__		EN_		PA3
	MTR		KA_		DDH		PHA
	_A_		1__		T__		BHA
	AAA		KH_		TTA		MA_
	2__		G__		TT_		YA_
	I__		G1_		MN_		0__
	II_		G2_		MN1		LA_
	6__		GH_		THA		SA_
	UU_		UN_		DA_		MSA
	DHA		C__		DH1		S__
	BA_		CH_		NA_		S1_
	RI_		JA1		PA1		KT_

Figure 5.1.5: Basic strokes of Bangla character set

5.1.2.1 Rule based approach [101]

In this approach, the recognition module has been divided into two parts: (i) Recognition of strokes and (ii) Formation of valid character from recognized strokes.

- Recognition of strokes

  Based on the distance based features mentioned in section 4.2.2.2, an MLP based classifier is adopted for the recognition of the strokes. This experiment designs a 2-layered MLP for the recognition of handwritten online Bangla strokes. The numbers of neurons in the input and output layers of the perceptron are considered as 136 and 52 respectively. The number of neurons in hidden unit is set to 85 whereas the learning rate and acceleration factor of the back propagation algorithm are set to suitable values, based on trial runs. A network of 136-85-52 is thereby designed.

- Formation of valid character from recognized strokes

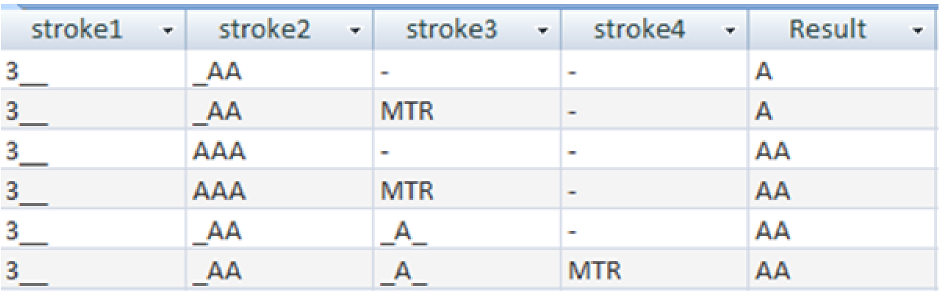
  Each character is constructed with the help of its recognized strokes. To do so, firstly, after extracting all the strokes from all the characters, strokes are classified into three categories depending on positions of the same.

  Major Stroke: A stroke is called Major stroke if it occupies major portion of the character symbol. For the character 'অ' the major stroke is '৩'.

  Minor Stroke: A stroke is called Minor stroke if it occupies minor portion of the character. For the character 'অ' the minor stroke is '◌া'.

  Matra(MTR): A stroke is called Matra if it is a horizontal straight line resides at the upper part of the character. For the character 'অ' the matra stroke is '-'.

  This categorization plays an important role at the time of forming characters from the identified strokes. To minimize the problem that may arise due to different stroke order even in case of same character, Major stroke and Matra are identified and they are placed at first and last positions respectively in stroke sequence. This reduces the possible number of stroke order combinations while forming the character. A rule-based approach is then constructed to form all such stroke-order combinations. Few such rules for recognizing characters 'অ' and 'আ ' are shown in Fig. 5.1.6



stroke1	stroke2	stroke3	stroke4	Result
3__	_AA	-	-	A
3__	_AA	MTR	-	A
3__	AAA	-	-	AA
3__	AAA	MTR	-	AA
3__	_AA	_A_	-	AA
3__	_AA	_A_	MTR	AA

Figure 5.1.6: Rules for constructing characters 'অ' and 'আ '

The experimental evaluation of the above technique has been carried out on isolated Bangla strokes. A total of 32, 534 strokes are prepared from the collected online handwritten Bangla basic characters. In this work, MLP classifier produces 89.39% accuracy in recognizing strokes of online Bangla basic characters. It is to be noted that, recognition of handwritten online Bangla basic characters from constituent strokes using rule-based technique heavily depends on stroke recognition result.

5.1.2.2 DFA based approach [105]

In this approach, the recognition module has been divided into two parts: (i) Recognition of strokes and (ii) Formation of valid character from recognized strokes.

- Stroke Recognition

  The combination of Distance based and ZPT based feature vectors produces an 200 (i.e. $136 + 64$) element feature vector. An MLP based classifier is used for stroke recognition and it achieves 91.27% classification accuracy. A 2-layered MLP (200-98-52) has been designed for classification purpose. The input, output and hidden layer neurons are set to 200, 52 and 98 respectively. Values of acceleration factor, learning rate of BP algorithm have been finalized after few trial runs.

- Character Formation

It has been mentioned earlier that there are few strokes that are used to write various/multiple character samples (see [82] for detail). From this table, we have observed that few strokes (e.g. BA_, 3__, 6__) describe the major part of the character, while some strokes help to form different characters as a minor part of the character (e.g. 0__, _A_). Considering these facts, the strokes are categorized into three groups Major Stroke, Minor Stroke and Matra as mentioned in the earlier section.

In this work, all the strokes are analyzed that are used to write Bangla basic characters. From this analysis, a list of strokes has been prepared that act as major parts for the construction of multiple characters (major stroke list). After tracking the major stroke of a character sample, the stroke sequence is rearranged by placing the major stroke and Matra at the start and end positions respectively to minimize the stroke order variation. Then the procedure starts with a node representing a major stroke and constructed a DFA to describe different paths for all possible characters that can be derived from this major stroke. Fig. 5.1.7 illustrates on such DFA to recognize the characters 'ড', 'জ' and 'ড়' from major stroke 'ড'.

This major stroke list is important for the construction of DFAs because for each such major stroke, a separate DFA needs to be formed and most of the symbols belonging to Bangla basic alphabet set contains a major stroke. Please note major strokes are the starting node in the DFA. There are also few characters that may not contain any major stroke but each of them contain at least one such stroke that can distinguish the character from others and treated them as starting node of DFA for that character (e.g. 'গ' can be written by using strokes *G*1_ and *G*2_ and both the strokes are minor strokes but *G*1_ can only be present character 'গ').

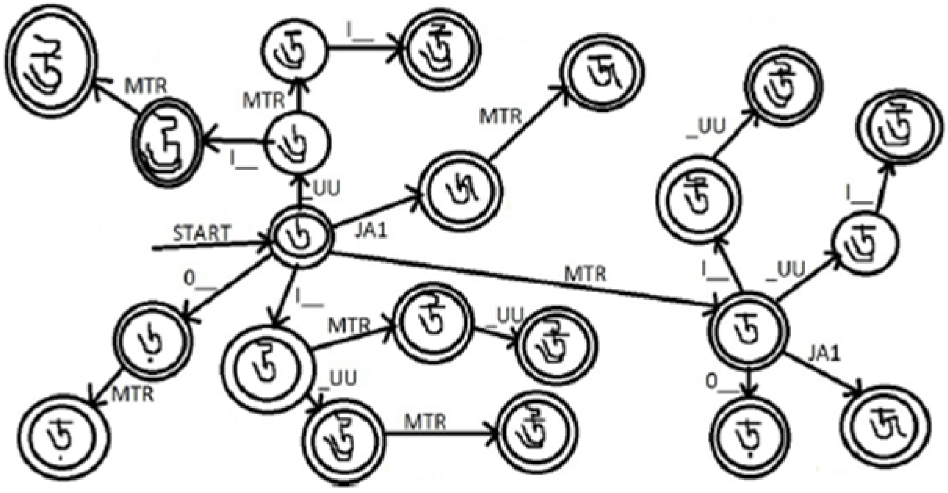


Figure 5.1.7: Formation of different characters from major stroke 'ড'

5.1.3 Character recognition by deep learning approach

In this experiment, 50 Bangla character classes are considered, where each class contains 200 different samples. The impact of different learnable parameters of CNN such as kernel size (3x3, 5x5 and 7x7), pooling strategies (Max_Pooling and Avg_Pooling) and activation functions (Softmax and Sigmoid) on the overall recognition model are analyzed. Here, 30% of the entire characters have been considered as test set and rest 70% have been used to train the recognition model.

In this work, a deep learning based architecture, CNN has been used for recognition of online handwritten Bangla characters which has a input layer, two convolutional layers and two pooling layers followed by a fully-connected classification layer with output layer (see Fig. 5.1.8).

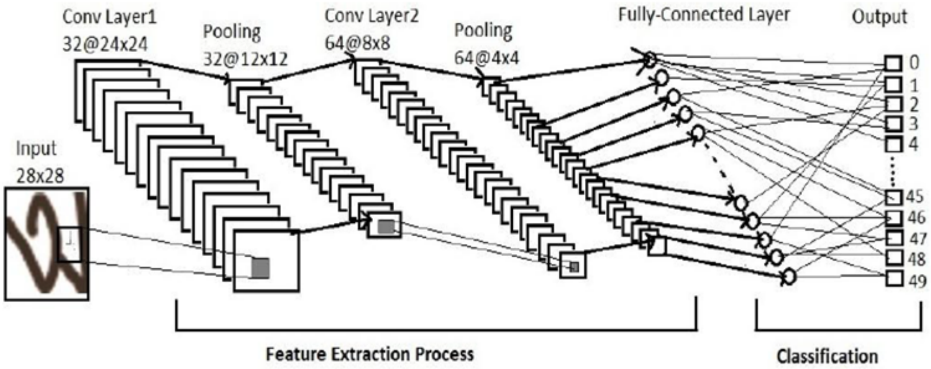


Figure 5.1.8: Architecture of CNN used in the present work

The input layer contains 784 nodes for a 28x28 pixels image. Convolution layer is the first layer through which the input image passes. The functionality of this layer involves arbitrary number of learnable filters to move along the width and height of the image to produce feature map. A filter can be considered as an array of numbers where the numbers are called weights or parameters.

After sliding the filter over all the locations, a 24x24 array of numbers have been achieved, which is known as activation map or feature map. The first convolution layer produces 32 such feature maps. Convolution operation with kernel of spatial dimension 5 (i.e. size of 5x5) converts 28 spatial dimension to 24 (i.e., 28-5+1) spatial dimension [79]. So, size of the first level feature maps becomes 24x24.

The output ($32@24x24$) of first convolution layer goes to first pooling layer where each feature map has been resized. In this experiment, both Max_pooling and Avg_pooling schemes have been applied. For both the cases at the first pooling layer, feature maps (produced at first convolution layer) have been down-sampled from 24×24 into 12×12 feature maps by applying a local averaging with 2×2 area, multiplying by a coefficient, adding a bias and passing through an activation function. Formally, it can be expressed by Eq. 5.1 [79].

$$x_j^l = f(\beta_j^l down(x_j^{l-1}) + b_j^l) \tag{5.1}$$

Where, down(.) represents a pooling function through local averaging; β and b are multiplicative coefficient and additive bias, respectively. The output image becomes twice smaller in both spatial dimensions for 2×2 local area averaging in down(.) function. This operation reduces the spatial resolution of the feature map.

Following the similar fashion, second convolution and pooling operations have been performed. In this work, second convolutional operation produces 64 distinct feature maps. A 5×5 filter produces a feature map size of 12×12 into 8×8. Then second pooling operation resizes each feature map to size of 4×4. These 64 features map values are considered as 1024 (=64×4×4) distinct nodes those are fully connected to 50 feature maps (corresponding to each character class). In CNN, the error (E) has been minimized by using Eq. 5.2 [79].

$$E = \frac{1}{2}\frac{1}{PO}\sum_{p=1}^{P}\sum_{o=1}^{O}(d_o(p) - y_o(p))^2 \tag{5.2}$$

Where, P is the total number of patterns (here 10000); O is the number of output nodes (i.e., 50); d_o and y_o are the desired and actual outputs of a node respectively for a particular pattern p.

It is to be noted that padding of 0 valued 2 pixels has been done on all four sides of an image after passing through each convolutional layer to retain the previous size. In the current experiment, two different activation functions (Softmax and Sigmoid) have been used at the fully connected network during classification in the output layer. The Sigmoid function is generally used for the two-class logistic regression, whereas the Softmax function is used for the multiclass logistic regression. Corresponding predicted probabilities that can be computed by using Sigmoid and Softmax functions have been mentioned in Eqs. 5.3-5.4 [54, 71].

$$F(X_i) = \frac{1}{1 + Exp(-X_j)} \tag{5.3}$$

$$F(X_i) = \frac{Exp(X_i)}{\sum_{j=0}^{k} Exp(X_j)} \tag{5.4}$$

Both Softmax and Sigmoid function use cross-entropy loss function for error calculation and can be described by Eq. 5.5 [56].

$$E(t, o) = -\sum_{j} t_j log o_j \tag{5.5}$$

where t and o act as the target and output at neuron j respectively. The sum is carried on over each neuron in the output layer. o_j denotes the result of the Softmax function $F(X_i)$.

Table 5.1.9 summarizes the outcomes for all possible combinations of those variations on test set. From this table, it has been observed that recognition accuracy has reached to maximum (shown in bold style in Table 5.1.9) when filter size is 5x5, Max_Pooling is the pooling scheme and Softmax is the activation function. As the current experiment has been conducted on image of size 28x28, so, if filter size becomes larger (i.e. 7x7) then it fails to capture minute detail of the structurally similar character patterns. Whereas, if the filter size becomes very small (i.e. 3x3), then it may generate some redundant information which, in turn, would reduce the recognition ability of the model. Hence, 5x5 is proven to be optimal filter dimension. The experimental result shows that in this recognition procedure, the Softmax activation function becomes more effective than sigmoid. Generally pooling is used to reduce variance, computation complexity and extract low level features from neighbourhood. Although both 2x2 Max_pooling and Avg_pooling reduces half of the image size by dividing it into four equal quadrants, Max_pooling considers maximum value presented in each quadrant and hence can extract the most important features like edges. But the, Avg_pooling may overlook some important information due to averaging out all the values. Hence, in general,

Table 5.1.9: Achieved classification accuracies by CNN with varying kernel size, pooling scheme and activation function

Kernel size	Pooling scheme	Activation Function	Test Accuracy (in %)
3X3	Max_Pooling	Softmax	99.27
	Avg_Pooling		99.13
	Max_Pooling	Sigmoid	99.07
	Avg_Pooling		99.03
5X5	**Max_Pooling**	**Softmax**	**99.40**
	Avg_Pooling		99.27
	Max_Pooling	Sigmoid	99.33
	Avg_Pooling		99.00
7X7	Max_Pooling	Softmax	99.23
	Avg_Pooling		99.13
	Max_Pooling	Sigmoid	99.20
	Avg_Pooling		99.03

Max_pooling has the better potential than Avg_pooling.

5.2 Discussion:

In this chapter the recognition result of online handwritten Bangla characters (holistically) by applying five types of features (distance based, hausdorff distance based, transition count, CG based circular with topological feature and few local features in terms of area, chord length, mass distribution) have been discussed. By analyzing the results it has been noticed that highest recognition accuracy 99.48% achieved when transition count features are combined with CG based circular features (global, local and CQMD) and topological features on 10000 online handwritten Bangla basic characters by SVM classifier.

For stroke based character recognition methodology (on same 10000 character database), applying only distance based feature yields 89.39% correct stroke recognition by MLP classifier. When this feature vector is combined with ZPT feature vector then stroke recognition accuracy is increased by 0.88% and becomes

91.27% using the same classifier.

In deep learning based approach, a CNN has been employed for the recognition purpose on the same 10000 online Bangla character database. In CNN based method, feature extraction is not needed rather they are extracted by selecting suitable filter size in convolution time followed by pooling scheme. The great advantage of adopting this technique over the methods described in the previous two sections is to cut-off the overhead of computing handcrafted features because sometimes they do not ensure good performance. The recognition outcome 99.40% is also appreciable compare to some handcrafted features applied here.

6
Word Recognition

In the OHR domain, word recognition accuracy heavily depends on proper segmentation of word samples to obtain the valid component strokes. A valid component stroke is one that belongs to any of the basic stroke classes. The works described by Bhattacharya and Pal in [19], Bhattacharya et al. in [20], Ghosh in [46], have tried to collect stroke level information from target word by estimating busy zone over it and then applying the DUD approach for segmentation. To find busy zone of the word sample, an offline image is generated from online information. A horizontal histogram is then computed in order to identify the busy zone boundary. Proper segmentation is heavily dependent on busy zone formation, which in turn depends on writing style and the nature of the skew of the word sample. If the skew is straightforward (either upward or downward) then skew detection and correction methodologies could be adopted easily, but when the skew is multi-directional (i.e. constituting strokes are at different levels of height) then busy zone formation really becomes complex and generates under-segmentation errors. Moreover due to inherent writing pattern of characters like 'ল', 'শ' , DUD approach within busy zone causes certain over-segmentation issues. Ghosh has proposed a modified scheme in [46] for online Bangla word segmentation. Firstly, the word image is partitioned into two parts; upper zone and lower zone. Words are then segmented into a sequence of basic strokes by slightly modifying DUD concept. Author has proposed to segment the word at the point where, there is a downward movement in upper zone and the slope of six consecutive pixels satisfies certain angular value.

Building of an online handwritten Bangla word recognition system using a stroke-based approach is not a much explored research area. In the works mentioned in [19, 20, 46], authors have only tried to segment handwritten Bangla words to obtain constituent strokes from it. To the best of our knowledge no further work has been made in this area since 2012 either to overcome the problems (such as handling skew word samples and under/over segmentation issues) or to recognize the segmented strokes by adopting any suitable feature extraction strategy, that is the prior step to re-construct the words from the obtained segmented strokes.

In this chapter, a novel Bangla word segmentation technique based on stroke-level busy zone formation procedure has been proposed. In an unconstrained domain, people often write text where strokes may be poorly aligned (due to multi-directional skewness) and varied combination of strokes with various types of joining between them are possible while forming the words. Hence, a segmentation based approach for stroke extraction is pertinent for any stroke-based word recognition system. The presence of a large volume of symbols set (58 basic symbols with more than 280 compound characters) in Bangla script makes the task more challenging. In the current experiment, proposed stroke-level segmentation approach effectively handles such type of Bangla words. A sub-zoning scheme within busy zone followed by a modified Down->Up->Down (DUD) concept within these sub-zones has been used to find valid segmentation points. This scheme avoids over and under-segmentation issues caused by either inherent writing pattern or due to writing style variations up to certain extent. The proposed segmentation approach has been tested on 6, 500 online handwritten Bangla word samples.

In the present scheme, proposed a segmentation technique is used to obtain constituent basic strokes from online handwritten Bangla word samples and finally recognition of those strokes. For word segmentation context, firstly, the strokes are categorized into three lists based on their positional information; upper zone, lower zone and strokes-need-segmentation (SNS). A procedure is proposed to form busy zone on SNS strokes to handle multidirectional skew. Then a sub-zoning scheme within busy zone followed by a modified DUD concept within these sub zones has been used to find the segmentation points. The advantage and technical details of the stroke-level segmentation technique have been reported elaborately in the following subsections.

After performing the segmentation process, a stroke recognition module is required to validate the proposed segmentation technique. In stroke recognition module, some local information are estimated to produce discriminating features for the recognition purpose. Modified distance based feature discussed in Sen et al. [99, 101] and point based along with curvature based feature extraction technique [107] for Bangla stroke recognition have been used following the description mentioned in sections 4.2.2.2, 4.2.2.3, 4.2.2.4. After stroke recognition, an HMM based model has been built to construct words and a LM acts as a post processing module to help to predict correct choice from top-5 HMM recognition choices.

6.1 PROPOSED METHODOLOGY

6.1.1 Segmentation of words

In this stroke based segmentation approach, strokes are firstly categorized into three groups based on their positional information. These groups are named as upper zone, lower zone and SNS. The first two types are supposed to be free from segmentation for Bangla script but the SNS contains such type of strokes that need to be segmented. (The stroke in the SNS list contains multiple strokes with different types of joining between them due to writing without lifting up the pen). After categorization of the strokes, busy zone is formed over the strokes belonging SNS stroke list. To do so, strokes are vertically divided into two equal halves and then for each half, CG has been computed. Angle made by line joining between two CGs and X axis describes the degree of skew for the stroke sample. This line is then shifted upward and downward up to certain height of the stroke to form the busy zone of the stroke sample. This process enables us to handle skewed strokes. A modified DUD approach is then adopted for segmentation purpose by dividing the stroke-level busy zone into different sub-zones. Up to a certain limit, the sub zoning scheme helps to handle different joining patterns between strokes that are not aligned horizontally and finds the valid segmentation points. This scheme is also useful for the characters like 'ল', 'শ' etc. whose inherent writing pattern may mislead DUD approach causing over-segmentation. Algorithm 15 describes the overall procedure of the proposed segmentation algorithm. At the first step, constituent strokes are categorized into three lists. It is clearly seen from Fig. 6.1.1 (a), that strokes #2, #3 and #5 should be included into lower zone list that need no more segmentation. Whereas, strokes #1 and #4 should be included into SNS because they contain multiple connected strokes written in a single stretch (written without lifting up the pen) and thereby those need to go through some efficient segmentation process. Algorithm 16 describes the process of strokes categorization and Fig. 6.1.1 (b) pictorially describes this algorithm for first three strokes of Fig. 6.1.1(a).

Algorithm 15: Stroke based segmentation

```
STEP 1: Divide all the strokes into three stroke lists; upper_zone, lower_zone and SNS using Algorithm 16.
STEP 2: For strokes i=1 to N from SNS do,
STEP 3: Calculate busy zones for strokes using Algorithm 17.
STEP 4: Apply modified DUD approach within stroke-level busy zone for segmenting them into basic strokes.
```

During the execution of Algorithm 16, step 3.2 finds the set of strokes s_k within the width of s_i . For example, as shown in Fig. 6.1.1, when stroke #1 is considered then strokes #2 and #3 are also examined because these two strokes fall within the width of stroke #1. Due to nature of Bangla script it can be said that within $\{s_k + s_i\}$ one stroke should be major that needs to go through segmentation process and others may belong to upper zone or lower zone list or both that need not be segmented. Step 3.3 finds the major stroke s_t on the basis of pixels count from the set of overlapping strokes $\{s_k + s_i\}$. The major stroke is the one which contains maximum number of pixels. We have set this criterion because ascendant (upper zones) and descendant (lower zones) strokes generally contain less number of pixels than strokes in the middle zones (position between upper zone and lower zone) of a word.

Algorithm 16: Stroke categorization

```
STEP 1: For all strokes i=1 to N
STEP 2: $flag_i$=0
STEP 3: For all strokes $s_i$ where i=1 to N
STEP 3.1: if $flag_i$=0 then do
STEP 3.2: Find the set of strokes $\{s_k\}$ that contributes some pixels within
the width of $s_i$
STEP 3.3: Find stroke $s_t$ from set of strokes $\{s_k\}$ and $s_i$ that contains maximum number of pixels
STEP 3.4: Place $s_t$ in SNS and set $flag_t$=1
STEP 3.5: For each stroke $s_p$ from set of strokes $\{s_k + s_i - s_t\}$ do,
STEP 3.5.1: Check if CG of $s_p$ lies top of minY and maxY of $s_t$ then
STEP 3.5.1.1: Place stroke $s_p$ in upper_zone list and set $flag_p$=1
STEP 3.5.2: Check if CG of $s_p$ lies below minY and maxY of $s_t$ then
STEP 3.5.2.1: Place stroke $s_p$ in lower_zone list and set $flag_p$=1
```

In Fig. 6.1.1, stroke #1 qualifies as SNS because of having maximum number of pixel points. Then, CGs of all overlapping strokes #1, #2 and #3 have been computed that are termed as G1, G2 and G3 respectively. In Fig. 6.1.1, d1, d2, d3, d4 denote the distance measures from G2 and G3 to maxY and minY respectively of stroke #1. As d1 and d3 both have greater values than minY, as well as d2 and d4 have greater values than maxY of stroke #1, hence both the strokes #2 and #3 lie below the stroke #1 and thereby can be placed in the lower zone. Similarly, between strokes #4 and #5, first one goes into SNS whereas, later one falls into lower zone category.

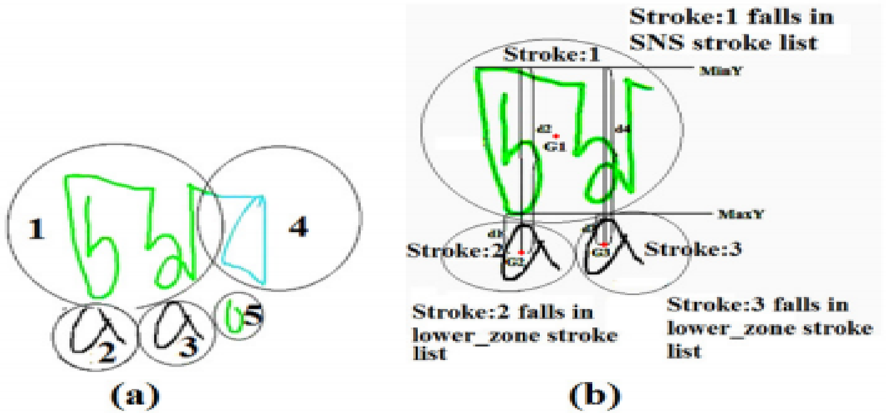


Figure 6.1.1: (a) Strokes of a sample word are numbered according to writing order, (b) Stroke categorization using Algorithm 16

After getting SNS, the busy zone has been formed by adopting Algorithm 17 to enter into the segmentation process. This helps to get rid of estimating busy zones for SNS which are poorly aligned due to writing variations of individuals.

```
Algorithm 17: Stroke-level busy zone formation
    STEP 1: Vertically divide a stoke into two parts.
    STEP 2: Calculate CG for both the left and right parts.
    STEP 3: Move the line joining two CGs up to Top_Line and Bottom_line
    to form busy zone of the stroke
```

Fig. 6.1.2 (a-c) describes the pictorial view of busy zone formation for strokes #1 and #4 from Fig. 6.1.1 (b). Fig. 6.1.2 (a) reflects the vertical division of strokes #1 and #4 based on calculated CGs. Fig. 6.1.2 (b) depicts the CGs for two sub-parts that are connected by drawing a line between them and the angle theta (θ) made by this line with x axis denotes the skew of the stroke. In this current experiment, detected skew is not corrected but this line is shifted in upward and downward directions (up to Top Line and Bottom line as shown in Fig. 6.1.2 (c) by taking the help of computed vertical histogram of the sample stroke) to form the busy zone for the stroke sample. This strategy minimizes the time complexity by reducing the extra calculation for skew correction. Again, as this is a stroke based word segmentation approach, hence poor alignment of strokes has minimal impact on the final segmentation of the words even if the words are skewed multi-directionally.

After forming stroke-level busy zone, the modified DUD concept has been adopted for stroke segmentation purpose to yield the required component strokes. Before adopting this concept, a minor change has been made in the area within Top_Line and Bottom_Line (as shown in Fig. 6.1.2 (c)). This zone is divided into five subzones to cope with the variability due to different structural patterns of strokes and also the writing variations of individuals. These sub-zones have been named as *Zone1* (Top_Line to Top_Line+t1), *Zone2* (Top_Line+t1 to Top_Line+t2), *Zone3* (Top_Line+t2 to Top_Line+t3), *Zone4* (Top_Line+t3 to Top_Line+t4) and *Zone5* (Top_Line+t4 to Bottom_Line) as shown in Fig. 5.3(a). Here, t1=height of busy zone/6, t2= height of busy zone/3, t3=height of busy zone/2 and t4= height of busy zone*3/4 where height of busy zone= Bottom_Line - Top_Line.

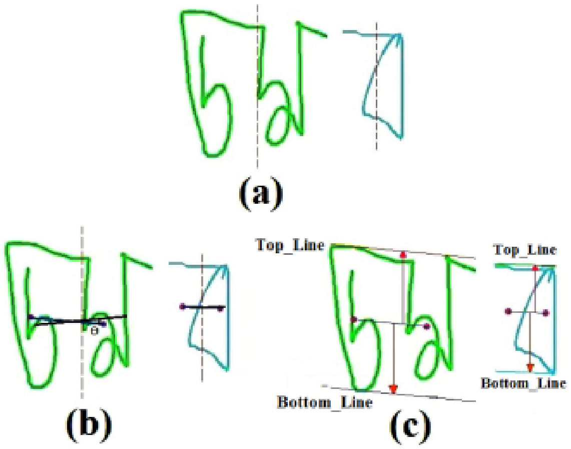


Figure 6.1.2: Vertical division of SNS strokes into two parts (b) measurement of skew angle (c) formation of busy zone

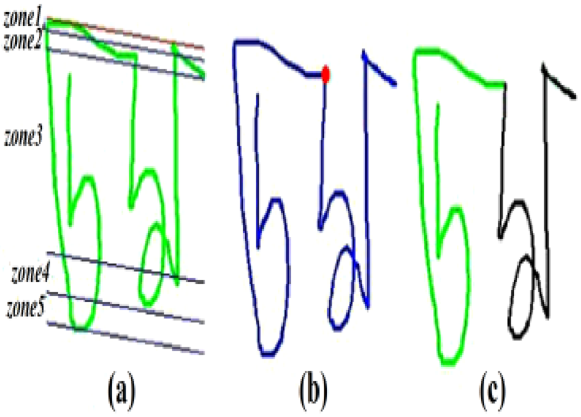


Figure 6.1.3: Division of busy zone into sub-zones, (b) identification of valid segmentation point, and (c) component strokes after segmentation

As the joining between characters in Bangla words are found in the matra region, hence we assume that segmentation takes place in the *Zone2* only. After forming sub-zones, the path/positions of all the pixel points (according to writing sequence) are named by either ”up” or ”down” on the basis of relative positional order of different sub-zones. The convention for the direction of ”up” to ”down” movement used here is shown in the Fig. 6.1.9 (b) and reverse direction is true for ”down” to ”up” movement. For example, if a pixel goes from *Zone2* to *Zone3* then it is named as ”down”, whereas, when pixel goes from *Zone3* to *Zone2* then it is named as ”up” and pixels within the same region get the same naming convention. Initially it starts by ”do not know” status. Then depending on writing style of the stroke, pixels are named as ”up” or ”down” by looking at the movement of the pattern through different sub-zones. Now the strokes (having multiple connected components) is segmented in *Zone2* region when it tries to go downward, i.e. towards *Zone3*, after checking the movement over different subzones from start or previous segmentation point of the stroke to the point of consideration. If this movement satisfies any of the following five conditions (described from case 1 to case 5) then only segmentation takes place. Figs. 6.1.4-6.1.8 depict correct segmentation outcomes for these five cases respectively. Red marked circles highlights the joining between strokes.

Case 1: Movement starts from *Zone1* or *Zone2*, and then follows the sequence *Zone3->Zone4->Zone3->Zone2/(Zone2->Zone1->Zone2)->Zone3*

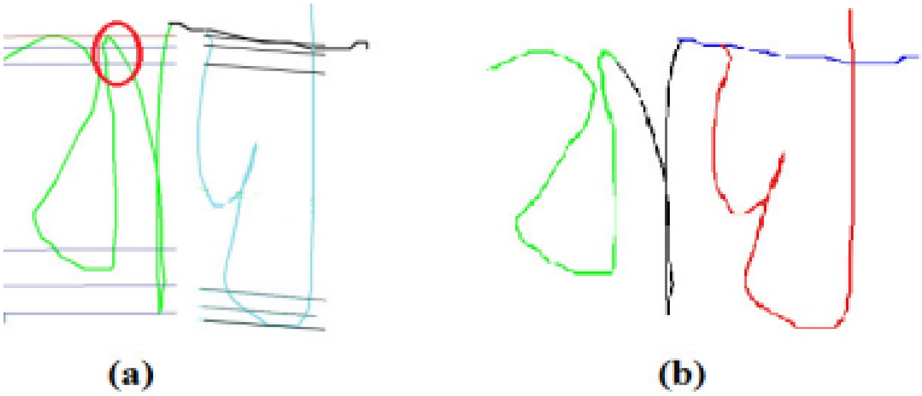


Figure 6.1.4: (a) Stroke sample follows pattern movement (movement of pixels) as mentioned in case 1, (b) Segmented elementary strokes (elementary strokes are represented by different colours)

Case 2: Movement starts from *Zone1* or *Zone2*, and then follows the sequence *Zone3->Zone4->Zone5->Zone4->Zone3->Zone2/(Zone2->Zone1->Zone2)->Zone3*

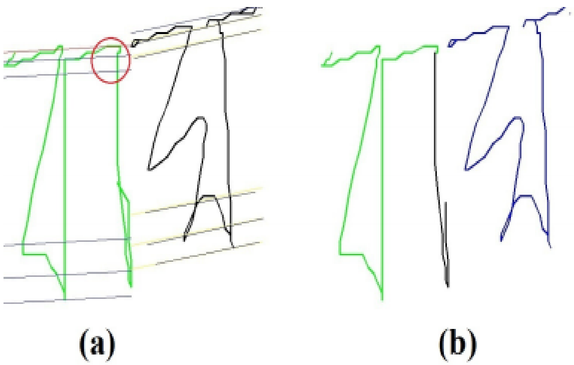


Figure 6.1.5: (a) Stroke sample follows pattern movement (movement of pixels) as mentioned in case 2, (b) Segmented elementary strokes (elementary strokes are represented by different colours)

Case 3: Movement starts from *Zone3* and follows the sequence *Zone4->Zone5->Zone4->Zone3->Zone2->Zone3*

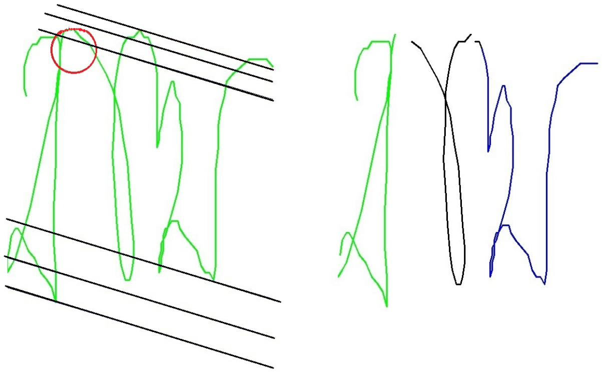


Figure 6.1.6: (a) Stroke sample follows pattern movement (movement of pixels) as mentioned in case 3, (b) Segmented elementary strokes (elementary strokes are represented by different colours)

Case 4: Movement starts from *Zone3* and follows the sequence *Zone4->Zone5->Zone4->Zone3->Zone2->Zone1->Zone2->Zone3*

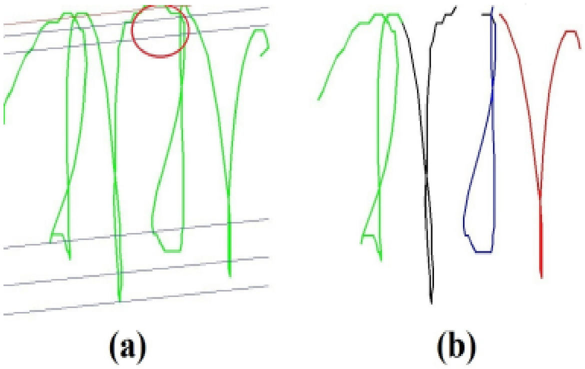


Figure 6.1.7: (a) Stroke sample follows pattern movement (movement of pixels) as mentioned in case 4, (b) Segmented elementary strokes (elementary strokes are represented by different colours)

Case 5: Movement starts from *Zone3* and follows the sequence *Zone4-> Zone3->Zone2->Zone1->Zone2->Zone3*

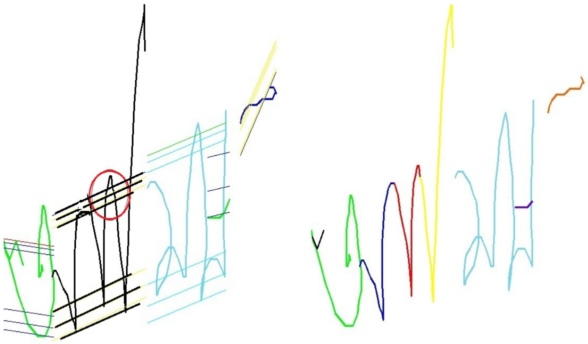


Figure 6.1.8: (a) Stroke sample follows pattern movement (movement of pixels) as mentioned in case 5, (b) Segmented elementary strokes (elementary strokes are represented by different colours)

Fig. 6.1.3 (b) reflects the positions of segmentation points after executing this modified segmentation scheme for stroke #1 as shown in Fig. 6.1.2 (c). Clearly stroke #1 joins two different symbols from basic alphabet set. The pattern movements of pixels for this stroke satisfy case 4 and hence it gets segmented at position shown in Fig. 6.1.3 (b). Fig. 6.1.3 (c) reflects the obtained component strokes after segmentation.

Fig. 6.1.9 (a-b) highlights the advantage of using sub-zoning scheme with modified DUD approach which is applied on two different word samples belong to SNS. Fig. 6.1.9 (a) displays the pixel contributions of a stroke (shown in black colour) within different sub-zones that contains two different isolated stroke symbols and depicts the faced complexity as well as the advantage of using this scheme for proper segmentation. The joining between the symbols is happened in such a way that they lie at different heights, thus making some skew. The proposed busy zone formation technique works well in this case too. From the figure it can be observed that second part of the stroke resides quite below of the TOP LINE which could have been misled the segmentation technique causing under-segmentation error, but the sub-zoning scheme along with modified DUD concept prevents such issues and helps to locate the proper segmentation point. From Fig. 6.1.9 (a), it is observed that the stroke satisfies Case 3 and thus gets segmented properly.

In Fig. 6.1.9 (b), the writing pattern of stroke 'ল' starts from *Zone3* and after crossing *Zone4* it enters into Zone1 and tends to move towards *Zone3*. Clearly, writing pattern follows a down- >up->down movement inside *Zone2* (see Fig. 6.1.9 (b)). But the stroke is not eligible for segmentation, because the previous stroke has not been traversed through *Zone5* and thereby it does not satisfy any of the conditions mentioned before.

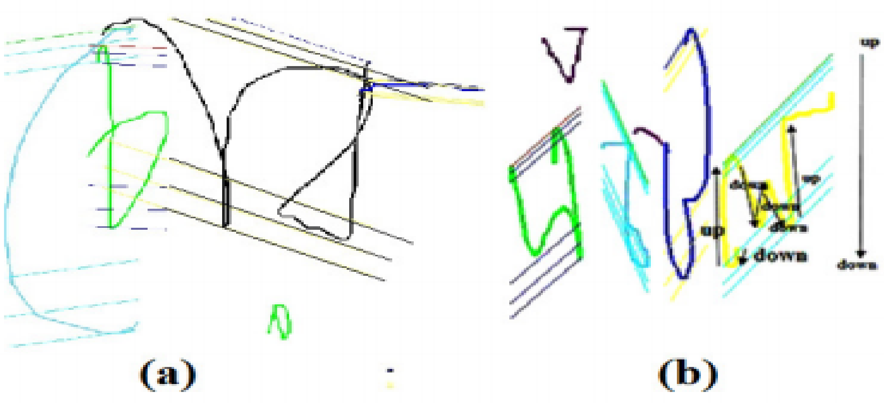


Figure 6.1.9: Sub-zoning scheme together with modified DUD approach to handle tricky stroke patterns for two different type of skewed words

The examples of the steps involved in the segmentation module for a skewed word sample are shown in Fig. 6.1.10 (a-e).

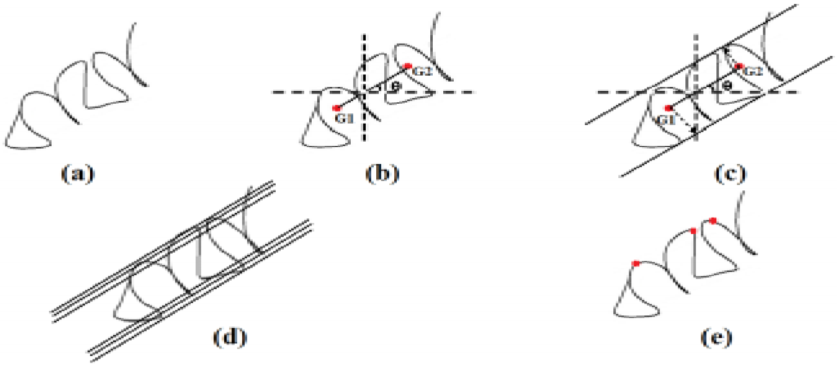
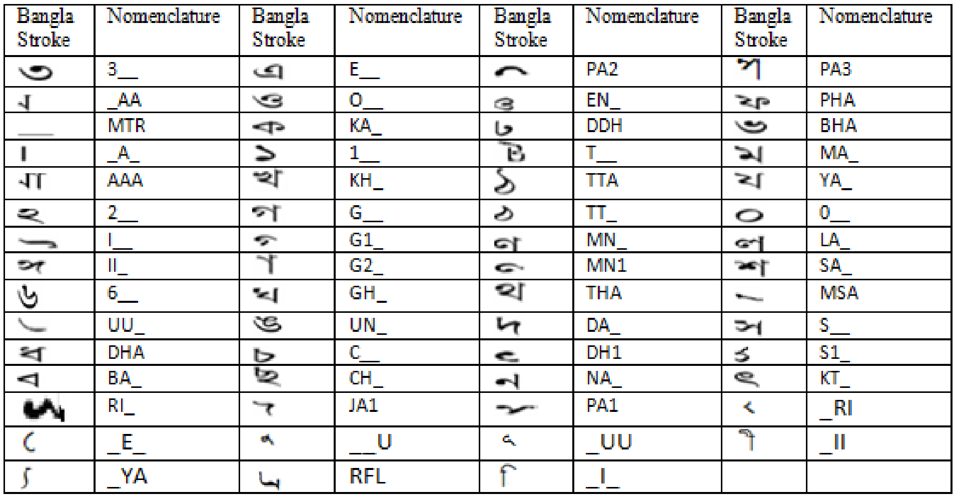


Figure 6.1.10: (a) Skewed handwritten Bangla word sample (b-c) skew detection and busy zone formation using Algorithm 17 (d) formation of sub-zones within busy zone (e) achieved segmentation points after applying modified DUD approach within sub-zones

6.1.2 Stroke-based feature extraction

After passing through the segmentation process, the obtained strokes are analyzed and grouped them into 59 classes [104] reflected in Fig. 6.1.11. Existence of some nearly similar shape structures has been observed during statistical analysis of the stroke database considered here. In order to differentiate such similar shaped patterns, some local information are estimated to produce discriminating features for the recognition purpose. Modified distance based feature discussed in Sen et al. [99, 101] and point based along with curvature based feature extraction technique for online Bangla handwritten stroke recognition have been used in the way mentioned in section 4.2.2.2, 4.2.2.3 and 4.2.2.4.



Bangla Stroke	Nomenclature	Bangla Stroke	Nomenclature	Bangla Stroke	Nomenclature	Bangla Stroke	Nomenclature
	3__		E__		PA2		PA3
	_AA		O__		EN_		PHA
	MTR		KA_		DDH		BHA
	_A_		1__		T__		MA_
	AAA		KH_		TTA		YA_
	2__		G__		TT_		0__
	I__		G1_		MN_		LA_
	II_		G2_		MN1		SA_
	6__		GH_		THA		MSA
	UU_		UN_		DA_		S__
	DHA		C__		DH1		S1_
	BA_		CH_		NA_		KT_
	RI_		JA1		PA1		_RI
	_E_		__U		_UU		_II
	_YA		RFL		_I_		

Figure 6.1.11: List of all stroke samples obtained after segmentation of online handwritten Bangla words

6.1.3 Experimental result of word segmentation and stroke recognition

In this current experiment, 6,500 online handwritten Bangla word samples have been used to verify the efficiency of the proposed segmentation algorithm. The benefits of this approach are 3-fold:

1) Usage of available online stroke information to make the segmentation process effective for multi-directional skewed word sample (as shown in Fig. 6.1.12 (a)), where constituent strokes are poorly aligned, and for samples like in Fig. 6.1.12 (b) where almost every character contains modifier in the lower part. Looking at the multi-directional skew in Fig. 6.1.12 (a) and structural pattern of Fig. 6.1.12 (b), this can be understood that busy zone formation for the whole word by following the approach mentioned in [19, 20] may be erroneous, and which in turn may result either over or under-segmentation. Whereas, in the proposed approach, main focus is on stroke, rather than the entire word. Hence, the issue of skewness become easier to deal with as stroke-level local skew is much lesser than the word-level global skew. Along with that, stroke categorization technique helps to identify only those strokes, which need to be segmented, having multiple connected components.

2) Proposed busy zone formation over target strokes avoids skew correction process as discussed earlier. Hence, this approach reduces execution time needed for the skew correction before entering into segmentation process.

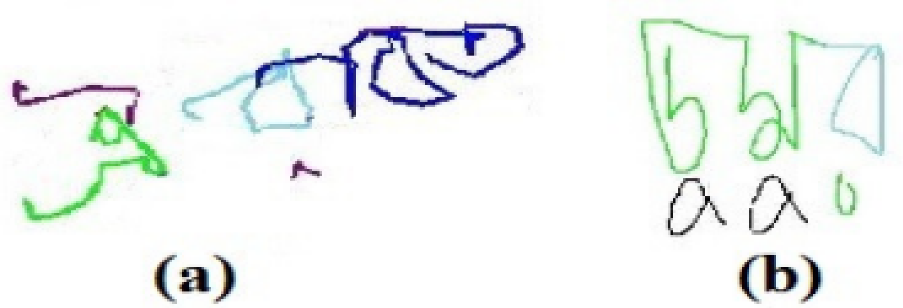


Figure 6.1.12: Example of words for which busy zone formation by computing horizontal histogram may mislead segmentation process. The reason for (a) is skew and for (b) every character has a symbol in its lower part

3) Division of busy zone into sub-zones followed by modified DUD technique, not only reduces the time complexity and difficulty of verifying different constraints

for candidate segmentation point selection, introduced in [19, 20], but also helps to handle different types of handwriting styles in an unconstrained domain.

The effectiveness of proposed segmentation technique is evaluated by its prediction ability to give a correct classification of events as an attack or a normal behaviour. According to the real nature of a given event and the prediction, four possible outcomes may be there. True negative (TN) and true positive (TP) correspond to correct operations observed. TN is a event that is actually normal and also successfully labelled as normal, event which is actually attack and is successfully labelled as attack is termed as TP. Similarly, false positive (FP) corresponds to normal event but classified as attack and false negative (FN) is actually attack event but incorrectly classified as normal event. Hence, in our case, TP, FP and FN reflect correctly segmented, over-segmented and under-segmented points respectively. In the proposed approach the value of TN is absent.

From the generated values of TP, FN and FP, precision ($P = TP/(TP+FP)$), recall ($R = TP/(TP+FN)$) and F-measure ($F = 2 * ((P * R)/(P + R))$) of the proposed system are computed. Table 6.1.1 describes the detail segmentation results of the proposed scheme.

Table 6.1.1: Detail segmentation results of online handwritten Bangla words

Observation Parameters	**Observed Outcome**
Ideal Segmentation points	3930
Correctly Segmented Points (TP)	3869 (98.45%)
Under-Segmented Points (FN)	61 (1.55%)
Over-Segmented Points (FP)	72 (1.83%)
P	0.98
R	0.98
F	0.98

Table 6.1.2 describes the detail recognition results of a total of 13,630 stroke samples when distance based features have been applied. Some well-known classifiers such as MLP, SVM, Simple Logistic, BayesNet and NaiveBayes have been

employed for stroke recognition purpose. From boldfaced entries in Table 6.1.2, it can be observed that SVM outperforms other classifiers by showing highest stroke recognition accuracy of 92.95%.

Table 6.1.2: Success rates of different classifiers for the recognition of strokes of the online Bangla words with distance based features

Stroke database size (#)	**Classifier**	**Recognition accuracy (in %)**
13630 (59 class)	MLP	92.65
	SVM	92.95
	Simple Logistic	92.4
	BayesNet	86.63
	NaiveBayes	83.5

As in the preliminary experimentation of stroke recognition, C-SVC type SVM shows the best result among other classifiers, so we have decided to test more with varying the parameters of SVM in order the enhance the performance of the same. Table 6.1.3 reports result observed here. Three kernel variants of SVM are applied in the experiment. Given vectors u, v the polynomial, Gaussian and linear kernels are defined through

$$k(u,v) = (\gamma * u^t * v + a)^d \tag{6.1}$$

$$k(u,v) = \exp\left(-\beta * \|u - v\|^2\right) \tag{6.2}$$

$$k(u,v) = u^t * v \tag{6.3}$$

where a, β, γ, d are real numbers and $d, \gamma > 0$. d is the degree of polynomial kernel. From Table 6.1.3 it can be seen that polynomial kernel of SVM gives best classification result when all the tuning parameters are set with default value (e.g.

the default value of d is 3).

Table 6.1.3: Stroke recognition results with C-SVC type SVM for three different kernels

Kernel variants of SVM	**Recognition accuracy (in %)**
Gaussian	92.95
Polynomial	**95.33**
Linear	94.95

This experiment also investigates the accuracy of the SVM by tuning the parameters *epsilon* and *gamma* for polynomial kernel. *epsilon* controls tolerance for termination criteria and *gamma* tunes the applied kernel function. The detailed information regarding recognition accuracy and time taken to build the model are mentioned in Table 6.1.4. Here, two situations are considered for tuning the parameters. In the first case, the value of *gamma* is fixed and increased the epsilon values gradually. Consequently, it has been observed that either stroke recognition accuracy decreases or model building time increases with increasing *epsilon* value. In the second case, the value of *epsilon* is kept fixed and gradually the value of *gamma* is increased. In this situation, model building time increases and recognition accuracy also gets decreased with the increase of *gamma* value. Gray cell of Table 6.1.4 highlights the situation where best recognition rate along with minimum model building time is observed (when the values of *epsilon* and *gamma* are set to 0.001 and 0 respectively).

The individual and combined effect of point based and curvature based feature extraction strategies for the stroke recognition also have been discussed here. An MLP based classifier and 5-fold cross-validation scheme are used as a stroke recognition model. Achieved recognition accuracies are 93.17% and 91.3% for point based and curvature based features respectively. The numbers of neurons in the hidden layer of the MLP for these two features are set to 107 and 62 respec-

Table 6.1.4: Stroke recognition results produced by SVM (with polynomial kernel) varying the *epsilon* and *gamma* values

SVM Type	Kernel type	epsilon	gamma	Recognition accuracy (in %)	Model building time (in sec)
C-SVC	Polynomial	0.001	0.0	95.33	23.56
		0.002	0.0	95.32	48.81
		0.003	0.0	95.32	50.45
		0.005	0.0	95.32	50.47
		0.001	0.1	95.01	65.75
		0.001	0.2	94.99	67.62
		0.001	0.3	94.96	69.14

tively. Table 6.1.5 reflects the performance measurements regarding features, their dimension and achieved classification accuracy.

Table 6.1.5: Measurements related to applied features, their dimension and achieved classification accuracy

Feature	Dimension	Accuracy (in %)
Point based Feature (1)	192	93.17
Curvature based Feature (2)	64	91.3
Combination of (1+2)	256	93.2

From Table 6.1.5, it can be noted that point based features perform better than curvature based features. It has also been observed the individual class level accuracy for both the procedures and noticed that in spite of slightly lower recognition performance, curvature based features become more effective for certain classes that are not recognized properly by point based features. To enhance the prediction capability of such classes, point based features are combined with curvature based features (early fusion). In effect, overall stroke recognition accuracy increases and it becomes 93.2% which is marginally higher than the performance of individual feature vectors but the purpose to increase the recognition accuracy of weaker classes are not fully met by this early fusion methodology.

In order to increase the overall stroke prediction ability, late fusion technique [74] is applied, i.e. after classification of the strokes by both the feature vectors (point based and curvature based) using MLP. In this scheme, MLP classifier is trained for each modality (m) ”point based” and ”curvature based”. Each modality is assigned an expertise trust score e_{sm} that is specific to each activity s (prediction choice). This score is defined as a normalized performance computed from the performance measure P_{sm}(observed probability/prediction percentage from classifier) obtained by the modality m for the activity s as defined in Eq. 6.4.

$$e_{sm} = \frac{p_{sm}}{\sum_m p_{sm}} \tag{6.4}$$

The final decision is computed as follows: for each observation, allocate a label to an activity that has the best trusted expert score of both the modalities. For each observation O_i , we have applied the argmax fusion operator as defined in Eq. 6.5:

$$s_i = argmax(e_{sipointbased}, e_{sicurvaturebased}) \tag{6.5}$$

The application of late fusion technique with max fusion operator, gives stroke prediction choice that has the highest trusted score between both the modalities. Hence, the resultant choice may be selected from either point based or curvature based modality. As a result, overall stroke recognition accuracy increases and it reaches to 95.4%.

Consider a sample word গাছ, which when goes through segmentation module generates four strokes গ(G___), ◌া(_A_), ছ(CH_) and matra - (MTR). The preferred naming conventions are mentioned within parentheses. These strokes first undergo recognition procedure by applying point based and curvature based features, then top-5 stroke recognition choices produced by each feature vector along with the effect of applying late fusion technique are recorded (see Fig.6.1.13). By observing the figure it can be stated that point based features recognize first three strokes correctly as top-most choice but the choice for the last stroke is not listed even within top-5 choices. On the other hand, the curvature based features fail to recognize the first stroke but choices for the remaining strokes are placed within top-5 list. When late fusion is applied on the classifier confidences, all the constituent strokes of the word sample are recognized within top-5 choices which in turn create an appropriate environment for the HMM to predict the word correctly.

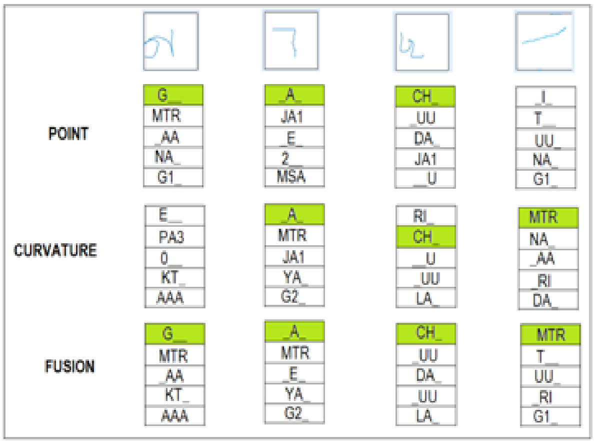


Figure 6.1.13: Top-5 stroke recognition choices for point based features, curvature based features and after application of late fusion technique

As the adopted late fusion technique becomes more effective than individual distance based procedure, hence in this work, the late fusion strategy is a used for stroke recognition.

6.2 Word recognition using HMM

No stroke recognition method can ensure that all the strokes of a certain word sample will be recognized as top choice when dealing with handwritten word samples. Hence, to counter this limitation, here we have attempted to re-construct the word from the recognized stroke samples by constructing a suitable model based on HMM which is fed with the top-5 stroke prediction choices (i.e. top-5 observation symbols). Fig. 6.2.1 portrays a word sample having 3 strokes with top-5 choices for each stroke.

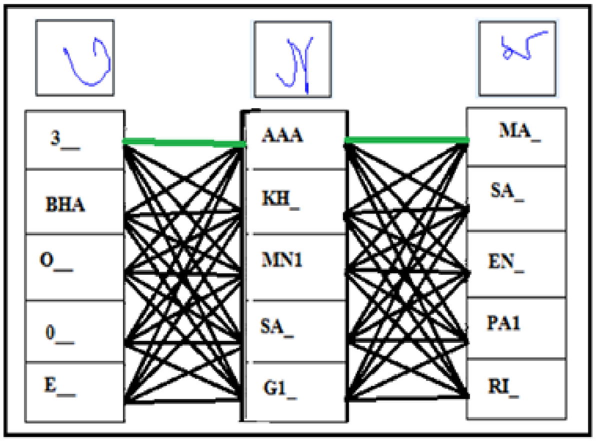


Figure 6.2.1: Word sample having 3 strokes with top-5 choices for each stroke

Fig. 6.2.1 highlights possible state sequences that may generate different observation sequences. It can be said that the actual state sequence (marked by green colour) is hidden. More specifically, the word construction problem can be defined as: given an observation sequence O and the complete parameter set of an HMM λ, what is the optimal state sequence $Q = Q_1Q_2....Q_T$ which maximizes $P(Q, O|\lambda)$. Clearly this is a decoding problem and can be efficiently solved by implementing Viterbi algorithm.

Let λ denotes the HMM by the triplet,{A, B, Π}, a complete parameter set of an HMM where,

A= {a_{ij}= P(s_j at t+1 | s_i at t)}: the state transition probabilities

B = {$b_i(v_k)$ = P(v_k at t | s_i at t}: the symbol observation probabilities and

Π = { π_i= P(s_i at t = 1)}: initial state probabilities

The decoding problem finds an optimal state sequence when given the observation sequence, $O = O_1O_2....O_T$, and the model parameter λ. It is assumed that the role of optimality criterion is to maximize $P(Q, O|\lambda)$, i.e., the joint probability of the state sequence, $Q = Q_1Q_2....Q_T$, and the observation sequence O given the model λ. The optimal state sequence is denoted by Q^*.

Such optimization problem can be solved by using well-known Viterbi algorithm which is based on the concept of dynamic programming. Let, $\delta_t(i)$ denotes the maximum probability of the optimal partial state sequence, $Q_1Q_2....Q_{t-1}$, with the state S_i at time t and observing the partial observation sequence, $O_1O_2....O_t$, given the model λ.

$$\delta_t(i) = max_{Q_1Q_2....Q_{t-1}} P(Q_1Q_2....Q_t = S_i, O_1O_2....O_t|\lambda) \tag{6.6}$$

$\delta_t(i)$ can be calculated recursively:

$$\delta_t(i) = \begin{cases} \pi_i b_{io_1} & t = 1, 1 \leq i \leq N \\ \delta_{t-1}(j) a_{ji} b_{io_t} & 2 \leq t \leq T, 1 \leq i \leq N \\ \underset{1 \leq i \leq N}{max} & \end{cases} \tag{6.7}$$

From the definition of $\delta_t(i)(y)$ it is clear that

$$P(Q^*, O|\lambda) = \underset{1 \leq i \leq N}{max} \delta_T(i) \tag{6.8}$$

Using Eq. 6.7 and Eq. 6.8, the joint probability of the optimal state sequence and the observation sequence can be computed when the model, P(Q^*,O$|\lambda$) is given. It is to be noted that, the memory usage by using this technique is very effective because only N forward variables($\delta_t(i)$) need to be stored at any time t. By keeping track of the argument i in both equations as P(Q,O$|\lambda$) is being maximized, the optimal state sequence can be recovered completely. When P(Q^*,O$|\lambda$) is a good approximation of $P(O|\lambda)$ Viterbi algorithm can be used for the evaluation problem.

6.3 Experimental result of Word Recognition by HMM

In order to demonstrate the proposed word recognition system, 6500 online handwritten Bangla word samples (50 different word samples for each 130 samples) are considered. After stroke recognition by adopting the late fusion technique, an HMM based recognition model has been followed for the construction of word samples. During evaluation of this model, three different cases have been noticed: (1) word samples where every stroke is recognized correctly as the top choice, (2) word samples where every stroke is recognized correctly within top-5 choices, and (3) prediction of at least one stroke of the word is not listed within top-5 choices.

Case 1:

Fig. 6.2.1 illustrates case 1 where the green line denotes the HMM predicted path carrying highest weight towards the construction of the target word sample. This path includes the sequence of top choices (having highest observation probability) of the constituent strokes. Now, depending on metric of the affinity between stroke pairs (transition probability P) in the associated corpus, correct path may be predicted as top choice or may be within top-5 choices. For example, if P(AAA/3__) and P(MA_/AAA) are both high then HMM predicts the correct choice at the top, but if say P(MA_/AAA) is much lesser than P(S__/AAA) and the observation probability of MA_ is marginally higher than S__, then HMM may predict the sequence 3__ AAA S__ (আস) as top choice, but that would be a wrong prediction.

Case 2:

Fig. 6.3.1 highlights the situation where all the strokes of the target word may not be recognized as the top choice but the correct choice lies within the top-5 choices (case 2). One such situation as depicted in Fig. 6.3.1 is where the metric of observation probability of the correct choice of stroke #3 is not high enough (the actual path sequence marked by a solid line). In that situation the metric of transition probability of stroke pairs helps to predict the state sequence correctly. Clearly in Fig. 6.3.1, if the transition probability P(S__/AAA) is much lower than P(MA_/AAA) and the observation probability of S__ gets marginally higher than

MA_, then HMM may find the correct path 3__ AAA MA__ (আম marked by solid line) otherwise, HMM reflects the sequence 3__ AAA S__ (আস) as top choice.

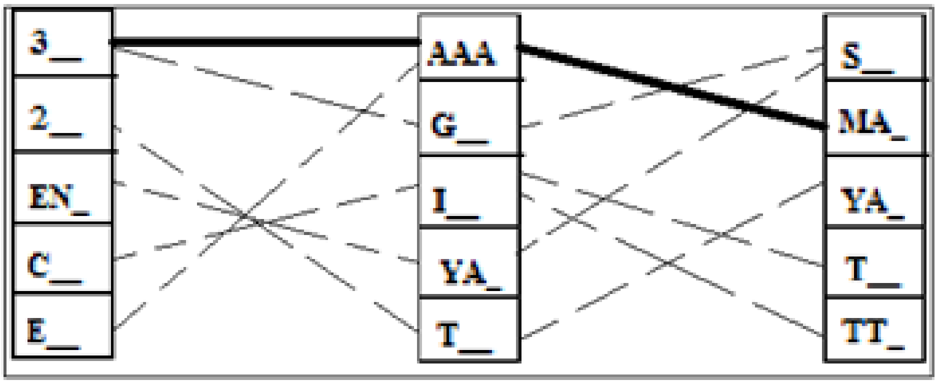
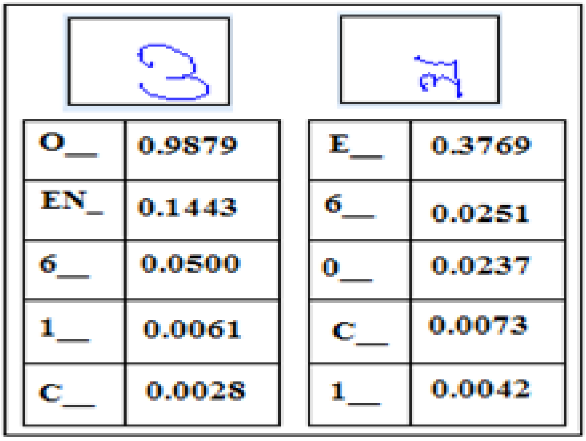


Figure 6.3.1: Example of a word sample where correct choice of strokes are within top-5 recognition choices

Case 3:

Fig. 6.3.2 depicts the scenario of case 3, where the correct choice for at least one stroke of the target word sample is not listed within top-5 choices. In this situation the actual state sequence is never reflected as the observation sequence and thus Viterbi algorithm fails to find the actual path and gives a wrong prediction choice.



O__	0.9879
EN_	0.1443
6__	0.0500
1__	0.0061
C__	0.0028

E__	0.3769
6__	0.0251
0__	0.0237
C__	0.0073
1__	0.0042

Figure 6.3.2: Example of a word sample where the correct choice of a stroke is not listed within the top-5 recognition choices

To prevent this unfavourable situation arising out due to the affinity between stroke pairs in the accounting corpus as mentioned in case (1) and case (2), we have used N-gram LM as a post-processing step to predict the correct state sequence obtained from the HMM prediction.

- N-gram Language Model

  Markov models are the class of probabilistic models which make the simplified assumption that that we can predict the probability of some future state without looking too far into the past. The joint probability of a sequence of strokes is decomposed using the chain rule of probability mentioned in Eqns. 6.9-6.10.

$$P(S_1^n) = P(S_1)P(S_2|S_1)P(S_3|S_1^2)....P(S|S_1^{n-1}) \tag{6.9}$$

$$= \prod_{k=1}^{n} = P(S_k|S_1^{k-1}) \tag{6.10}$$

The chain rule shows the link between computing the joint probability of a sequence and computing the conditional probability of a stroke given the previous stroke sequence. This chain rule helps predicting the correct option from the HMM predictions using corpus knowledge. According to the Markov assumption mentioned in Eq. 6.11, the probability of occurrence of a stroke depends only on the previous stroke and a bi-gram model is used to predict the conditional probability (also called transition probability) of the next stroke. We thus make the following approximation:

$$P(S_n|S_1^{n-1}) \approx P(S_n|S_{n-1}) \tag{6.11}$$

Hence, finally the model becomes as mentioned in Eq. 6.12.

$$P(S_1^n) = \prod_{k=1}^{n} P(S_n|S_{n-1}) \tag{6.12}$$

Consider the scenario described in Fig. 6.3.1. Even if the sequence (3__ AAA S__) may appear on the top by HMM due to higher transition probability of P(S__☒AAA)than P(MA_☒AAA) and higher observation probability of S__ than MA_, but the absence of P(S__|3__AAA) (stroke S__ having seen 3__ AAA) and the presence of P(MA_|3__ AAA) in the associated corpus make the chain rule of probability higher for the state sequence (3__ AAA MA_). Hence, N-gram LM helps find the actual state sequence. Table 6.3.1 highlights the detailed measurements observed for the calculation of word recognition accuracy. After analyzing the result it can be observed that 680 word samples are affected by stroke recognition module. Of these, HMM successfully recovers 104 samples (though some stroke choices are not in top) from the affected samples and exhibits a recognition accuracy of 89.53% which is comparably better than when used a rule based word recognition method considering only the top choice for the stroke samples (87.64%). Due to presence of the stroke sequences in the corpus, 42 more affected word samples that were not corrected by HMM, are now correctly predicted after passing through the LM. Thereby, amalgamation of HMM with LM yields overall 90.3% recognition accuracy when top-5 stroke recognition choices are considered, which enhances the overall word recognition performance by 0.77%.

Table 6.3.1: Detailed performance analysis of the proposed word recognition model

#Word sample	#samples having # of stroke error						#samples affected by stroke recognition error	Recognition accuracy by rule based approach	# samples corrected by HMM (recognition accuracy in %)	# samples corrected By LM (recognition accuracy in %)	Overall Accuracy
	1	2	3	4	5	6					
6500	415	274	74	31	8	2	804	87.64%	123(89.53%)	50(90.3%)	90.3%

6.4 Discussion

In this chapter, a novel stroke-based online Bangla word segmentation technique has been proposed and out of total tested words, 98.45% are segmented correctly by using this algorithm. Considering the word segmentation mechanisms proposed in [19, 20], it can be concluded that the technique proposed here is more effective than the works mentioned in [19, 20].

After stroke segmentation, three feature extraction procedures (distance based, point based and curvature based) have been used for the recognition of segmented strokes. These techniques have been applied over 13630 strokes (considering 6500 word samples) for recognition purpose. The achieved stroke recognition accuracies for distance based, point based and curvature based feature vectors are 95.33%, 93.17%, 91.3% respectively. When point based and curvature based features are combined, then accuracy gets increased and reaches up to 93.2%. Again as a post-processing, late fusion scheme has been applied after evaluation of point based and curvature based techniques as discussed in earlier section. Adopting late fusion mechanism, 95.4% stroke recognition accuracy has been achieved.

After stroke recognition, a rule based approach and an HMM model has been build in order to construct word samples from recognized strokes. The rule based word recognition technique produces 87.64% correct word recognition accuracy. On the other hand, HMM based model exhibits 89.53% word recognition accuracy considering top-5 stroke recognition choices. A LM based post-processing scheme has also been applied in order to rectify HMM generated outcome which helps to predict the correct word choice. After application of LM scheme, the overall word recognition accuracy has been increased to 90.3%.

7

Conclusion and Future Work

In this thesis, a system has been developed to recognize online handwritten Bangla characters and words. Due to unavailability of public standard character and word database, both character and word database are developed under this work. Various feature extraction techniques and classification schemes are clubbed for character recognition following the holistic and stroke-based approaches. Deep learning based procedure is also experimented for the comparison of the same. Segmentation of online handwritten Bangla words is successfully addressed and used as the first step of word recognition. Segmentation of words helps to find out the constituent strokes of word samples, which is one of the key contributions of the work and is detailed in section 6.1.1. Stroke recognition using various feature vectors are described in section 6.1.3. An HMM-based model is developed in order to construct the word sample from recognized strokes as reported in section 6.2. Finally, a LM scheme is adopted to rectify the predictions generated by HMM.

The databases developed under this work can be partitioned into three broad categories: character database, word database and stroke database. As strokes are the constituent parts of character and/or words, hence to preserve inherent variations of the strokes, they are extracted from characters and words rather than collecting them individually. The character database contains isolated online handwritten Bangla characters collected from various individual. For data collection a special form is designed. To develop word database, two different types of forms are designed. One type of form is used to collect 50 different pre-specified word samples and second one is used to collect the online handwritten text documents. In the present work, online handwritten Bangla character, word and stroke database of 10000, 6500 and 64164 (32534+13630) samples have been prepared.

In the present work, a word databases is extracted from unconstrained online handwritten Bangla documents and ground truth information for the same is prepared. For preparation of ground truth data, a semi-automatic interface is designed based on our developed modules on text line, word, stroke segmentation, and with adequate provision for manual correction in every phase. All detailed information like text line, word information etc. including stroke-name are stored accordingly.

Both holistic and stroke based recognition schemes have been applied for online isolated handwritten Bangla character recognition. In case of holistic character recognition, five type of features (distance based, hausdorff distance based, transition count, CG based circular with topological feature and few local features in terms of area, chord length, mass distribution) have been proposed. By analyzing the results of different feature extraction strategies, it has been noticed that highest recognition accuracy of 99.48% has been achieved when transition count feature is combined with CG based circular feature (global, local and circular quadrant mass distribution) and topological features on 10000 online handwritten Bangla basic characters by SVM classifier.

For stroke based character recognition, two different types of features are estimated from collected strokes namely distance based and ZPT feature extraction techniques. Applying only distance based features, it yields 89.39% correct stroke recognition by MLP classifier. When these features are combined with ZPT based features, the stroke recognition accuracy increases by 0.88% and reaches up to 91.27%. After stroke recognition, two methodologies are adopted for character construction from recognized strokes. The first one is a rule based procedure and the second describes a DFA based approach. In rule based procedure, different possible stroke order sequences for all the characters are mentioned as rule in the rule-base. To reduce the number of possible stroke-order combinations while forming the character, major stroke and matra are placed as the first and last symbol of the stroke order sequence.

In deep learning based approach, CNN has been employed for the recognition of the online handwritten Bangla basic characters. In CNN based method features need not be extracted manually rather they are extracted by selecting suitable filter size in convolution time followed by pooling scheme. The great advantage of adopting this technique over the methods described in the previous two sections is to cut-off the overhead of computing handcrafted features because they always do not ensure to generate good performance. The recognition outcome of 99.40% is also appreciable compare to the results obtained by the handcrafted features mentioned in the earlier sections.

In this thesis, a novel Bangla word segmentation technique based on stroke level busy zone formation procedure has been proposed. In this approach, a sub-zoning scheme within busy zone followed by a modified DUD concept within these sub-zones has been used to find valid segmentation points. This scheme avoids over and under-segmentation issues caused by either inherent writing pattern or due to writing style variations up to certain extent. The proposed segmentation approach has been tested on 6500 online handwritten Bangla word samplesand achieved satisfactory result of 98.45%.

In word recognition,the proposed segmentation technique is used to obtain constituent basic strokes from online handwritten Bangla word samples and then the next step deals with recognition of those segmented strokes. Here, at first the strokes are categorized into three lists based on their positional information. After that a procedure is proposed to form busy zone at stroke-level to handle multi-directional skewed words. Then a sub-zoning scheme within busy zone followed by a modified DUD concept within these sub zones has been used to find the segmentation points. By using this algorithm 98.45% words are segmented correctly. Considering the word segmentation mechanisms proposed in [19, 20], it can be concluded that the technique proposed here is more effective than the works mentioned in [19, 20].

After stroke segmentation, three feature extraction procedures (distance based, point based and curvature based) have been experimented for the recognition of segmented strokes. These techniques have been applied over 13630 strokes (collected from 6500 word samples) for recognition purpose. The achieved stroke recognition accuracy for distance based, point based and curvature based are 95.33%, 93.17%, 91.3% respectively. When point based features are combined with curvature based features, the accuracy has been increased up to 93.2%. As a post processing, late fusion scheme has been applied after evaluation of the point based and curvature based techniques as discussed in earlier section. Adopting late fusion mechanism, 95.4% stroke recognition accuracy has been achieved.

After stroke recognition, two approaches have been experimented in order to construct word samples from recognized strokes: a rule-based approach and an HMM based model. The rule-based word recognition technique produces 87.64% word recognition accuracy. On the other hand, HMM based model exhibits 89.53% word recognition accuracy considering top-5 stroke recognition choices. As a post-processing, LM based scheme is also applied to rectify HMM generated outcome that helps to predict correct word choice. After application of LM scheme, the overall word recognition accuracy becomes 90.3%.

Future scope:

In future, more handcrafted features will be investigated for both character recognition and stroke recognition. Further investigation with deep learning based approaches with variations in network structure, kernel etc. can also be experimented in order to enhance the recognition accuracy.

The work may also be extended to languages like Assamese, Manipuri etc. and for other Bramhi scripts like Devanagari, Gurumukhi etc. due to their similarity with Bangla script. The implemented features can also be tried on off-line Bangla or similar scripts. The work may be extended for development of online bio-metric based systems for writer identification/verification purposes. The present work not only can be applied for full-fledged Bangla handwriting recognition system, but it can also be modified to a light-weight prototype system for its possible use in portable and hand-held devices like mobile, PDAs etc. The present work can also be extended for the development of online handwritten script identification systems.

Hence, the impact of present work, not only limited to the development of online handwriting recognition of the Bangla language used by millions people in India, but it has immense potential to contribute in realizing the digital India drive of the Government of India and also to help the people getting the benefits of digital initiatives in their own mother tongue.

Few more handcrafted features can be investigated in future for both character and stroke recognition. More investigation with deep learning based approaches with variations in network structure, kernel etc. can also be experimented in order to enhance the recognition accuracy as well as to minimize the computation time.

The work may also be extended to languages like Assamese, Manipuri etc. that poses same characteristics as Bangla script and for other similar Brambhi scripts like Devanagari, Gurumukhi etc. due to their similarity with Bangla script. The developed features can also be investigated for off-line base systems also. The work may also be extended for online bio-metric systems for writer identification or verification systems. The present work not only can be extended for full-fledged Bangla handwriting development system, but also its light-weight prototype can be helpful to developed systems for portable and hand-held devices like mobile, PDAs etc.

The present work can also be extended for the development of online handwritten language/script identification systems. Hence, the impact of present work, not only contribute in the development of online handwriting recognition in Indian sub-continent, it has immense potentiality for contribution in digital India and also to help the people for getting the benefits of digital initiatives in our mother tongue.

Bibliography

[1] URL http://en.banglapedia.org/index.php?title=Bangla_Language.

[2] URL http://visitbangladesh.gov.bd/about-bangladesh/culture-religion-language/.

[3] URL https://www.britannica.com/topic/Bengali-language.

[4] URL https://www.ethnologue.com/country/bd/languages.

[5] URL https://einfon.com/nationalsymbols/national-languages-of-bangladesh/.

[6] URL http://www.viadelivers.com/bengali-language-facts/.

[7] The world factbook — central intelligence agency. URL www.cia.gov.

[8] URL https://www.ethnologue.com/language/ben.

[9] A. Acharyya, S. Rakshit, R. Sarkar, S. Basu, and M. Nasipuri. Handwritten word recognition using mlp based classifier: a holistic approach. *in International Journal of Computer Science Issues)*, 10(2 Part 2):422, 2013.

[10] A. Aggarwal and K. Singh. Handwritten gurumukhi character recognition. *in International Conference on Computer, Communication and Control*, pages 1–5, 2015.

[11] B. B. Ahmed, S. Naz, M. I. Razzak, S. F. Rashid, M. Z. Afzal, and T. M. Breuel. Evaluation of cursive and non-cursive scripts using recurrent neural networks. *in International Journal on Neural Computing and Applications*, 27:603–613, 2016.

[12] L. Koerich Alessandro and L. Yann. A hybrid large vocabulary word recognition system using neural networks with hidden markov models. *in IWFHR, http://citeseer. nj. nec. com/koerich02hybrid. html*, 2002.

[13] C. Bahlmann and H. Burkhardt. The writer independent online handwriting recognition system frog on hand and cluster generative statistical dynamic time warping. *in IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):299–310, 2004.

[14] A. Bandyopadhyay and B. Chakraborty. Development of online handwriting recognition system: a case study with handwritten bangla character. In *in World Congress on Nature & Biologically Inspired Computing*, pages 514–519. IEEE, 2009.

[15] S. Basu, C. Chaudhuri, M. Kundu, M. Nasipuri, and D. K. Basu. Segmentation of offline handwritten bengali script. *in arXiv preprint arXiv:1202.3046*, 2012.

[16] C. Y. Suen. M. Berthod and S. Mori. Automatic recognition of handprinted characters-the state of the art. *Proceedings of IEEE*, 68(4):469–487, 1980.

[17] M. Berthod and P. Maroy. Learning in syntactic recognition of symbols drawn on a graphic tablet. *in International Journal on Computer Graphics and Image Processing*, 9(2):166–182, 1979.

[18] A. Bharath and S. Madhvanath. Hidden markov models for online handwritten tamil word recognition. In *in International Conference on Document Analysis and Recognition*, pages 506–510. IEEE, 2007.

[19] N. Bhattacharya and U. Pal. Stroke segmentation and recognition from bangla online handwritten text. In *in International Conference on Frontiers in Handwriting Recognition*, pages 740–745. IEEE, 2012.

[20] N. Bhattacharya, U. Pal, and K. Roy. Individual character segmentation from single stroke of bangla online handwritten text. *In International Journal of Machine Intelligence*, 3(4):251–258, 2011.

[21] N. Bhattacharya, U. Pal, and P. P. Roy. Stroke-order normalization for online bangla handwriting recognition. In *in International Conference on Document Analysis and Recognition*, pages 206–211. IEEE, 2017.

[22] U. Bhattacharya, B. K. Gupta, and S. K. Parui. Direction code based features for recognition of online handwritten characters of bangla. In *in 9th International Conference on Document Analysis and Recognition*, volume 1, pages 58–62. IEEE, 2007.

[23] U. Bhattacharya, A. Nigam, YS. Rawat, and S. K. Parui. An analytic scheme for online handwritten bangla cursive word recognition. *in International Conference on Frontiers in Handwriting Recognition*, pages 320–325, 2008.

[24] T. K. Bhowmik, U. Roy, and S. K. Parui. Lexicon reduction technique for bangla handwritten word recognition. In *in 10th IAPR International Workshop on Document Analysis Systems*, pages 195–199. IEEE, 2012.

[25] M. Blumenstein and B. Verma. A new segmentation algorithm for handwritten word recognition. In *in International Joint Conference on Neural Networks*, volume 4, pages 2893–2898. IEEE, 1999.

[26] R. M. Bozinovic and S. N. Srihari. Off-line cursive script word recognition. *in IEEE Transactions on pattern analysis and machine intelligence*, 11(1):68–83, 1989.

[27] C. Y. Suen C. C. Tappert and T. Wakahara. The state of art in online handwriting recognition. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 12(8):787–807, 1980.

[28] S. Knerr C. Viard-Gaudin, P. M. Lallican. Recognition-directed recovering of temporal information from handwriting images. *in Pattern Recognition Letters*, 26(16):2537–2548, 2005.

[29] C. C. Chang and C. J. Lin. Libsvm: a library for support vector machines. *in ACM transactions on intelligent systems and technology*, 2(3):27, 2011.

[30] J. H. Chiang. A hybrid neural network model in handwritten word recognition. *in Neural Networks*, 11(2):337–346, 1998.

[31] K. Chowdhury, L. Alam, S. Sarmin, S. Arefin, and M. M. Hoque. A fuzzy features based online handwritten bangla word recognition framework. In *in International Conference on Computer and Information Technology*, pages 21–23. IEEE, 2015.

[32] S. Chowdhury, U. Garain, and T. Chattopadhyay. A weighted finite-state transducer (wfst)-based language model for online indic script handwriting recognition. In *in International Conference on Document Analysis and Recognition*, pages 599–602. IEEE, 2011.

[33] S. D. Connell, R. M. K. Sinha, and A. K. Jain. Recognition of unconstrained online devanagari characters. *in* $15^{th} internationalconferenceonPatternRecognition, pages368 - -371, 2000$.

[34] A. Rosenfeld D. S. Doermann. Recovery of temporal information from static images of handwriting. *in International Journal of Computer Vision*, 15 (1-2):143–164, 1995.

[35] N. Das, K. Acharya, R. Sarkar, S. Basu, M. Kundu, and M. Nasipuri. A benchmark image database of isolated bangla handwritten compound characters. *in International Journal on Document Analysis and Recognition*, 17(4): 413–431, 2014.

[36] T. Dimond. Devices for reading handwritten characters. *in Proceedings of the Eastern Joint Computer Conference*, pages 232–237, 1957.

[37] J. Du, J. Zhai, and J. Hu. Writer adaptation via deeply learned features for online chinese handwriting recognition. *in International Journal on Document Analysis and Recognition*, 20:69–78, 2017.

[38] A. El-Sawy, H. EL-Bakry, and M. Loey. Cnn for handwritten arabic digits recognition based on lenet-5. *in International Conference on Advanced Intelligent Systems and Informatics*, pages 566–575, 2016.

[39] M. Farha, G. Srinivasa, A. J. Ashwini, and K. Hemant. Online handwritten character recognition. *in International journal of computer science*, pages 30–36, 2013.

[40] G. A. Fink, S. Vajda, U. Bhattacharya, S. K. Parui, and B. B. Chaudhuri. Online bangla word recognition using sub-stroke level features and hidden markov models. In *in International Conference on Frontiers in Handwriting Recognition*, pages 393–398. IEEE, 2010.

[41] V. Frinken, N. Bhattacharya, and U. Pal. Design of unsupervised feature extraction system for on-line handwriting recognition. In *in International Workshop on Document Analysis and Systems*, pages 355–359. IEEE, 2014.

[42] A. Fu, C. Dong, and L. Wang. An experimental study on stability and generalization of extreme learning machines. *in International Journal of Machine Learning and Cybernetics*, 6(1):129–135, 2015.

[43] L. Cordella A. Marcelli G. Boccignone, A. Chianese. Recovering dynamic information from static handwriting. *in Journal of Pattern Recognition*, 26 (3):409–418, 1993.

[44] P. Gader, M. Whalen, M. Ganzberger, and D. Hepp. Handprinted word recognition on a nist data set. *in Machine Vision and Applications*, 8(1): 31–40, 1995.

[45] U. Garain, B. B. Chaudhuri, and T. T. Pal. Online handwritten indian script recognition: a human motor function based framework. In *in 16th International Conference on Pattern Recognition*, volume 3, pages 164–167. IEEE, 2002.

[46] R. Ghosh. Segmentation of online bangla handwritten word. In *in International Conference on Advance Computing*, pages 658–663. IEEE, 2009.

[47] R. Ghosh and P. P. Roy. A novel feature extraction approach for online bengali and devanagari character recognition. In *in 2nd International Conference on Signal Processing and Integrated Networks*, pages 483–488. IEEE, 2015.

[48] G. Groner. Real-time recognition of handprinted text. *in Proceedings of in FJCC*, pages 591–601, 1966.

[49] DJ. Grover. Graphics tablets—a review. *in International Journal on Displays*, 1(2):83–93, 1979.

[50] C. Halder. *Writer Identification and Verification on Bangla Handwriting*. PhD thesis, West Bengal State University, 2018.

[51] K. Han and I. K. Sethi. Off-line cursive handwriting segmentation. In *in International Conference on Document Analysis and Recognition*, page 894. IEEE, 1995.

[52] L. D. Harmon. Automatic recognition of print and script. *Proceedings of IEEE*, 60(10):1165–1176, 1965.

[53] J. J. Hull. A database for handwritten text recognition research. *in IEEE Transactions on pattern analysis and machine intelligence*, 16(5):550–554, 1994.

[54] K. Janocha and W. M. Czarnecki. On loss functions for deep neural networks in classification. *in arXiv preprint arXiv:1702.05659*, 2017.

[55] H Jeffrey. Bengalis - facts and details. URL factsanddetails.com.

[56] J. M. Jeong, S. J. Choi, and C. Kim. Correcting method of numerical errors of soft-max and cross entropy according to cnn's output value. *in International Journal on Neural Computing and Application*, 1(1):15–20, 2017.

[57] N. Joshi, G. Sita, A.G. Ramakrishnan, and V. Deepu. Machine recognition of online handwritten devanagari characters. *in Proceedings of International Conference on Document Analysis and Recognition*, pages 1156–1160, 2005.

[58] J. Kim and CC. Tappert. Handwriting recognition accuracy versus tablet resolution and sampling rate. In *in Proceeedings of* $7^{th} InternationalConferenceonPatternRecognition, pages917 - -918, 1984$.

[59] S. Kubatur, M. Sid-Ahmed, and M. Ahmadi. A neural network approach to online devanagari handwritten character recognition. *in International conference on High Performance Computing and Simulation*, 2012.

[60] A. Kumar and S. Bhattacharya. Online devanagari isolated character recognition for the iphone using hidden markov models. *in International Conference on Students' Technology Symposium*, pages 300–304, 2010.

[61] K. Kumara, R. Agrawal, and C. Bhattacharyya. A large margin approach for writer independent online handwriting classification. *in International journal of Pattern Recognition Letters*, 29:933–937, 2008.

[62] V. L. Lajish and S. K. Kopparapu. Online handwritten devanagari stroke recognition using extended directional features. In *in* $8^{th} InternationalConferenceonSignalProcessingandCommunicationSystems, pages1 - -5.IEEE, 2014$.

[63] N. Lindgren. Machine recognition of human language, part iii - cursive script recognition. *Proceedings of IEEE*, 2(5):104–116, 1965.

[64] D. Mohammad M. F. Zafar and M. M. Anwar. Recognition of online isolated handwritten characters by back propagation neural nets using sub-character primitive features. *in* $9^{th} inInternationalConferenceonINMIC, pages157 - -162, 2006$.

[65] N. Gupta M. Gupta and R. Agrawal. Recognition of online handwritten gurmukhi strokes using support vector machine. *in Proceedings of* $7^{th} InternationalConferenceonBio - InspiredComputing : Theoriesandapplication, pages495 - -506, 2012$.

[66] J. Mantas. An overview of character recognition methodologies. *Pattern Recognition*, 19(6):425–430, 1986.

[67] Sk. Mohiuddin, U. Bhattacharya, and S. K. Parui. Unconstrained bangla online handwriting recognition based on mlp and svm. In *in Proceedings of the 2011 Joint Workshop on Multilingual OCR and Analytics for Noisy Unstructured Text Data*, page 16. ACM, 2011.

[68] T. Mondal, U. Bhattacharya, S. K. Parui, K Das, and D. Mandalapu. Online handwriting recognition of indian scripts-the first benchmark. In *in* 12^{th} *International Conference on Frontiers in Handwriting Recognition*, pages 200–205. IEEE, 2010.

[69] P. S. Mukherjee, U. Bhattacharya, S. K. Parui, and B. Chakraborty. A hybrid model for end to end online handwriting recognition. In *in International Conference on Document Analysis and Recognition*, pages 658–663. IEEE, 2017.

[70] F. Yarman-Vural N. Arica. An overview of character recognition focused on off-line handwriting. *in IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 31(2):216–233, 2001.

[71] A. Oland, A. Bansal, R. B. Dannenberg, and B. Raj. Be careful what you backpropagate: A case for linear output activations & gradient boosting. *in arXiv preprint arXiv:1707.04199*, 2017.

[72] F. Tortorella M. Vento P. Foggia, C. Sansone. Combining statistical and structural approaches for handwritten character description. *in Journal of Image and Vision Computing*, 17(9):701–711, 1999.

[73] S. K. Parui, K. Guin, U. Bhattacharya, and B. B. Chaudhuri. Online handwritten bangla character recognition using hmm. In *in 19th International Conference on Pattern Recognition*, pages 1–4. Citeseer, 2008.

[74] J. Pinquier, S. Karaman, L. Letoupin, P. Guyot, R. Mégret, J. Benois-Pineau, Y. Gaëstel, and J. F. Dartigues. Strategies for multiple feature fusion with hierarchical hmm: application to activity recognition from wearable audiovisual sensors. In *in International Conference on Pattern Recognition (ICPR)*, pages 3192–3195. IEEE, 2012.

[75] T. Plötz and G. A. Fink. Markov models for offline handwriting recognition: a survey. *in International Journal on Document Analysis and Recognition*, 12 (4):269, 2009.

[76] J. D. Prusa and T. M. Khoshgoftaar. Improving deep neural network design with new text data representations. *in Journal of Big Data*, 4(1):7, 2017.

[77] J. Lyall R. H. Davis. Recognition of handwritten characters a review. *Image Vision Computing*, 4(4):208–218, 1986.

[78] S. Srihari R. Plamondon. Online and off-line handwriting recognition: a comprehensive survey. *in IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):63–84, 2000.

[79] Md M. Rahman, A. MAH, S. Islam, P. Chandra Shill, and MM H. Rahmanand others. Bangla handwritten character recognition using convolutional neural network. *in International Journal of Image, Graphics and Signal Processing*, 7(8):42–49, 2015.

[80] A. Ray, S. Rajeswar, and S. Chaudhury. Text recognition using deep blstm networks. In *in 8th international conference on Advances in pattern recognition*, pages 1–6. IEEE, 2015.

[81] K. Roy, N. Sharma, T. Pal, and U. Pal. Online bangla handwriting recognition system. In *Advances In Pattern Recognition*, pages 117–122. World Scientific, 2007.

[82] K. Roy, A. Bandhopadhyay, and R. Mondal. Stroke-database design for online handwriting recognition in bangla. *in International Journal of Modern Engineering Research*, 2(4):2534–2540, 2012.

[83] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical report, in California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[84] V. Kumar S. Agarwal. Online character recognition. *in Proceedings of the* 3^{rd}*InternationalConferenceonInformationTechnologyandApplications*, *pages*698— —703, 2005.

[85] M. K. Sachan, G. Lehal Singh, and V. K. Jain. A novel method to segment online gurmukhi script. *in International Conference on Information Systems for Indian Languages*, pages 1–8, 2011.

[86] M. K. Sachan, G. Lehal Singh, and V. K. Jain. A system for online gurmukhi script recognition. *in International Conference on Information Systems for Indian Languages*, pages 299–300, 2011.

[87] K. C. Santosh and L. Wendlingc. Character recognition based on nonlinear multi-projection profiles measure. *in International Journal of Frontiers of Computer Science*, 9:678–690, 2015.

[88] K.C. Santosh and J. Iwata. Stroke-based cursive character recognition. *in Xiaoqing Ding, INTECH*, 2012.

[89] K.C. Santosh and C. Nattee. Stroke number and order free handwriting recognition for nepali. *in Q. Yang G. I. Webb (eds), Proceedings of the Pacific Rim International Conferences on Artificial Intelligence*, 4099:990–994, 2006.

[90] K.C. Santosh and C. Nattee. Structural approach on writer independent nepalese natural handwriting recognition. *in Proceedings of the International Conference on Cybernetics and Intelligent Systems*, pages 1–6, 2006.

[91] K.C. Santosh and C. Nattee. Template-based nepali natural handwritten alphanumeric character recognition. *in International Journal of Science and Technology*, 12(1):20–30, 2007.

[92] K.C. Santosh, C. Nattee, and B. Lamiroy. Spatial similarity based stroke number and order free clustering. *in International Journal of Science and Technology*, pages 652–657, 2010.

[93] K.C. Santosh, C. Nattee, and B. Lamiroy. Relative positioning of stroke-based clustering: a new approach to online handwritten devanagari character recognition. *in International Journal of Image Graphics*, 12(2), 2012.

[94] R. Sarkar. *Development of Line Extraction and Word Segmentation Techniques for unconstrained Handwritten Bangla Documents, 2012*. PhD thesis, Jadavpur University, 2012.

[95] R. Sarkar, N. Das, S. Basu, M. Kundu, M. Nasipuri, and D. K. Basu. A two-stage approach for segmentation of handwritten bangla word images. In *in Proceedings of International Conference on Frontiers in Handwriting Recognitions*, pages 403–408. Citeseer, 2008.

[96] K. M. Sayre. Machine recognition of handwritten words: A project report. *in International Journal on Pattern recognition*, 5(3):213–228, 1973.

[97] I. Scholey. Digitizer technology comparison: inductive and resistive technology. *in Pen Comput. Mag.*, 2006.

[98] S. Sen, M. Mitra, S. Chowdhury, R. Sarkar, and K. Roy. Quad-tree based image segmentation and feature extraction to recognize online handwritten bangla characters. In *in IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, pages 246–256. Springer, 2016.

[99] S. Sen, R. Sarkar, and K. Roy. A simple and effective technique for online handwritten bangla character recognition. In *in Proceedings of the 4th International Conference on Frontiers in Intelligent Computing: Theory and Applications*, pages 201–209. Springer, 2016.

[100] S. Sen, A. Bhattacharyya, A. Das, R. Sarkar, and K. Roy. Design of novel feature vector for recognition of online handwritten bangla basic characters. In *in Proceedings of the First International Conference on Intelligent Computing and Communication*, pages 485–494. Springer, 2017.

[101] S. Sen, R. Sarkar, and K. Roy. An approach to stroke-based online handwritten bangla character recognition. In *in Advanced Computing and Systems for Security*, pages 153–163. Springer, 2017.

[102] S. Sen, R. Sarkar, K. Roy, and N. Hori. Recognize online handwritten bangla characters using hausdorff distance-based feature. In *in Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications*, pages 541–549. Springer, 2017.

[103] S. Sen, A. Bhattacharyya, P. K. Singh, R. Sarkar, K. Roy, and D. Doermann. Application of structural and topological features to recognize online handwritten bangla characters. *in ACM Transactions on Asian and Low-Resource Language Information Processing*, 17(3):20, 2018.

[104] S. Sen, S. Chowdhury, M. Mitra, F. Schwenker, R. Sarkar, and K. Roy. A novel segmentation technique for online handwritten bangla words. *in International Journal on Pattern Recognition Letters*, 2018.

[105] S. Sen, D. Shaoo, M. Mitra, R. Sarkar, and K. Roy. Dfa-based online bangla character recognition. In *in Information Technology and Applied Mathematics*, pages 175–183. Springer, 2018.

[106] S. Sen, D. Shaoo, S. Paul, R. Sarkar, and K. Roy. Online handwritten bangla character recognition using cnn: A deep learning approach. In *in Intelligent Engineering Informatics*, pages 413–420. Springer, 2018.

[107] S. P. Sen, A. Bhattacharyya, M. Mitra, R. Sarkar, and K. Roy. Online bangla handwritten word recognition using hmm and language model. *in International Journal on Neural Computing and Application*, communicated.

[108] A. Sharma, R. Kumar, and R. K. Sharma. Online handwritten gurmukhi character recognition using elastic matching. *in Congress on Image and Signal Processing*, pages 391–396, 2008.

[109] A. Sharma, R. Kumar, and R. K. Sharma. Hmm-based online handwritten gurmukhi character recognition. *in International Journal on Machine Graphics Vision*, 19(4):439–449, 2010.

[110] A. Shivram, B. Zhu, S. Setlur, M. Nakagawa, and V. Govindaraju. Segmentation based online word recognition: A conditional random field driven beam search strategy. In *in International Conference on Document Analysis and Recognition*, pages 852–856. IEEE, 2013.

[111] K. R. Sivaramakrishnan and C. Bhattacharyya. Time series classification for online tamil handwritten character recognition – a kernel based approach. *in ICONIP*, pages 800–805, 2004.

[112] S. N. Srihari. Recognition of handwritten and machine-printed text for postal address interpretation. *in International Journal on Pattern recognition letters*, 14(4):291–302, 1993.

[113] S. N. Srihari. Recognition of handwritten and machine-printed text for postal address interpretation. *in International Journal on Pattern recognition letters*, 14(4):291–302, 1993.

[114] A. Srimany, S. Dutta Chowdhuri, U. Bhattacharya, and S. K. Parui. Holistic recognition of online handwritten words based on an ensemble of svm classifiers. In *in 11th IAPR International Workshop on Document Analysis Systems*, pages 86–90. IEEE, 2014.

[115] C. Y. Suen. Character recognition by computer and applications. *in Handbook of Pattern Recognition and Image Processing*, pages 569–586, 1986.

[116] M. Sumner, E. Frank, and M. Hall. Speeding up logistic model tree induction. In *in European Conference on Principles of Data Mining and Knowledge Discovery*, pages 675–683. Springer, 2005.

[117] H. Swethalakshmi, A. Jayaraman, V. S. Chakravarthy, and C. C. Sekhar. Online handwritten character recognition for devanagari and telugu scripts using support vector machines. *in Proceedings of International Workshop on Frontiers in Handwriting Recognition*, pages 367–372, 2006.

[118] H. Swethalakshmi, C. C. Sekhar, and V. S. Chakravarthy. Spatiostructural features for recognition of online handwritten characters in devanagari and tamil scripts. *in Proceedings of International Conference on Artificial Neural Networks*, pages 230–239, 2007.

[119] N. Tagougui, M. Kherallah, and A. M. Alimi. Online arabic handwriting recognition: a survey. *in International Journal on Document Analysis and Recognition*, 16(3):209–226, 2013.

[120] X. Tang, Y. Ding, and K. Hao. A novel method based on line-segment visualizations for hyper-parameter optimization in deep networks. *in International Journal of Pattern Recognition and Artificial Intelligence*, 32(3):1–15, 2017.

[121] A. Tripathi, S. S. Paul, and V. K. Pandey. Standardisation of stroke order for online isolated devanagari character recognition for iphone. *in International conference on Technology Enhanced Education*, pages 1–5, 2012.

[122] G. Del Vescovo and A. Rizzi. Online handwriting recognition by the symbolic histograms approach. In *in International Conference on Granular Computing*, pages 686–686. IEEE, 2007.

[123] C. Wu, W. Fan, Y. He, J. Sun, and S. Naoi. Handwritten character recognition by alternately trained relaxation convolutional neural network. In *in 14th International Conference on Frontiers in Handwriting Recognition*, pages 291–296. IEEE, 2014.

[124] Q. Wu, Z. Gui, S. Li, and J. Ou. Directly connected convolutional neural networks. *in International Journal of Pattern Recognition and Artificial Intelligence*, 32(5):1–17, 2017.

[125] M. Yasuhara Y. Qiao, M. Nishiara. A framework toward restoration of writing order from single-stroked handwriting image. *in IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1724–1737, 2006.

[126] W. Yang, L. Jin, D. Tao, Z. Xie, and Z. Feng. Dropsample: A new training method to enhance deep convolutional neural networks for large-scale unconstrained handwritten chinese character recognition. *in International Journal of Pattern Recognition*, 58:190–203, 2016.

[127] B. Yanikoglu and P. A. Sandon. Segmentation of off-line cursive handwriting using linear programming. *in International Journal on Pattern recognition*, 31(12):1825–1833, 1998.

[128] B. Yanikoglu and P. A. Sandon. Segmentation of off-line cursive handwriting using hear programming. *in International Journal on Pattern recognition*, 31(12):1825–1833, 1998.

[129] B. A. Yanikoglu and P. A. Sandon. Off-line cursive handwriting recognition using style parameters. In *in Department of Mathematics and Computer Science, Dartmouth College*. Citeseer, 1993.

[130] W. Zhao, J. F. Liu, and X. L. Tang. Online handwritten english word recognition based on cascade connection of character hmms. In *in International Conference on Machine Learning and Cybernetics*, volume 4, pages 1758–1761. IEEE, 2002.