Recognition of Online Handwritten Devanagari Numerals using Support Vector Machine

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering
in
Computer Science and Engineering

Submitted By
Deepika Wadhwa
(Roll No. 801032007)

Under the supervision of:
Mr. Karun Verma
Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT
THAPAR UNIVERSITY
PATIALA – 147004

June 2012

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, **"Recognition of Online Handwritten Devanagari Numerals using Support Vector Machine"**, in partial fulfillment of the requirements for the award of degree of Master of Engineering in Computer Science and Engineering submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Mr. Karun Verma and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(Deepika Wadhwa)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Mr. Karun Verma)
Assistant Professor
Computer Science and Engineering Department
Thapar University
Patiala

Countersigned by

(Dr. Maninder Singh)
Associate Professor and Head
Computer Science and Engineering Department
Thapar University
Patiala

(Dr. S. K. Mohapatra)
Dean (Academic Affairs)
Thapar University
Patiala

ACKNOWLEDGMENT

Nothing big can be achieved without the guidance, assistance and co-operation of people and one must be indebted to them by words and deeds. A coordinated effort of others bridges the gap between where you are and where you want to be.

I would like to express my deep and sincere gratitude to my supervisor, Mr. Karun Verma, Assistant Professor, Computer Science and Engineering Department. His wide knowledge and logical way of thinking have been of great value for me. He was always there with his competent guidance and valuable guidance throughout the pursuance of this research project. His valuable suggestions always led my work to be in a progressive mode.

I owe my sincere thanks to Dr. Maninder Singh, Associate Professor, Head of Computer Science and Engineering Department for providing keen interest, unfailing support, inspiration and the necessary research facilities. I am also thankful to all the faculty and staff members of the Computer Science and Engineering Department for their full co-operation and help. I also extend my thanks to the writers who helped me collect the samples of handwriting for my work.

Above all, I thank my parents and my friends for their constant support and encouragement throughout my life. Most importantly, I pay my reverence to the almighty God for giving me inner peace and strength and not letting me down at the time of crisis.

Deepika Wadhwa

(801032007)

ABSTRACT

Handwriting has continued to persist as a mean of communication and recording information in day-to-day life even with the invention of new technologies. Natural handwriting is one of the easiest ways of information exchange between a human and a computer. Handwriting recognition has attracted many researchers across the world since many years. Recognition of online handwritten Devanagari numerals is a goal of many research efforts in the pattern recognition field.

The main goal of the work presented in this dissertation is the recognition of online handwritten Devanagari numerals using support vector machine. In the data collection phase, co-ordinate points of the input handwritten numeral are collected as the numeral is written; various algorithms for pre-processing are applied for normalizing, resampling and interpolating missing points, smoothing and slant correction. Two low-level features i.e. direction angle and curvature are extracted from the pre processed data. These features along with the x and y coordinates of the input handwritten character are stored in a .csv file and fed directly to the recognition phase. Recognition is done using four kernel functions of SVM by partitioning the data into different schemes. The recognition accuracies are obtained on different schemes of data using the four kernel functions of SVM for each scheme.

TABLE OF CONTENTS

CERTIFICATE **i**

ACKNOWLEDGEMENT **ii**

ABSTRACT **iii**

TABLE OF CONTENTS **iv**

LIST OF FIGURES **vii**

LIST OF TABLES **ix**

CHAPTER 1: INTRODUCTION **1-14**

1.1 Areas of handwriting Recognition 2

1.1.1 Offline Handwriting Recognition 3

1.1.2 Online Handwriting Recognition 3

1.1.3 Online Recognition vs. Offline Recognition 4

1.2 Issues in Online Handwriting Recognition System 5

1.2.1 Variations in Handwriting Styles 5

1.2.2 Personal or background Factors 8

1.2.3 Situational Factors 8

1.2.4 Material Factors 8

1.2.5 Writer Dependent vs. writer Independent Handwriting Recognition 8

1.2.6 Constrained and Unconstrained Handwriting 9

1.2.7 Limited Resources in Small Devices 10

1.2.8 Stroke Number and Order Variation 11

1.2.9 Similarity in Shape of some Characters 11

1.3 Overview of Devanagari Numerals 12

CHAPTER 2: LITERATURE REVIEW **15-34**

2.1 Data Collection 16

2.2 Pre Processing 18

2.3 Feature Extraction 20

2.4 Segmentation 22

2.5 Recognition 25

2.5.1 Template Matching 25

2.5.2 Syntactic or Structural Matching 26

2.5.3 Neural Network Methods 26

2.5.4 Statistical Classification 27

2.6 Post Processing 30

2.7 Previous Recognition Results in Online Handwriting Recognition 31

2.8 Problem Statement 34

CHAPTER 3: DATA COLLECTION, PRE PROCESSING AND FEATURE EXTRACTION **35-44**

3.1 Data Collection Phase 35

3.2 Pre Processing phase 37

3.2.1 Size Normalization and Centering 37

3.2.2 Interpolating Missing points 38

3.2.3 Resampling 39

3.2.4 Smoothing 40

3.2.5 Slant Correction 40

3.3 Feature Extraction 42

3.3.1 Direction Angles 42

3.3.2 Curvature 43

CHAPTER 4: RECOGNITION PROCESS **45-55**

4.1 Introduction to SVM 45

4.2 Recognition of Numerals 48

4.2.1 SVM Kernels 49

4.3 Results and Discussions 50

CHAPTER 5: CONCLUSIONS AND FUTURE SCOPE **56**

REFERENCES **57-60**

LIST OF PUBLICATIONS **61**

LIST OF FIGURES

Figure 1.1: Areas of handwriting recognition 2

Figure 1.2: A tablet, input sampling and communication to the computer 3

Figure 1.3: Variations in numerals written by five writers 6

Figure 1.4: Variations in numerals written by the same writer 7

Figure 1.5: Boxed discrete handwriting 10

Figure 1.6: Different styles of writing in Devanagari 10

Figure 1.7: Similarities in characters 11

Figure 1.8: Different forms of Devanagari characters 12

Figure 1.9: Devanagari numerals and their names 14

Figure 2.1: Phases of online handwriting recognition system 15

Figure 2.2: Commonly used hardware devices for capturing handwriting 17

Figure 2.3: Common steps in pre processing phase 19

Figure 2.4: Direction values used in Freeman's chain code 21

Figure 2.5(a): Shape of a stroke 23

Figure 2.5(b-f): Five substrokes (respectively W, S, E, N and W) obtained from Figure 2.5(a) 23

Figure 2.6: Examples of top, baseline and bottom strokes 24

Figure 2.7: Feature extraction of numeral '3' using box method 25

Figure 3.1: Points of pen movement collected while writing 36

Figure 3.2: Writing area for writing the numerals 36

Figure 3.3: Handwritten character after normalization 38

Figure 3.4: Handwritten character after Bezier interpolation 39

Figure 3.5: Handwritten character after resampling 40

Figure 3.6: Pre processing steps 41

Figure 3.7: Data collected in a .csv file 44

Figure 4.1: Linear classification using decision planes 45

Figure 4.2: Complex classification 46

Figure 4.3: Idea behind SVM 47

Figure 4.4: Concept of support vectors 47

Figure 4.5: Recognition of numerals with SCHEME 1 50

Figure 4.6: Recognition of numerals with SCHEME 2 51

Figure 4.7: Recognition of numerals with SCHEME 3 52

Figure 4.8: Recognition of numerals with SCHEME 4 53

Figure 4.9: Recognition of numerals with SCHEME 5 54

Figure 4.10: Recognition of numerals with SCHEME 6 55

LIST OF TABLES

Table 2.1: Selected recognition results from the literature of online handwriting recognition systems 32

Table 3.1: Range of direction angles according to 8-directional code 42

Table 4.1: Recognition accuracies with SCHEME 1 50

Table 4.2: Recognition accuracies with SCHEME 2 51

Table 4.3: Recognition accuracies with SCHEME 3 52

Table 4.4: Recognition accuracies with SCHEME 4 53

Table 4.5: Recognition accuracies with SCHEME 5 54

Table 4.6: Recognition accuracies with SCHEME 6 55

CHAPTER 1
INTRODUCTION

Today's computers are very powerful and capable of processing large amounts of data in very less time. The amount of information that can be processed and stored by computers is increasing at a tremendous rate. With this increase in rate, the ease with which the information can be exchanged between a computer and a user is causing serious problems. In order to fully utilize the tremendous processing capability of the computer, a user interface is needed which should not only be efficient but also natural to the user.

The user interface must be natural so that user does not find it difficult to communicate with the computer and efficient in order to speed up the communication. The primary modes of data input between a user and a computer are still the conventional input devices such as keyboards and mouse. These devices have certain limitations when compared to the input through natural handwriting. For scripts such as Chinese and Japanese which have a very large alphabet set and due to complex typing nature of scripts such as Devanagari and Gurmukhi, it becomes difficult to input data to the computer through the conventional input devices. Natural handwriting is one of the easiest ways to exchange information between a human and a computer. Thus, the handwriting recognition field has great potential to improve the communication between the user and the computer. Handwriting recognition is basically the ability of a computer to interpret the data which is basically the intelligible handwritten input from various sources such as paper documents, touch screens and other devices [1]. The basic objective of handwriting recognition is to interpret the data and then to generate a description of that data in any desired format.

Handwriting recognition is in research for many decades and many researches on this field are going around all over the world. Great advances have been made in this field and due to this reason, the usage and reliability of online handwriting based devices such as Tablet PC's and PDA's (Personal digital assistant) has increased a lot. In addition to this, it provides a natural way of communication between the user and the computer and

will lead to an increase in usage of PDA's and Tablet PC's in languages having complex scripts such as Indian languages. In addition to handwritten character recognition, numeral recognition is also important and recognition of handwritten numerals is also a goal of various research efforts in the handwriting recognition field. The present study is based on the recognition of online handwritten Devanagari numerals. An introduction to Devanagari numerals is given in Section 1.3.

1.1 Areas of Handwriting Recognition

As already mentioned, Handwriting recognition is a technique by which a computer system can recognize characters and other symbols written by hand in natural handwriting. At the highest level, handwriting recognition can be broken into two categories on the basis of how the raw data is acquired [2]. Therefore, on the basis of raw data acquisition and the nature of handwritten data, handwriting recognition is divided into two distinct areas as follows:

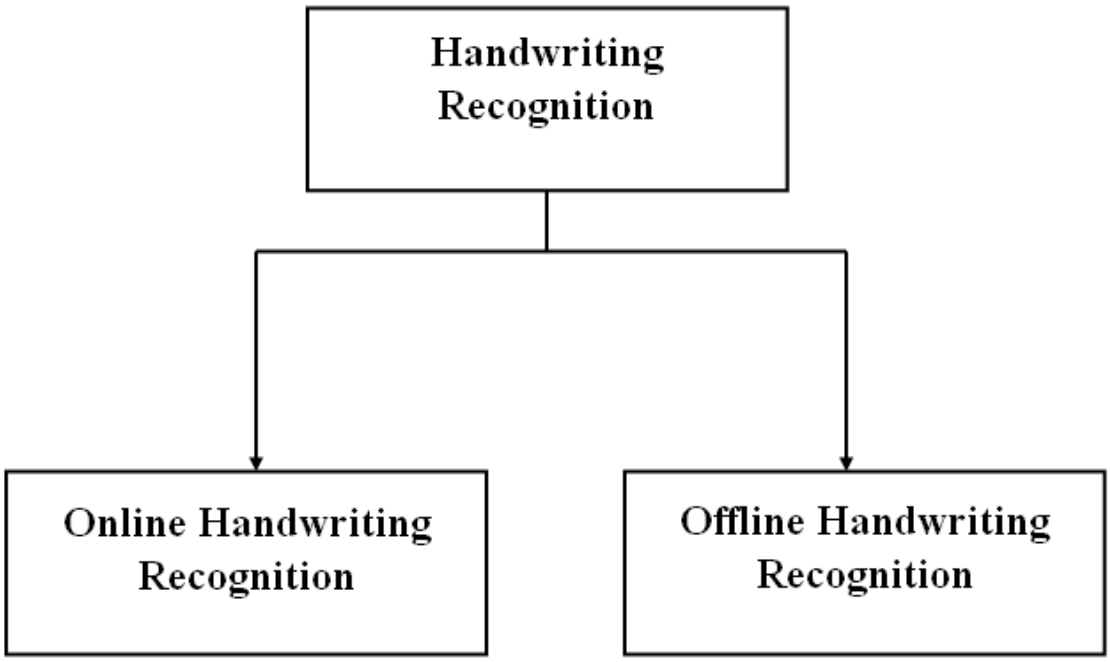

Figure 1.1: Areas of handwriting recognition

1.1.1 Offline Handwriting Recognition

In this type of recognition, the text is not recognized at the same time as it is produced but after the user has finished writing i.e. in this case, the text is originally written on a surface such as paper and from there on it is recognized by the computer by scanning the surface. The scanned handwriting is first stored digitally in grey scale format e.g. bitmap image, and then further processing is done on it to have a good recognition accuracy. Features for recognition are enhanced and extracted from the stored bitmap image by using digital image processing. This type of recognition is also sometimes called as "Optical Character Recognition" [3]. Recognition of machine printed characters is also a part of Optical Character Recognition. Offline methods are less suitable for man-machine communication because no real time interactivity is present. They are suitable for automatic conversion of paper documents to electric documents which then may be interpreted by computers.

1.1.2 Online Handwriting Recognition

In contrast to the offline method of handwriting recognition, online handwriting recognition is done in real time i.e. at the same time as the handwriting is produced. The surface used for handwriting is usually a digitized tablet and it is used along with a digital pen also sometimes called "Stylus", in order to write on the surface. As the pen moves across the surface, the two-dimensional co-ordinates of successive points are collected and stored as a function of time as shown in Figure 1.2.

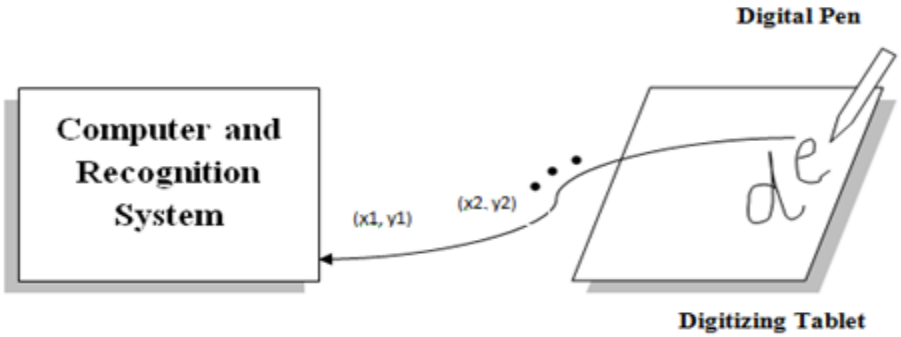

Figure 1.2: A tablet digitizer, input sampling and communication to the computer

1.1.3 Online Recognition vs. Offline Recognition

- **Real Time Interactivity:** Online handwriting recognition method is a real-time process i.e. recognition is done as the data is written. There is no real time interactivity in offline methods. Offline Handwriting recognition, by contrast, is performed after the writing is completed. It can be performed days, months or years later.
- **User Adaptation:** In case of online methods, recognition errors can be corrected immediately. If a particular character is not being recognized properly, user can change his drawing then and there and thus user adapts to the online method. No such adaptation is possible in case of offline method.
- **Machine Adaptation:** In case of online handwriting recognition, a machine can be trained to a particular user's style with samples of his misrecognized characters stored to aid subsequent recognition. No such machine adaptation is possible in offline method.
- **Preprocessing Required:** Preprocessing operations such as smoothing, de-skewing, Detection of loops etc is faster in case of online methods. Thus, lesser preprocessing time is required in case of online methods as compared to offline methods.
- **Easy Segmentation:** Due to the temporal and dynamic information collected for each stroke in case of online methods, segmentation operations are made easier.
- Applications using direct pointing and manipulation e.g. editing, annotating etc are well suited to online handwriting recognition only [3].
- Online methods capture the dynamic and temporal information of the pen trajectory which consists of number and order of strokes, direction and speed of handwriting for each stroke.
- Special equipments are required in case of online handwriting recognition e.g. Digital tablets etc. In contrary, just a pen and paper is required in case of offline handwriting recognition. Offline methods are well suited for documents printed or written on paper. Online methods cannot be applied in such cases.

- Recognition accuracies for the available systems of online handwriting recognition are low and the process is slow for the handwriting that is not clean.

The present study is based on Online Handwriting Recognition.

1.2 Issues in Online Handwriting Recognition System

At first glance, handwriting recognition may appear to be an easy task but actually there are a number of difficulties that may arise in this approach of Online Handwriting Recognition. The major problem is the vast variations present in the handwriting styles of different writers or within the handwriting style of the same writer. Variations may be present due to several reasons ranging from personal to material factors. These variations evolve with time and practice. The performance of the handwriting recognition system thus depends heavily on how well the different personal writing styles and their variations are modeled.

A good recognition system should be such that it is invariant to the meaningless variations but still be able to recognize different but similar looking characters. Due to variability in handwriting styles and distortions caused by the digitizing process, even the best handwritten character recognizer is unreliable.

1.2.1 Variations in Handwriting Styles

Variations in handwriting style occur mostly between the handwriting of different writers but may also occur within the handwriting of any individual writer. Writing units reveal number, shape and size, order of stroke and speed of handwriting. Variations in any of these are important factors to be considered. Characters can look similar although their number of strokes, drawing order and direction of strokes may vary considerably [3]. Following Figures illustrate samples of different writing styles of five different persons, variations can be noticed from these. Figure 1.3 illustrates five samples of few Devanagari numerals written by different persons. Figure 1.4 illustrates five samples of few Hindi numerals written by same writer. These samples share high degree of similarity but still some variations exist.

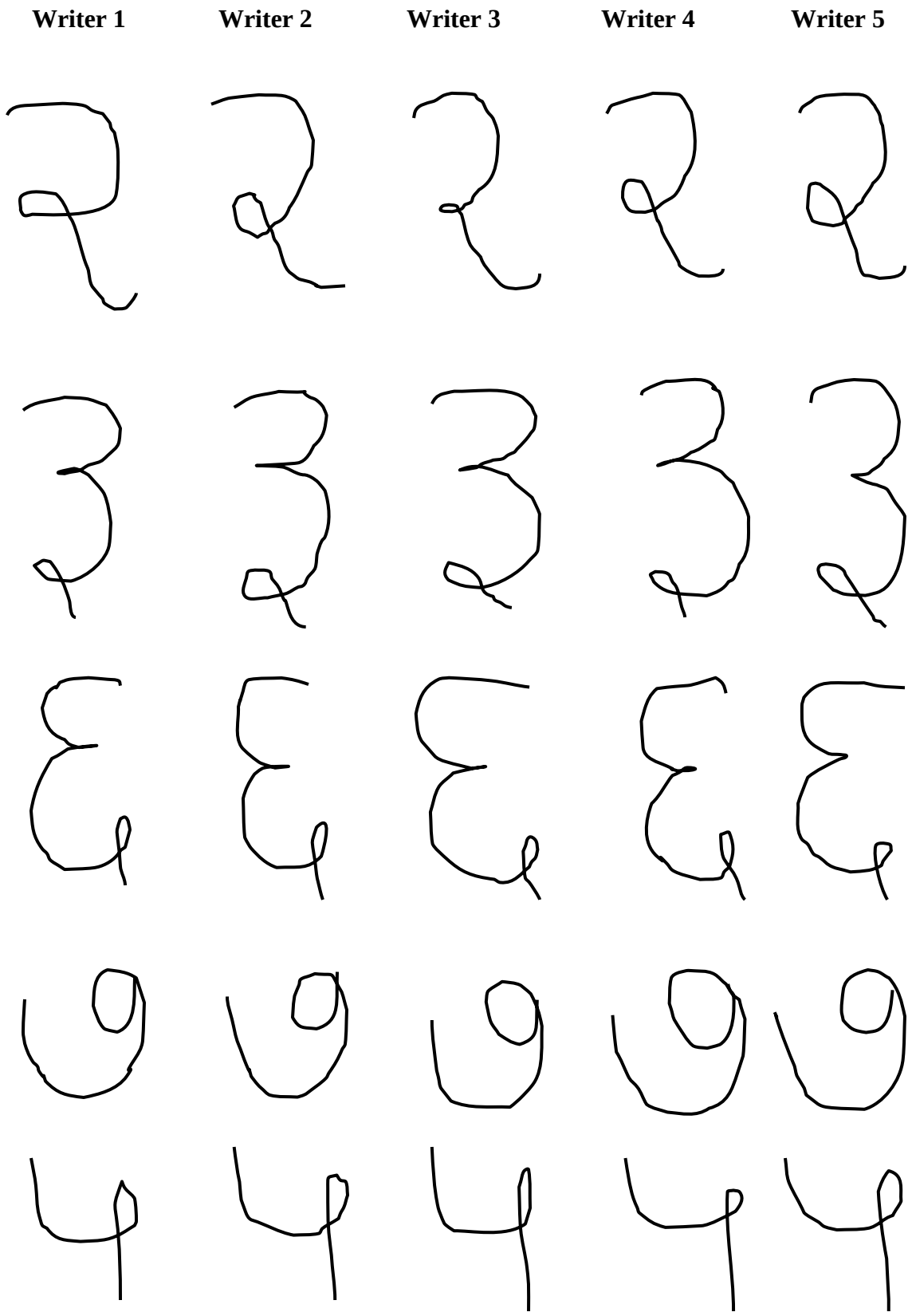

Figure 1.3 Variations in numerals written by five writers

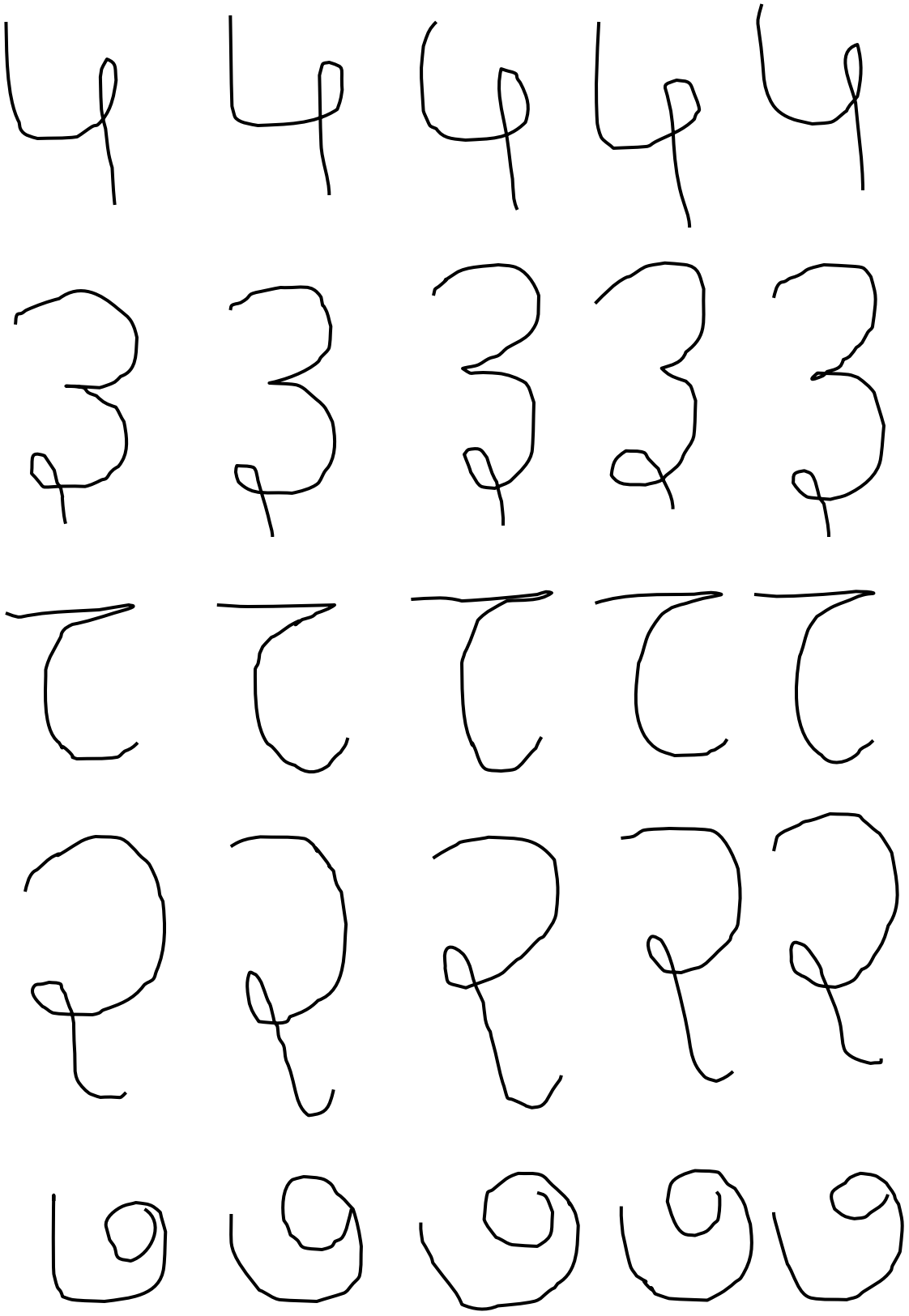

Figure 1.4 Variations in numerals written by the same writer

1.2.2 Personal or Background Factors

A writer may be left-handed or right-handed and due to this, the direction and position in handwriting may be different. Age and health may also lead to variations in handwriting by affecting the motor control of writing. Apart from these, education, origin and profession of a person may also lead to handwriting style variations.

1.2.3 Situational Factors

Different ways of presentation of handwriting also lead to handwriting variations. Situational factors include stress, hurry or distraction from the writing task. Fatigue and mood may also have an effect on the handwriting leading to variations in speed of handwriting and hence variation in handwriting styles.

1.2.4 Material Factors

Material factors depend on the hardware used in writing. Hardware may produce comfort or discomfort to the writer due to type, position and size of the writing board.

1.2.5 Writer Dependent vs. Writer Independent Handwriting Recognition

Handwriting recognition is further divided into two categories depending upon the data used for training. The two categories are namely:

- Writer Dependent Handwriting Recognition System
- Writer Independent Handwriting Recognition System

Writer Dependent Handwriting Recognition System

The systems which are based on known handwriting styles are referred to as writer dependent handwriting recognition system. In this case, the training data is collected from the same writers whose handwriting would be recognized in the future by the recognition system. Thus, it is specialized for recognizing only certain handwriting styles. These systems have achieved greater accuracy as compared to the writer dependent systems due to smaller variability present in the testing data.

Writer Independent Handwriting Recognition System

In this scenario of writer independent system, the training data is collected from different users than those from which testing data would be collected i.e. the end users of the recognition system. Thus, these systems are meant for unknown handwriting styles and are thus difficult to develop as compared to the writer dependent counterparts because all common aspects of handwriting needs to be studied. Due to their ability to recognize unknown handwriting styles, the writer independent systems are more in demand.

1.2.6 Constrained and Unconstrained Handwriting

Handwriting styles could be constrained or unconstrained in nature [4]. Constrained handwriting is further of two types: Boxed Discrete and Spaced Discrete, while unconstrained handwriting is Cursive or Mixed-Cursive in nature.

In Boxed Discrete handwriting, as the name implies, each character is written inside a special box. In case of Spaced discrete, each character is written separately with spaces and no characters touch each other. On the other hand, Run-On Discrete handwriting is the one in which each character is written separately and touches other characters. Cursive handwriting is one in which characters in one word are connected and strokes are used more than once in each individual character. Mixed-Cursive includes a mixture of all the above mentioned styles.

Mostly Cursive and Mixed-Cursive styles are adopted by people. Cursive handwriting is the most difficult to be recognized due to large amount of variability present. Each writer has their own speed of handwriting and different shapes to recognize characters. Characters in Cursive handwriting are hard to distinguish as no clear boundaries are specified between the characters.

Following Figures illustrate the concept of constrained and unconstrained handwriting. Figure 1.5 illustrates the concept of Boxed Discrete Handwriting. Figure 1.6 illustrates various styles of writing which includes Spaced Discrete, Run-On Discrete, Cursive and Mixed Cursive styles of handwriting.

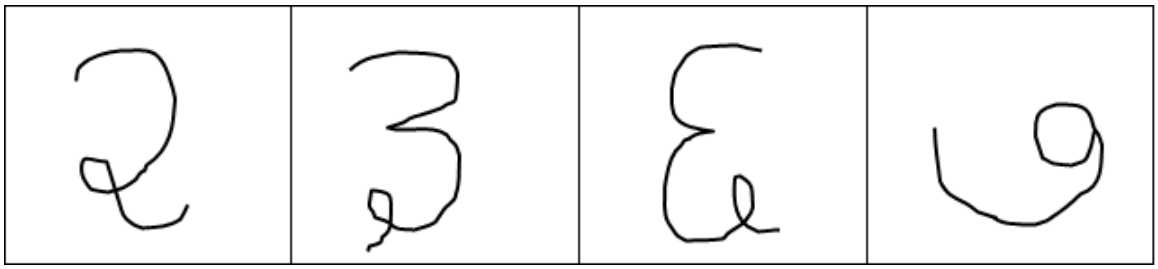

Figure 1.5 Boxed discrete handwriting

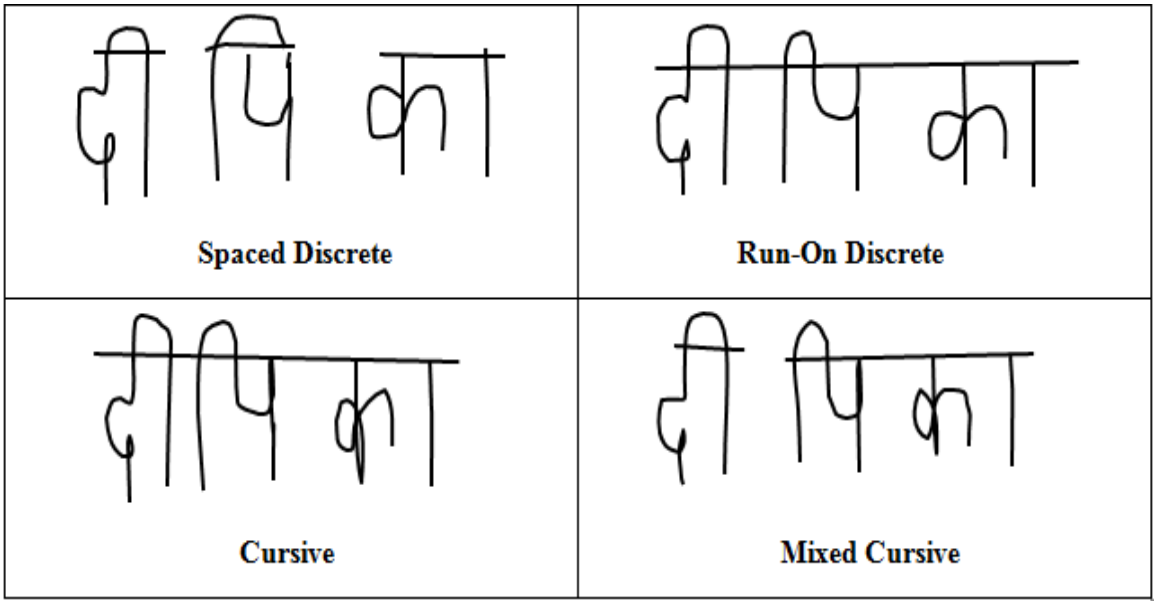

Figure 1.6 Different styles of writing in Devanagari

1.2.7 Limited Resources in Small Devices

Limitations in resources such as storage and processing capability of small handheld devices have been a bottleneck for handwriting recognition engine developers. This problem gets even more severe for languages with complex data sets such as Chinese or Japanese [5]. Thus, carefully designed algorithms which aim to optimize the amount of data and the processing requirements are needed for Online Handwriting Recognition systems.

1.2.8 Stroke Number and Order Variation

A stroke is a smallest unit in handwriting recognition and is defined as a sequence of co-ordinate points that occur between successive pen-up and pen-down movements. A character can be formed by a single stroke called a 'uni-stroke' character or may be formed using two or more strokes, called as a 'multi-stroke' character. Variations in stroke number and stroke order in a single character leads to many complications in online handwriting recognition. Thus, the issue of handling stroke order and stroke number variations is one of the major design goals while designing the online handwriting recognition systems.

1.2.9 Similarity in Shape of Some Characters

There are many characters in various scripts which have almost the same shape. Because of the same shapes, it becomes difficult to recognize them accurately. Some of these similar characters are shown in the following Figure 1.7.

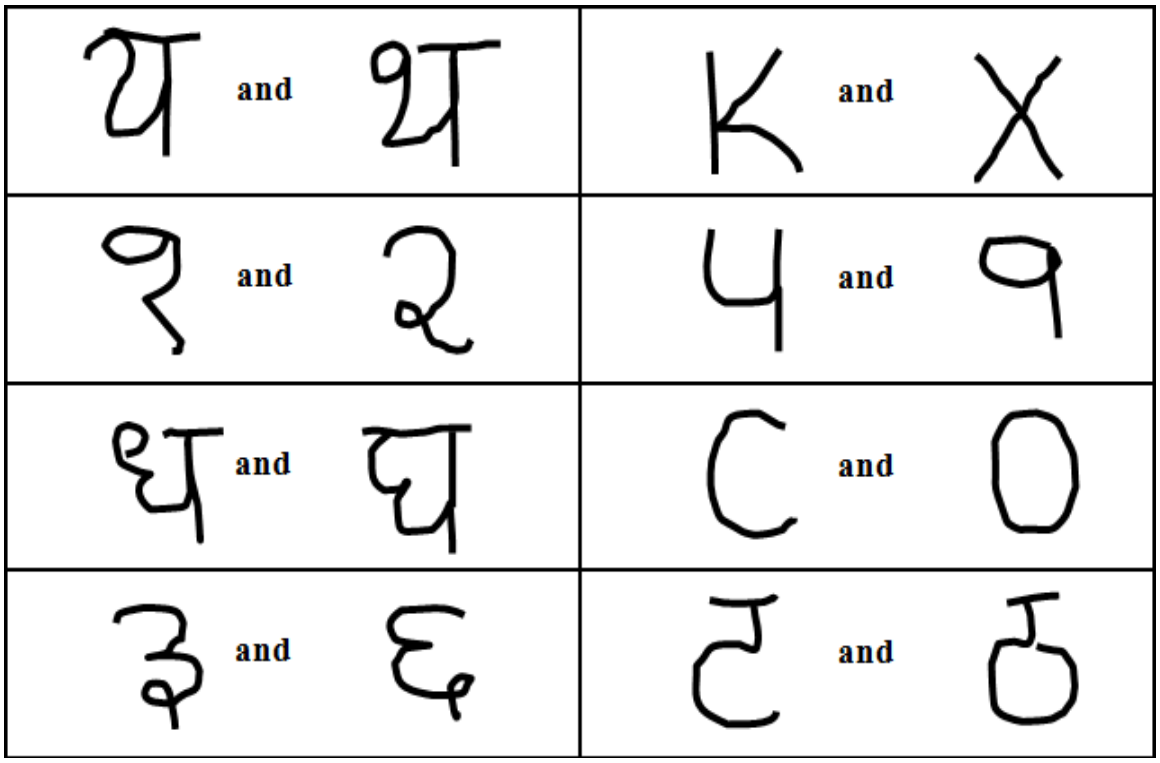

Figure 1.7 Similarities in characters

1.3 Overview of Devanagari Numerals

Devanagari script is a logical composition of its constituent symbols in two dimensions. A character is usually written such that it is vertically separate from its neighbors. It has eleven vowels, thirty three simple consonants and ten numeral symbols. The different forms of Devanagari characters [1] are illustrated in Figure 1.8.

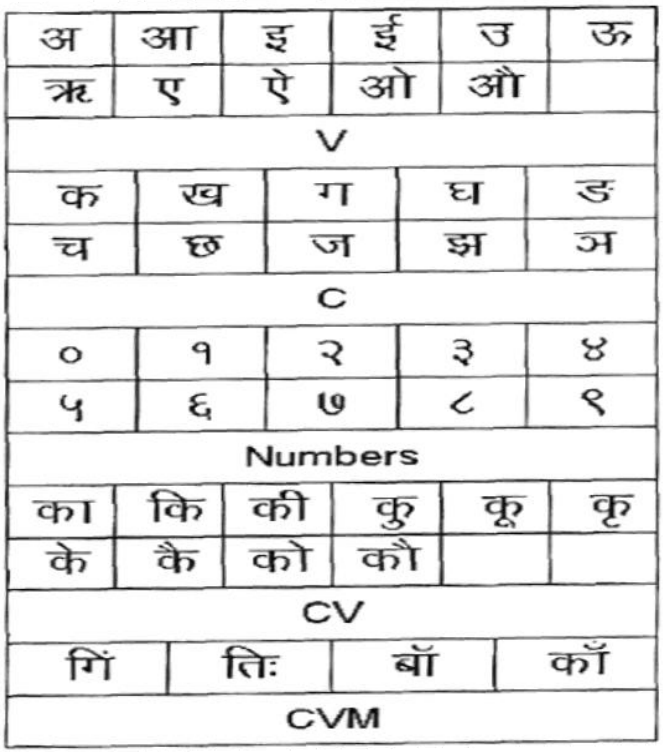

Figure 1.8 Different forms of Devanagari characters [1]

C (consonants), V (vowels), N (numerals), CV (consonant modified by vowel sound) and CVM (M is modifier that modifies the nature of the preceding vowel). The vowel Modifiers are the anuswara, the visagra, the ardhachandra and the chandrabindu. Devanagari script is written in a left to right manner and has no capital letters. It has a moderately large symbol set and shapes of characters are extremely cursive [6]. As mentioned before, it has a native set of numerals which consists of ten numeral symbols. The symbols of the numerals and their respective illustration are as shown in Figure 1.9. Hindi number system follows a single pattern of repeating itself after every ten numbers, known as Decimal system.

SYMBOL	ILLUSTRATION
०	**Shoonya**
१	**Ek**
२	**Do**
३	**Teen**
४	**Char**
५	**Paanyach**
६	**Chah**

	**Saat**
	**Aath**
	**Nau**

Figure 1.9 Devanagari numerals and their names

This dissertation is divided into four chapters. First chapter (the ongoing one), focuses on introducing the concept of online handwriting recognition. Chapter 2 contains a brief literature review which mentions the work that has already been done in this field by researchers all around the world. Chapter 3 contains the implementation part which focuses on data collection, preprocessing and feature extraction phases in order to recognize handwritten Hindi numerals. Chapter 4 consists of a brief introduction to support vector machine and explains the recognition process which is done using support vector machine. Chapter 5 concludes the work and also some light is thrown on the future work that can be carried out in this direction.

CHAPTER 2

LITERATURE REVIEW

It has been decades since the digitizer tablets have existed. These tablets used resistive techniques and analog-to-digital conversion techniques in order to measure the pen tip and capture the handwritten data.

A number of technologies were available for tablets or writing pads. These techniques were based on electromagnetic/electrostatic and pressure sensitive techniques [2]. The combination of digitizer and display in the same surface has become very common since many years. There is an established procedure to recognize online handwritten data which includes the following phases or components: Data collection, pre-processing, feature extraction, segmentation, recognition and post processing [5][7]. These phases are illustrated in Figure 2.1.

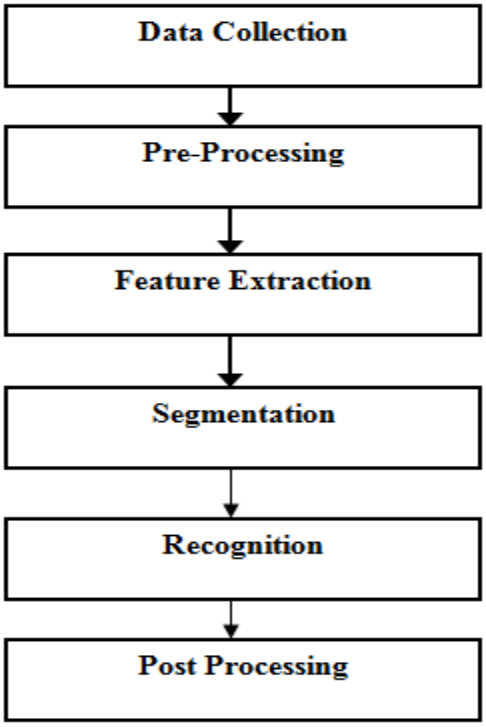

Figure 2.1 Phases of online handwriting recognition system [5]

The Segmentation phase can be performed before or after the Pre-processing phase [8]. The output of one phase becomes input to the next phase. Sections 2.1 to 2.6 discuss the literature of these phases in detail.

2.1 Data Collection

Data collection is the first phase of online handwriting recognition that collects the sequence of co-ordinate points of the moving pen. A transducer is required to capture the handwriting as it is written. The most common devices used for this purpose are electronic tablets or digitizers. A digital pen is used for writing on these devices. Digital pen is also sometimes called as 'stylus'. A typical digital pen includes two actions namely PenDown and PenUp.

STROKE: The connected parts of the pen trace between PenDown and PenUp is referred to as a stroke. It is considered as a smallest unit in handwriting recognition. All the unique strokes of a script are manually identified and given unique labels. Stroke is basically a smallest physically identifiable unit in online handwriting. The pen traces are sampled at a constant rate and thus, these pen traces are evenly distributed in time but not in space. The most common examples of electronic tablet or digitizers include Personal Digital assistant (PDA), tablet PC, cross pad (or pen tablet) and a digimemo. A digimemo is a portable device which digitally captures and stores the ink written on ordinary paper using a digital pen and pad, thus providing a natural interface for collecting handwritten data samples. These devices look like as shown in Figure 2.2. Common personal digital assistants available in the market are Pen Pad PDA 600, Apple, Casio, IBM, IBM Work Pad, Motorola, Marco, Sharp, Sony, Digital, Nokia etc.

Cross Pads available in the market are Badger, Cyrix, WebPad, Microslate, Telepad, IBM, Toshiba Tusk, Zenith, MiniWriter and ScriptWriter XL by DES etc.

Tablet PC's that are available in the market include Acer – Travel Mate C110, Fujitsu LifeBook 3000, Panasonic ToughBook C18 etc.

The selection of these hardware devices is mainly based on compatibility with operating system in use, active area dimensions and report rate of pen movements [8].

Tablet PC

Cross Pad

Personal Digital Assistant

Digimemo

Figure 2.2 Commonly used hardware devices for capturing handwriting

2.2 Pre-Processing

While inputting data through a pen on the digitizer tablets, there may be certain noise and distortions present in the input text due to some limitations, which may make the recognition of input difficult. Irregular size, missing points due to fast movement of the pen, uneven distances of points from neighboring positions are various forms of noise and distortions. These noise and distortions present in the input text are removed in the second phase of online handwriting recognition i.e. pre-processing phase.

The pre processing phase includes five common steps [5] namely:

- size normalization and centering
- interpolating missing points
- smoothing
- slant correction
- resampling of points

These steps are illustrated as in Figure 2.3.

Size normalization is required to compensate for the size, slant and rotation of the input data to make it comparable. Input handwritten texts differ in their size and location. This step is done to fix the size of the input text to a constant frame and centering is done such that the text is placed at a fixed distance from the origin. Due to fast handwriting speed, there could be some missing points in the input handwritten character. These missing points may hinder correct recognition. Thus, the missing points are interpolated using various methods. Smoothing of strokes is required to remove any noise in the stroke due to the erratic pen movement which causes jitter in handwriting. Shape of the input handwritten character is disturbed as most of the writers handwriting is bend to left or right directions. This problem of bend or slant in handwriting is solved using the slant correction methods. Duplicate points are redundant and do not contain any information. Thus, these are removed from the captured ink. Resampling of points is done to fix the number of points to be used for recognition and the fixed points are selected in such a way that the original character can be retraced from those points.

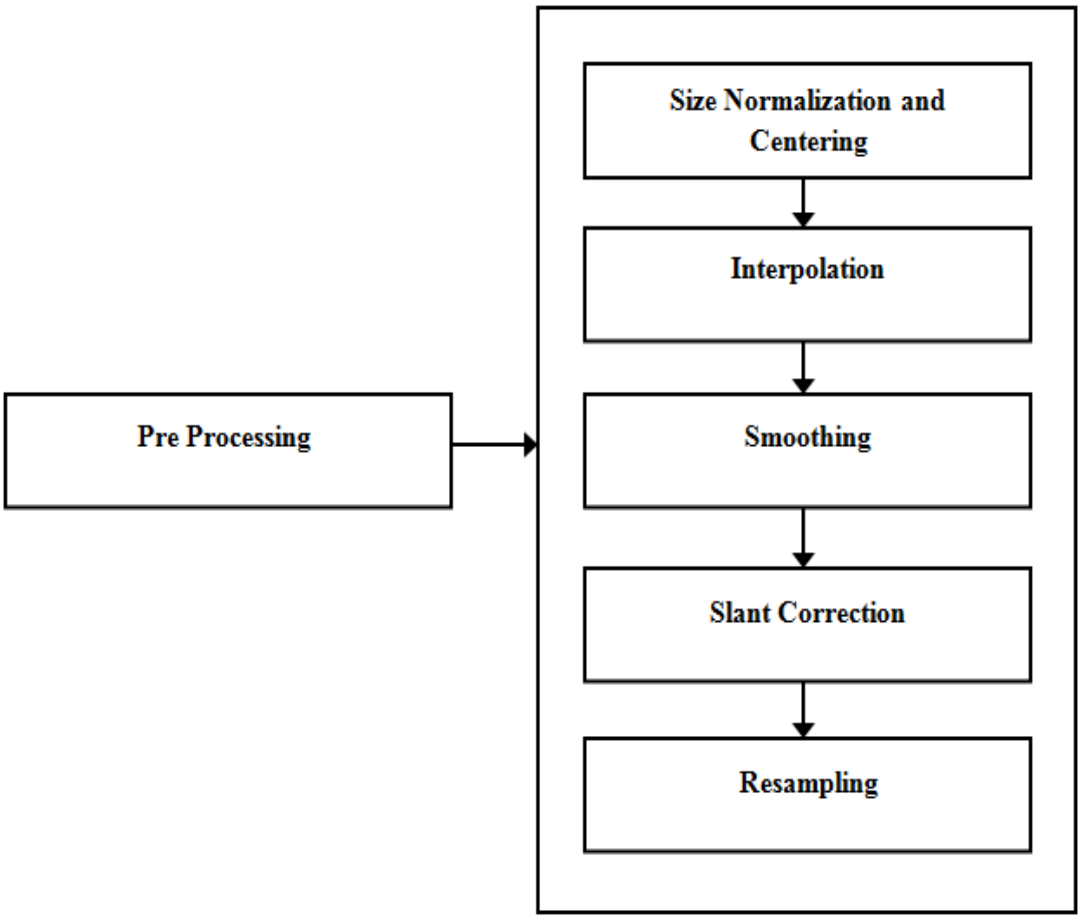

Figure 2.3 Common steps in pre processing phase [5]

Before segmentation, data undergoes smoothing and interpolation in [6]. Pre-processing phase includes the steps that are necessary to bring the input data into an acceptable form for the next phase [9]. The data is resampled to obtain constant number of points in space rather than in time as illustrated in [10]. Various slant and skew correction methods are discussed in [11]. In case of [12], the input character sample are subjected to a sequence of pre processing operations which include removal of duplicate points, resampling producing a new set of equidistant points and normalization. Size normalization is performed to rescale the input handwritten ink. Another important pre processing step i.e. elimination of hooks is carried out in [13]. Hooks occur either at the beginning or end of the stroke. These are removed by interpolating points and then detection of sharp points which are at the peak of the wavelet where the writing direction has changed. In addition,

normalization and smoothing of data is also done as part of the preprocessing step. In [14], data of a character is normalized to a fixed scale which helps to maintain the relative size and position of strokes in the character. A Gaussian filter is then used to smooth the stroke. The Gaussian filter function is given by:

$$G(u) = 1*\left(\quad e^{-u^2} / 2\sigma_{f^2} \quad \right) \Big/ \sqrt{2}\,\pi\sigma_f$$

where, σ_f is the width of the filter

Dehooking is also performed in [15], where hooks are detected by their location, small size and large angular variation. Resampling by removing the redundant points and normalization are also performed in case of [16] and [17]. In [18], missing points are calculated using Bezier interpolation. In the present study, Bezier interpolation method has been opted. The concept of Gaussian filter smoothing is also implemented in [19]. The smoothing achieved depends on the size of the window function and the smoothing kernel parameters used. The fixed number of points to which the stroke is interpolated is chosen based on the average number of points per stroke in the given data set. Size normalization and smoothing were also performed in [20]. Thinning of the input character image is also performed as a part of pre processing in certain cases. According to [21], a good thinning algorithm should preserve the connectivity and reduce the width of the skeleton to unity. Pre processing is a process that deletes points of a region but it does not remove end points, connectivity or cause excessive erosion of the region [22]. Size normalization, slant and rotation angle normalization is done along with resampling to obtain a uniform sample of desired number of points in [23].

2.3 Feature Extraction

Feature extraction is a very crucial step as the success of a recognition system is often attributed to a good feature extraction method. The feature extractor determines which properties of the preprocessed data are most meaningful and should be used in further stages. Vertical position of a point, curvature, PenUp/PenDown, writing direction, aspect, slope are amongst the various features extracted in [5]. In [6], six scalar features are extracted from each substroke. Finally, the feature vector of the substroke is defined and

used in the recognition phase. The most important aspect of handwriting recognition scheme is the selection of good feature set, which is reasonably invariant with respect to shape variations caused by various writing styles [9]. [10] Presented a scheme in which structural features at the stroke level are extracted such as mean(x, y) value, length of stroke, offline features, directional codes etc. Features are extracted using Freeman's direction code [12], in which the change in directions while moving from one point to the next one of the sample are computed and quantized into one of the 8 possible values, viz 1, 2,.. , 8. Histograms of these direction codes are computed. Direction values used in Freeman's chain code are illustrated in Figure 2.4 below.

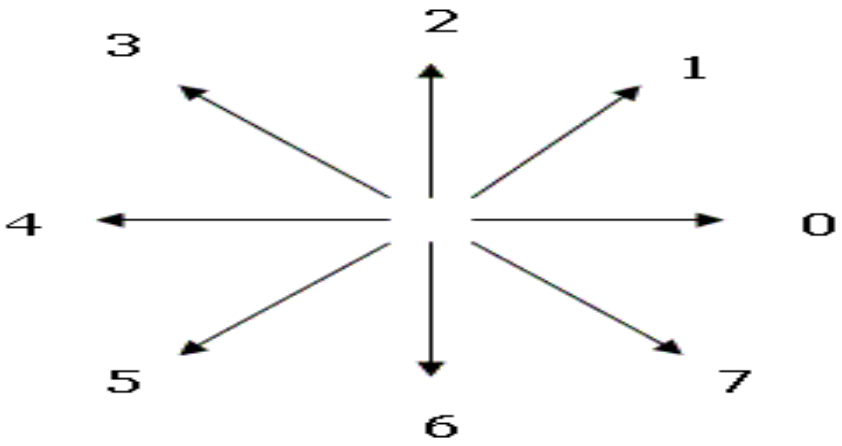

Figure 2.4 Direction values used in Freeman's chain code [12]

In [15], features such as zones and directions of start and end point, change in writing direction, height/width ratio are calculated and then represented using a global feature vector. Shape context feature, tangent angle feature, generalized shape context feature, normalized derivative features have been calculated in [16]. [17] Presented with an approach in which eighteen scalar features were extracted from the preprocessed data. Computational complexity of a classification problem can be reduced if suitable features are selected [18]. Features can be classified into two categories namely:

- Low-Level features
- High-Level features

A few neighboring points estimate low-level features whereas high level features are estimated on a larger scale as compared to low level features. A cavity is a region of points that is bounded by a stroke from atleast three sides. Cavity features provide a description of the numerals general shape which is useful for identification of the corresponding digit. These cavity features are extracted in [22]. For each point, the following seven features were extracted in [23]:

- Normalized x-coordinate
- Normalized y-coordinate
- Direction angle $\theta(x)$ of the curve according to the x-axis i.e. cosine($\theta(x)$)
- Direction angle $\theta(x)$ of the curve according to the y-axis i.e. sine($\theta(x)$)
- Curvature according to x-axis
- Curvature according to y-axis
- Position of stylus(up or down)

In [24] also, Freeman's chain code method is used for extracting direction angles of the input handwritten character. The features employed for recognition in [25] are normalized derivatives, angle features, and PenUp/PenDown bit. Almost all the researchers working on the online handwriting recognition problem take the pen trajectory directions as the main feature representing the online handwriting [26]. Normalized characters are divided into several non-overlapping zones and the pixel density is calculated for each zone and used as a feature in [27]. For extracting the features, the box approach is used in [28]. It requires the spatial division of the character image. Each character image is divided into certain number of boxes so that the portions of a numeral will be in some of these boxes. A normalized vector distance for each box is then obtained and used as a feature. This concept is also used in segmentation phase to divide the character image.

2.4 Segmentation

Segmentation is the phase in which data is represented at character or stroke level so that nature of each character or stroke can be studied individually. Segmentation is classified into two categories [8] namely:

- External Segmentation
- Internal Segmentation

External segmentation is performed prior to recognition. Segmentation performed during the process of recognition is called internal segmentation.

A character consists of one or more strokes. All possible strokes in the script are identified from the dataset and the characters are expressed in terms of strokes [4]. A stroke is segmented into several substrokes by categorizing each angle the stroke makes with the x-axis as north (N), south (S), east (E) or west (W) [6]. One substroke consists of maximum number of subsequences having the same symbol as illustrated in Figure 2.5.

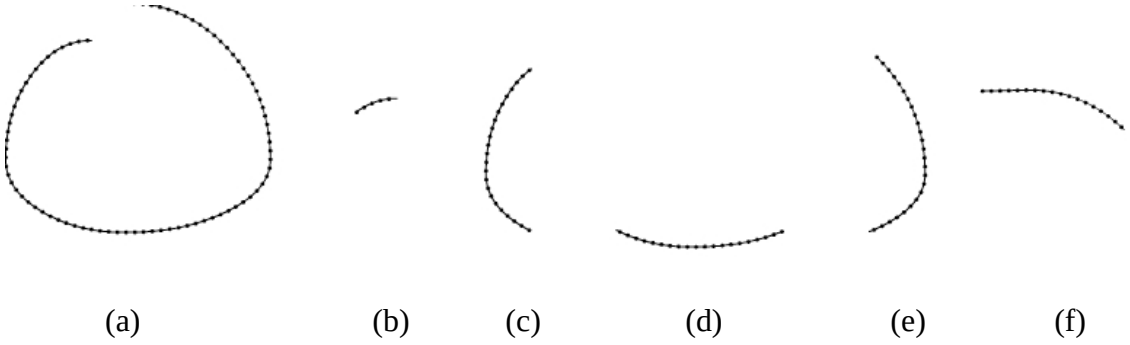

Figure 2.5 (a) Shape of a stroke, (b)-(f) Five substrokes (respectively W, S, E, N, and W) obtained from (a) [6]

The segmentation method performs an explicit segmentation of the words that deliberately proposes a high number of segmentation points offering in this way, several segmentation options [7]. A word is segmented into characters or pseudo characters, the algorithm for which is based on the analysis of the upper and lower contours, loops and contour minima. In case of [9], online handwritten character's centroid is computed and the character or numeral image is further divided into 50 equal zones (5x10 each) i.e. it is segmented. The proposed technique in [14] consists of a stroke set being divided into subsets based on their position with respect to baseline as follows:

- Group 1 : Strokes touching the baseline or just above the baseline (Baseline strokes)
- Group 2 : Strokes below the baseline (Bottom strokes)
- Group 3 : Strokes well above the baseline (Top strokes)

This concept is illustrated in Figure 2.6.

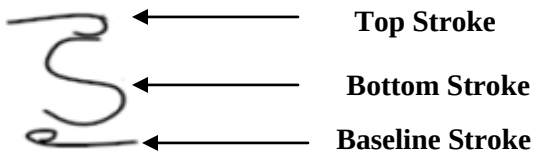

Figure 2.6 Examples of top, baseline and bottom strokes [14]

The whole trajectory of the pen forming a character sample is divided into N subdivisions in [17]. Each character is a group of one or many strokes. Therefore, all the collected strokes are assigned temporary unique identification number in [18]. A character consists of one or more strokes. All possible strokes in the script are identified from the data set and characters are expressed in terms of strokes [19]. Strokes are divided into several substrokes. According to [22], Character segmentation is an operation that decomposes an image of a character into a set of sub images of digital primitives such as arcs, curves, loops, lines, and dots. Segmentation method can be classified into three major classes namely:

- Image based segmentation or dissection, in which an image is segmented based on the structural and topological features of the input character.
- Recognition-based segmentation in which the image is searched for components that match classes in its alphabet.
- Holistic methods, in which whole words are recognized instead of characters.

An algorithm for extracting primitives is proposed in [24]. Primitives include line, arc and dot. Thus, a character is being segmented into a line, arc or dot and then these primitives are studied further. Segmentation is an important step. Therefore, the extent upto which separation of words, lines or characters is done directly affects the recognition rate of the script. Line and character segmentation is performed in [27].

In [28], each character image is divided into a fixed number of boxes; the choice of number of boxes is arrived by experimentation. There will be portion of numeral in some of these boxes. This method of segmentation is referred to as ‘Box Method’. The concept is illustrated in Figure 2.7.

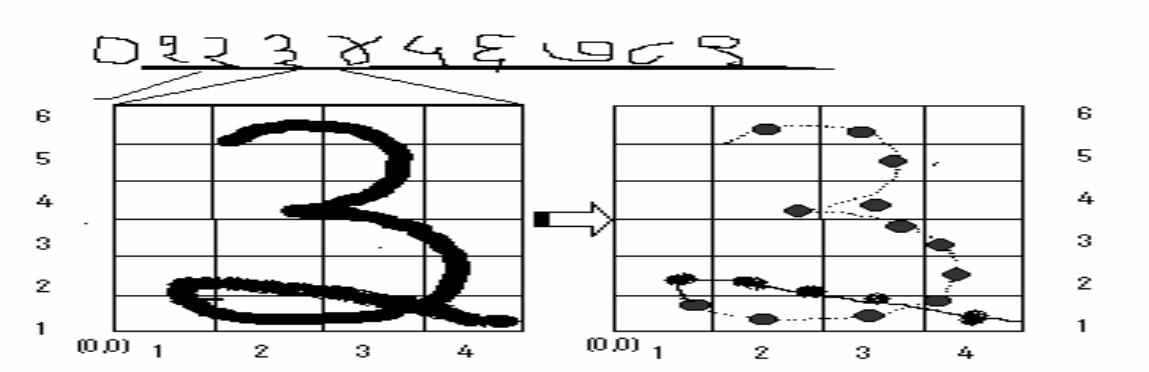

Figure 2.7 Feature extraction of numeral '3' using box method [28]

The segmentation algorithm applied in [29] renders the high degree velocity profile of raw handwriting into a simpler data-lognormal profile, each corresponding to a stroke.

2.5 Recognition

Recognition is the most important phase of the online recognition system and uses the features extracted in the previous stage to identify the input character according to preset rules. In many of the emerging new applications, it has become clear that no one method for recognition can be considered as optimal and usually a combination of multiple methods yield better recognition results. Combination of two or more recognition methods is a common practice now-a-days and is referred to as hybrid methods for recognition. The four best known approaches for pattern recognition are: 1) template matching, 2) syntactic or structural matching, 3) neural networks and 4) statistical classification [30] [31]. These methods are briefly described below:

2.5.1 Template Matching

Matching, as the name implies is used to measure the similarities between any two entities of the same type and is a generic operation. In case of template matching, the preprocessed pattern or the handwriting to be recognized is matched against a stored template or prototype which is already available. The measure of similarity is optimized using a training set. The stored template is also often learned from the training set. This method has a number of disadvantages as it fails in cases when the pattern is distorted due to imaging process or large intra-class variations amongst the patterns [30]. This

method is also referred to as ‘Elastic matching’ in its non-linear form. One of the important approaches based on elastic matching is the DTW i.e. Dynamic Time Warping approach.

2.5.2 Syntactic or Structural Matching

In handwritten patterns where structures and grammar are considered, syntactic or structural matching approach is used. It describes the construction of a pattern from its primitives. This paradigm has been used in situations where the patterns have a definite structure which can be captured in terms of a set of rules, such as waveforms, textured images, and shape analysis of contours. In case of syntactic approach, a formal comparison is made between the structure of the pattern and the syntax of a language. Patterns are viewed as sentences belonging to the language. Patterns are broken down into separate physically identifiable primitives which are viewed as the alphabets of the language and the sentences are generated according to the grammar. The grammar for each pattern class is inferred from the available training samples [30].

2.5.3 Neural Network Methods

Neural networks can be viewed as parallel computing systems consisting of an extremely large number of simple processors with many interconnections [8]. Neural network models attempt to use some organizational principles such as learning, generalization, adaptability, fault tolerance, distributed representation and computation in a network of weighted directed graphs in which the nodes are artificial neurons and directed edges are connections between neuron outputs and neuron inputs. The main characteristics of neural networks are that they have the ability to learn complex nonlinear input-output relationships, use sequential training procedures and adapt themselves to the data [30]. The most commonly used neural networks for pattern classification tasks are feed-forward network, which include multilayer perceptron and radial basis function networks. These networks are organized into layers and have unidirectional connections between the layers. Another popular network is the self-organizing map or kohonen network.

2.5.4 Statistical Classification

Statistical classification is the problem of identifying which of a set of categories a new observation belongs, on the basis of a training set of data containing observations whose category membership is known. In case of statistical approach, each pattern is viewed as a point in the d-dimensional space. Features are chosen so that disjoint regions in a d-dimensional feature space are occupied by the pattern vectors belonging to different categories [30]. Decision boundaries are established in the feature space to separate the patterns belonging to different classes. Parametric and non-parametric are the two classifications of statistical methods. Parametric methods are the one in which handwriting samples are categorized using a set of parameters which are selected based on the training data. Non-parametric methods are based on direct estimation from the training data.

HMM i.e. Hidden Markov Models, k-NN i.e. k-Nearest Neighbors and SVM i.e. Support Vector Machines are few important examples of statistical recognition. Present dissertation is based on recognition using SVM.

'Template matching' or elastic matching methods of recognition have been used in [8][16][18][24]. DTW i.e. Dynamic Time Warping has been used in [16] which is an elastic matching method. It is a technique that finds optimal alignment between two time series if one time series may be warped non-linearly by stretching or shrinking it along its time axis. This warping between 2 time series can then be used to find the similarity between them. In [18], an unknown stroke is matched against all the strokes in the stroke database using the elastic matching approach. Matching process in [24] attempts to find match between the drawn shape's model and the predefined models. When match is found, the shape is recognized.

In [10], 'structural and syntactic approach' has been employed for the recognition of Devanagari characters. Shirorekha is detected using the structural approach and then feature based recognition is done in order to recognize the residual character. The normalized x and y coordinate sequences are classified directly using the subspace method. After the feature extraction phase, a syntactic representation of the input image

is obtained which is then further matched against the syntactic representation prototype in [22].

Two 'neural network methods' namely TDNN i.e. Time Delay Neural Network and MLP i.e. Multi Layer Perceptron model are discussed in [5][15][17][32] and [33]. MLP is a feed forward artificial neural network which maps a set of input to appropriate output. It contains multiple layers which are fully connected to each other and uses supervised learning technique called back propagation for training. TDNN is a convolution neural network with temporal shift i.e. TDNN is able to recognize a pattern independently of its position in time. A MSTDNN i.e. Multi State Time Delay Neural Network is used as a recognizer in [5]. It is an extension to TDNN which has been applied successfully to online single character recognition task. MLP is used for the classification task in [17] and is trained using the back propagation algorithm. A common problem of choosing MLP as the classifier is finding a suitable choice of its architecture (number of hidden nodes) and values of different parameters (such as learning rate and momentum factor) of the BP algorithm. Convolution neural networks are derived from MLP [33].

The Hidden Markov Model (HMM) is a variant of a finite state machine having a set of hidden *states*, $Q=\{ q_i \}(i=1,\ldots,N)$, an output alphabet (observations), $O=\{ o_k \}(k=1,\ldots,M)$, transition probabilities, $A=\{ a_{ij} = P(q_j \text{ at } t+1 \mid q_i \text{ at } t)\}$, output (emission) probabilities, $B=\{ b_{ik} = b_i(o_k) = P(o_k \mid q_i)\}$, and initial state probabilities, $\Pi=\{ p_i = P(q_i \text{ at } t=1)\}$. The current state is not observable. Instead, each state produces an output with a certain probability (B). Usually the states, Q, and outputs, O, are understood, so an HMM is said to be a triple, (A, B, Π). A stroke based recognition approach, where strokes are identified using HMM i.e. Hidden Markov Models is employed in [6]. In case of [7], recognition is carried out by a lexicon-driven decoding algorithm where each word in the lexicon is modeled by a "super-HMM" created by concatenating character HMM's. The decoding algorithm finds the N best word hypotheses that have the highest likelihood given the observation space. The concept of HMM is also used in [25] for online handwritten Tamil word recognition. Word recognition system used in [34] is lexicon-driven and based on HMM based modeling of symbols and lexicon words. There are several reasons behind the success of HMM in the field of online handwriting

recognition. Since it is a stochastic model, thus coping up with the noise and distortions in handwriting is easier. Furthermore, word HMMs can solve the problem of segmentation implicitly. Recognition techniques based on a degenerate single state continuous density hidden markov model were used for the purpose of shape recognition in [35].

In case of NN classifier i.e. nearest neighbor classifier, a database is created from the available samples i.e. training set. Whenever there is a new object or pattern to be classified, system finds the database object most similar to the query. Query is classified to the same class as its nearest neighbor. The concept of NN classifier is used in [36] for the recognition of unconstrained Devanagari characters. The recognition task is accomplished using a combination of five classifiers. Two HMM classifiers, which focus on two different levels of local features along the trace of the online sample point sequence, and three NN classifiers that focus on offline features.

SVM stands for Support Vector Machine and this method of recognition is gaining immense popularity now-a-days. SVM classifier gives better results as compared to the other classifiers for the handwritten numeral recognition of Kannada script in [9]. In [14], SVM approach is used to recognize strokes in Telugu script. The set of strokes are segmented into subsets based on the relative position of the stroke in a character. An SVM based classifier is built for recognition of strokes in each subset. A rule based approach is used to recognize a character from the sequence of stroke classes given by the stroke classifier. SVM is a new classifier that is used in many pattern recognition applications with good generalization performance. SVM has been used in recent years as an alternative to popular methods such as neural network. The advantage of SVM, is that it takes into account both experimental data and structural behavior for better generalization capability based on the principle of structural risk minimization (SRM) [23]. The recognition rate using SVM is higher as compared to TDNN and MLP NN. The SVM is a new type of hyperplane classifier, developed based on the statistical learning theory of Vapnik, with the aim of maximizing the geometric margin of hyperplane [27]. Research of SVM has been a boom from the mid-1990s, and the application of SVMs to

pattern recognition has yielded state-of-the-art performance. SVMs are mostly considered for binary classification.

The present dissertation is based on recognition of online handwritten Hindi numerals using SVM. A brief introduction to SVM is given in chapter 4.

Apart from using single classifiers for recognition, hybrid classifiers are also used for better and optimal results. An NN classifier based on DTW is used in [12]. The hybrid HMM-MLP approach is used in [13] which combines hidden markov model and multi layer perceptron classifier. A new hybrid classification strategy in handwriting recognition field is introduced in [20]. In this strategy, a multilayer perceptron (MLP) architecture is used for feature extraction purpose and a support vector machine (SVM) is used for final classification. An approach that combines SVM and HMM is presented in [29] with the basic idea to employ a set of left-right HMMs as a feature extractor to produce HMM features, and combine them with global features into a new feature vector as input, and then use SVM as a classifier to finally identify unknown symbols. A recognizer based on this method inherits the practical and dynamical modeling abilities from HMM, and robust discriminating ability from SVM for classification tasks. This technique also reduces the dimensions of feature vectors significantly and solves the speed and size problem when using only SVM. In [36], five classifiers are used in all for the recognition process which include 2 HMM and 3 NN classifiers. The NN classifiers which focus on the offline features capture more global structural properties of the characters, which are calculated separately for each stroke. A novel classification approach for online handwriting recognition is presented in [37] which combines DTW and SVM by establishing a new SVM kernel called Gaussian DTW. This method has a main advantage over common HMM techniques that it directly addresses the problem of discrimination by creating class boundaries and thus is less sensitive to modeling assumptions. The experimental results show that this combined approach outperforms several classifiers reported in recent researches.

2.6 Post Processing

Post processing is the last and very important phase of online handwriting recognition in which misclassified results are corrected by applying linguistic knowledge. In case of handwriting input, certain shape recognizers yield string of characters, while others may provide a number of alternatives for each character, often with a measure of confidence for each. Post processor can then operate on this information to obtain estimates for larger linguistic units such as words [8]. Alternate choices provide more information for post processing. In post processing, a dictionary can be used to restrict the character combinations. This can be implemented as a grammar that supplies all possible combinations of characters. When shape recognizer yields a single choice for each character, in that case string correction algorithms are applicable [2]. In case of [6], look up tables are constructed on the basis of the training set of the character sample and are used in the second stage of classification to get the final characters. If the description is present in the look up tables for more than one character class, then one having the maximum probability is retained and others are removed. An approach with multiple SVM engines for stroke recognition was employed in [19] for the enhancement of the recognition accuracy. Usually, in a hybrid recognition technique involving an artificial neural network, the values at the output layer are considered for further processing before arriving at the final classification result. In the recognition of Hindi numerals in [22], in particular cases where two numerals are similar in topology, additional geometrical tests are performed to identify uniquely the input numeral for which several tests are designed. In [26], concepts of dynamic programming are used to find a globally optimal solution. More than one possible answer can be found after recognition. Comparison continues in a tree form until the whole stroke is totally recognized. Five classifiers are used in [36], two hidden markov models and three neural network classifiers, out of which the last two classifiers are used for post classification i.e. they capture more global structural properties of the characters, which are calculated separately for each stroke and thus help in the correction of misclassified results by the third classifier. For many real world applications, it is more important to minimize the misclassification rate than to maximize the rate of well classified patterns [38].

2.7 Previous Recognition Results in Online Handwriting Recognition

This section presents some of the selected recent results from literature of online handwriting recognition systems. Most of the results given in this subsection are based on online handwritten character recognition as present study is focused to this area only. Also, these results cannot be compared as most of these systems are tested and trained with different characters databases, writer's databases and recognition methods.

Table 2.1: Selected recognition results from literature of online handwriting recognition systems

Author(s)	Method	Dataset	Recognition /Error rate
Scott D. Connel et al. (2000)	HMM and NN	Unconstrained Devanagari characters with writer independent system	86.5%
Fabrice Alleau et al. (2003)	TDNN	(0-9) digits, (A-Z) characters and (a-z) characters; UNIPEN+ IRONOFF Database	97.9%, 91.6%, 94.0%
Abdul Rahim Ahmad et al. (2004)	SVM	(0-9) digits, (A-Z) characters and (a-z) characters; UNIPEN+ IRONOFF Database	98.68%, 93.76%, 95.13%
Brijesh Verma et al. (2004)	NN Classifier	(0-9) digits, (A-Z) characters and (a-z) characters; UNIPEN	98.0%, 98.0%, 99.3%

		Benchmark database with writer independent system	
Ibrahim M. M. El Emary et al. (2005)	Structural approach	Indian numbers with writer independent system	94.87% - 98.07%
Niranjan Joshi et al.	Structural Recognition and Feature based matching	Devanagari characters with writer dependent system	94.49%
B. Q. Huang et al. (2006)	Hybrid SVM-HMM approach	(0-9) digits, (A-Z) characters and (a-z) characters; UNIPEN With writer independent system	97.48%, 91.99%, 91.74%
B. Q. Huang et al. (2007)	Hybrid HMM and MLP approach	(0-9) digits, (A-Z) characters and (a-z) characters; UNIPEN with writer independent system	97.3%, 91.7%, 91.5%
Bharath A and Sriganesh Madhvanath (2007)	HMM	Tamil Characters with writer independent system; UNIPEN	Ranging from 98% to 92.2% with different Lexicon sizes
P. Bilane et al. (2007)	DTW algorithm	(0-9) digits, (A-Z) characters and (a-z) characters; UNIPEN with writer independent system	97%, 94.35%, 93.35%

Randa I. Elanwar et al. (2007)	Minimum edit distance based on Dynamic Programming	317 words(1814 characters) with writer independent system	92%-95%
Anuj Sharma et al. (2008)	Elastic Matching	Gurumukhi characters with writer independent system	90.08%
S. V. Rajashekararadhya et al. (2009)	SVM	Numerals of Kannada script	97.75%
L. Prasanth et al. (2010)	DTW	Telugu and Tamil scripts with writer dependent system	90.6%
Ujjawal Bhatacharya et al. (2011)	MLP	Bangla characters with writer dependent system	93.90% training 83.61% testing

2.8 Problem Statement

Based on the survey carried out in this section, we can conclude that there is no work done for Devanagari script for recognizing the online handwritten Devanagari or Hindi numerals using SVM as the recognizer. Also, SVM as a recognition method is gaining immense popularity and the recognition results obtained are excellent. Thus, in the present study, SVM has been used as a recognition method for online handwritten Devanagari numerals. Pre processing and feature extraction phases aid to the recognition process. The more is the amount of pre processing done, the better is the recognition accuracy. Also, the more are the number of features extracted from the input character, the easier it becomes to recognize the character for the recognition system. The present study is carried out focusing on pre-processing and feature extraction along with recognition.

CHAPTER 3

DATA COLLECTION, PRE-PROCESSING and FEATURE EXTRACTION

In the present dissertation, recognition of Devanagari numerals is done using SVM i.e. Support vector Machines. Four phases of online handwriting recognition have been implemented namely: Data collection, pre-processing, feature extraction and recognition which are described in the present chapter and chapter 4. Data collection, pre-processing and feature extraction are three important phases which are performed before recognition in online handwriting recognition. Section 3.1 focuses on first phase i.e. data collection. Pre-processing has been explained in section 3.2 and section 3.3 gives an overview of the feature extraction phase.

3.1 Data Collection Phase

A typical format of online handwriting data is a sequence of coordinate points of the moving pen point. The characters are inputted with the help of mouse or written on writing pads by a digital pen. The time sequential signal from the pen movements on the writing pad collected by the sensor attached to the pad is described as a sequence of consecutive points on the x-y plane. While writing in a slow motion, the sample points are located very close to each other i.e. densely and thus more number of points are collected. On the other hand, fast handwriting leads to points that are sparsely located. The points generated by these pen movements are stored in a list.

In case of Devanagari numerals, a total of ten symbol set is there. The numerals are written by ten writers. Each writer was asked to write ten samples of each Hindi numeral, thus 1000 samples in all are created during this work. The following Figure 3.1 represents the points when the numeral is written on the writing pad or the writing area using a mouse. As the numeral is written in the writing area, its x and y coordinate points are collected as the numeral is written. These collected x and y coordinates may vary in number depending upon whether the numeral is written in a slow or fast handwriting.

This variation is taken care of in the pre-processing phase by resampling the points. The collected x and y coordinates are used further in the feature extraction phase to calculate the direction angle and curvature between the points which is discussed later in this chapter. Thus, these points collected in the data collection phase act as intermediary to be used in further phases. The data collected in the data collection phase is sent to the pre-processing phase, the next phase of the recognition system.

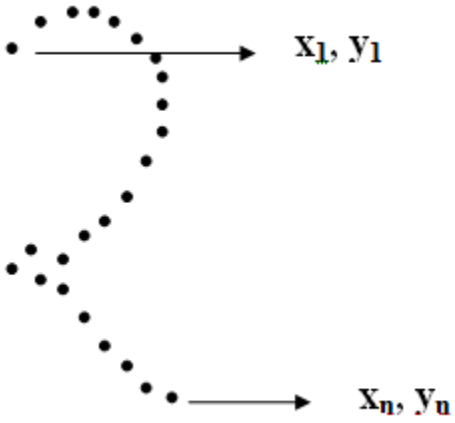

Figure 3.1 Points of pen movement collected while writing

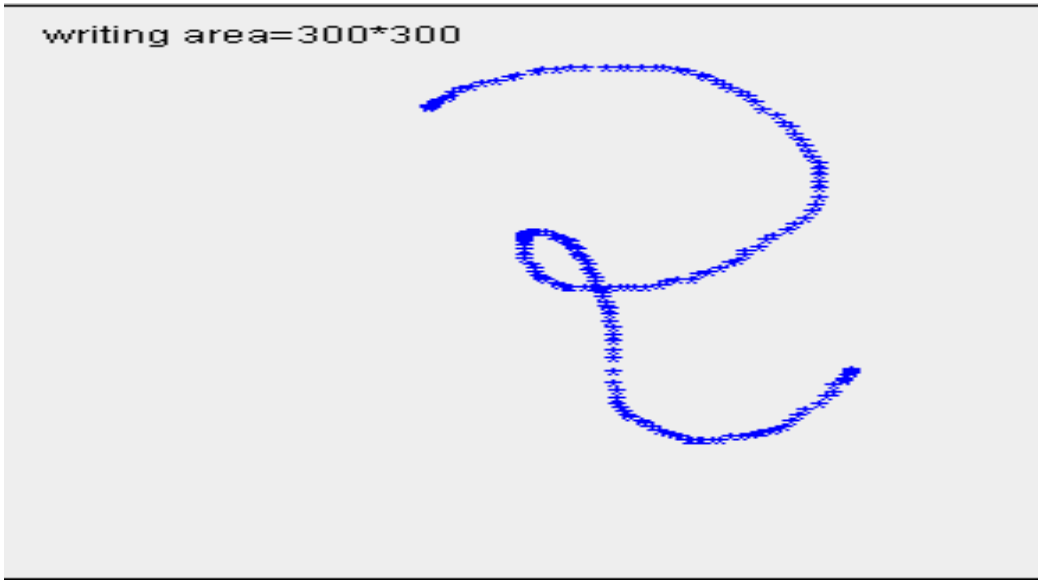

Figure 3.2 Writing area for writing the numerals

Figure 3.2 represents the writing area of dimensions 300*300, where the writer writes the character or the numeral with the help of a mouse or a digital pen. It is from this writing area that the coordinate points are collected as the character is written.

3.2 Pre-processing Phase

The main aim of the pre-processing step is to remove any irrelevant information and variations present in the input handwritten characters, which would otherwise cause problems in recognition. Irrelevant information generally includes duplicate points, jitter etc. According to the literature survey done, pre-processing improves the rate of recognition. The main pre processing steps considered in this dissertation include:

- Size Normalization
- Interpolation
- Resampling
- Smoothing
- Slant Correction

These steps of pre processing are discussed in detail as below. The algorithms used in this phase of pre processing are referred from [8].

3.2.1 Size Normalization and Centering

Size normalization and centering are important processes that must be performed in order to recognize a character. The input stroke varies in size and depends upon how the writer moves the pen on the writing pad. The stroke is generally not centered. As the input handwritten texts differ in their size and location, size normalization and centering is required to fix the size of the input text to a constant frame and to place the text at a fixed distance from the origin. This can be achieved by comparing the input character border frame with an already assumed fixed size frame and the input character can be moved along to an assumed center location for centering.

The algorithm implemented in this dissertation fixes the size of the input character to a fixed size of 200x200 pixels and places it in the center of the fixed frame. Thus, if a character of size less than 200x200 pixels is entered, it is transformed into a fixed size of

200x200 pixels. Similar thing happens when a character of size larger than the fixed size is entered, it is reduced to the fixed size of 200x200 pixels. This concept is illustrated in Figure 3.3 where the window is divided into two parts. The first part represents the writing area where input stroke is drawn and the second part represents the normalization area where character after normalization is shown.

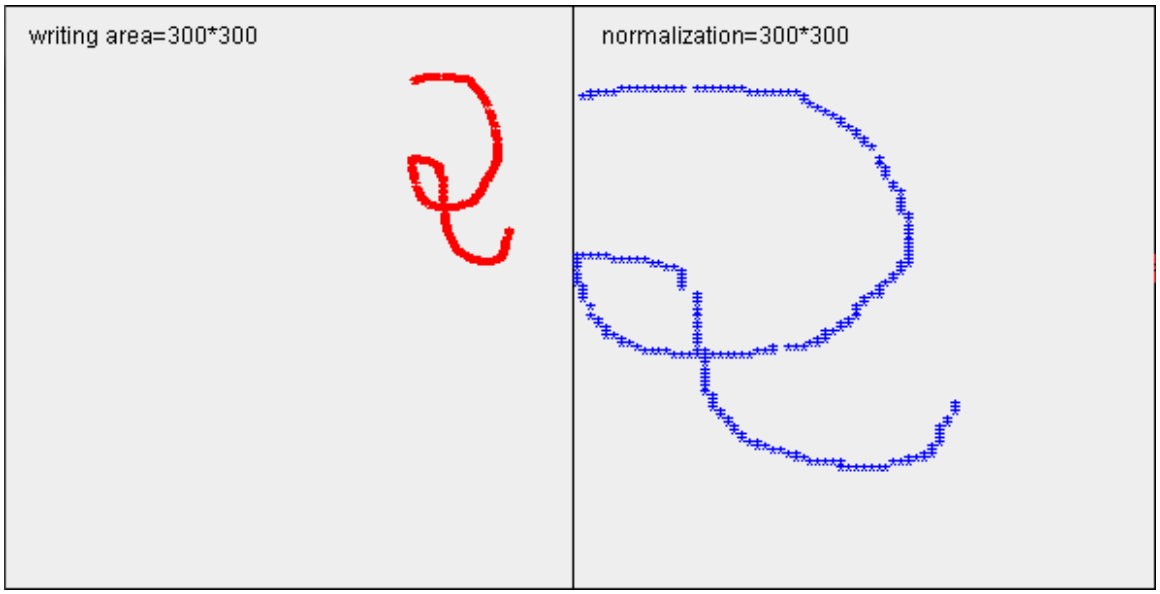

Figure 3.3 Handwritten character after normalization

3.2.2 Interpolating Missing Points

The number of points collected during the data collection phase depends on the speed of handwriting i.e. the speed at which the pen is moved on the writing surface. When the character is drawn with a high speed, it leads to some points missing in it. These missing points can be interpolated using techniques such as Bezier and B-Spline. In the present work, piecewise interpolation method has been used and it helps to interpolate points amongst fixed number of points and this also leads to equidistant points. In this interpolation technique, a set of four consecutive points is considered for obtaining a Bezier curve. The next set of four points gives the next Bezier curve.

Bezier interpolation is applied between the points where the distance is greater than one. First the algorithm finds the distance between the pair of consecutive points and calls Bezier where the distance between the points is greater than one. Interpolation is done

between those points whose distance is greater than one by making a Bezier curve. Figure 3.4 illustrates the input character after Bezier interpolation has been performed. The Figure consists of three partitions. The first one shows the writing area, the second shows the normalized character. The normalized points are then used for Bezier interpolation which is illustrated in partition three.

Figure 3.4 Handwritten character after Bezier interpolation

3.2.3 Resampling

Resampling of points is done to fix the number of points in the input handwritten character and also retain the original shape of the character. A filter is applied such that only a fixed number of points are selected. In the present study, the number of resampled points is fixed to 64. The points are selected in such a way that when the resample points are retraced, the shape of the original character must be regained. After interpolation of points, since any pair of points will be at a maximum distance of one, removal of points shall include two options: remove all points between pair of points having distance less than one, or, remove points at constant distances, *i.e.*, two or three and so on. Figure 3.5 illustrates the resampling done on the input handwritten character.

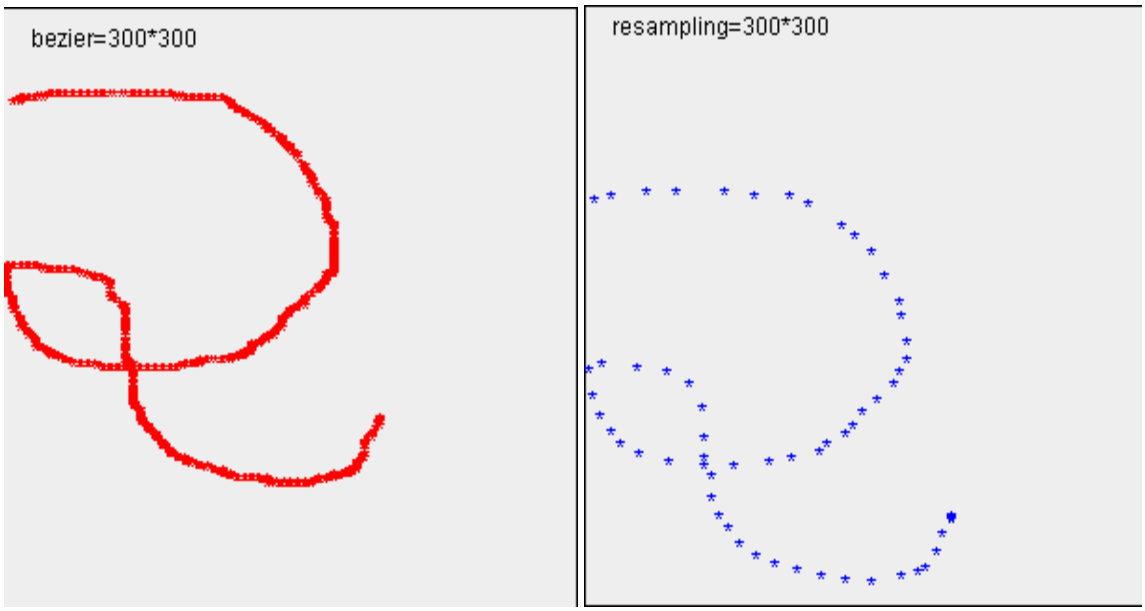

Figure 3.5 Handwritten character after resampling

3.2.4 Smoothing

Due to individual handwriting styles and hardware limitations, there may be flickers and jitters present in the handwriting. Smoothing is done in order to remove these flickers and jitters from the input handwritten character. The algorithm used for smoothing is based on modifying each point of the list with mean value of k neighbors and the angle subtended at k^{th} position from each end. In the present work, two neighbors from each side have been considered i.e. the value of k is chosen to be two. If three points are considered instead of five, it will not affect the nature of the stroke and if more than five points are considered then nature of the stroke may be lost in terms of sharp edges.

3.2.5 Slant Correction

The problem of bend or slant in handwriting is present due to different handwriting styles of the writers. This problem can be solved using the slant correction methods. Chain code estimation method has been applied in the present work, for slant correction of Hindi numerals. An eight directional chain code method [39] is applied as discussed in Chapter 2. Figure 3.6 illustrates the effects of all the preprocessing steps on the input handwritten character.

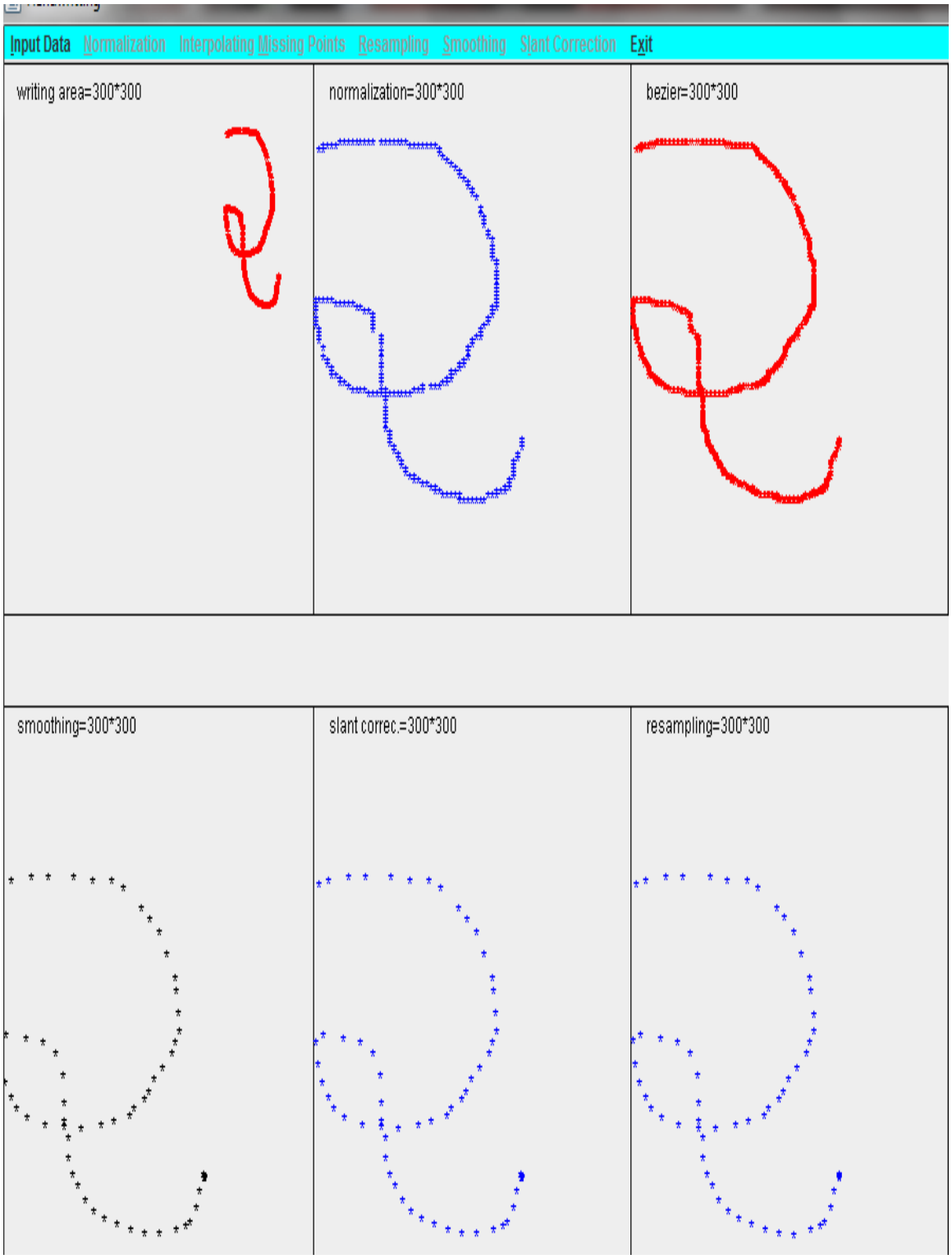

Figure 3.6 Pre processing steps

3.3 Feature Extraction

Feature extraction is a very important step and recognition accuracy largely depends upon the features extracted. Features are categorized into low level and high level features [27]. A few neighboring points determine the low level features while high level features are estimated on a larger scale as compared to low level features. High level features may include loops, crossings, straight lines etc. low level features comprise of direction, slope, slant etc.

In the present dissertation, two low level features have been extracted namely:

- Direction Angles
- Curvature

3.3.1 Direction Angles

Direction angle is the angle that the two successive coordinate points of the input handwritten character make with the x-axis in anti-clockwise direction. After the resampling done on the input character, the number of coordinate points is fixed to 64. These 64 resampled points are then used to calculate the direction angles. The direction angles are calculated from point 1 to point 2, then from point 2 to point 3 and so on. The angle for the first point of stroke sample has been considered as zero because of the non-availability of previous consecutive point. Thus, in all 64 direction angles are calculated of the 64 resampled points. In this study, direction angles are assigned numbers ranging from 1-8 depending upon the range of angles they fall into. The range of angles is illustrated in the following Table 3.1. Ranges are calculated using the 8-directional code method [39].

Table 3.1 Range of direction angles according to 8-directional code

Range of Angles	Assigned number
θ(j)>22.5 && θ(j)<=67.5	**1**
θ(j)>67.5 && θ(j)<=112.5	**2**
θ(j)>112.5 && θ(j)<=157.5	**3**
θ(j)>157.5 && θ(j)<=202.5	**4**

θ(j)>202.5 && θ(j)<=247.5	**5**
θ(j)>247.5 && θ(j)<=292.5	**6**
θ(j)>292.5 && θ(j)<=337.5	**7**
(θ(j)>337.5 && θ(j)<=360) ‖ (θ(j)>=0.0 && θ(j)<=22.5)	**8**

3.3.2 Curvature

Curvature is the amount by which a geometric object deviates from being flat, or straight in the case of a line. Curvature is calculated using the following formula of Radius of curvature:

$$\text{Radius of Curvature} = \frac{\left[1 + \left(\frac{dy}{dx}\right)^2\right]^{\frac{3}{2}}}{\frac{d^2y}{dx^2}}$$

Curvature is found by drawing a curve between three points and then drawing a curve joining those three points. Radius of curvature value of that curve is then found using the above mentioned formula. Curvature is found for points taken in triplet e.g. 1, 2 and3; then 2, 3 and 4; and so on. Total of 64 curvature values are obtained for the pre processed 64 resampled points. A value of 1000000 is assigned to the curvature in case the line joining the three points is not a curve but a straight line. The curvature for the first point of stroke sample has been considered as zero because of the non-availability of previous consecutive point.

The extracted features i.e. direction angles and curvature along with the x and y coordinates of each of the 64 resampled points are collected and stored in a .csv (Comma Separated Values) file as shown in Figure 3.7. Each row of the .csv file contains the 64 resampled point's coordinates and their respective angle and curvature values for each input handwritten character. The first column contains the x-coordinate, second contains

the y-coordinate, third contains the direction angle values ranging from 1-8 and fourth column consists the calculated curvature values. The next four columns repeat the same sequence and so on.

x4	y4	a4	c4	x5	y5	a5	c5	x6	y6	a6	c6	x7	y7	a7	c7
1068	132	3	12.94798	1065	137	3	52.68605	1060	140	3	17.65151	1056	143	2	39.4272
925	109	6	51.85349	934	105	6	149.1889	942	103	6	146.0038	953	101	6	85.89613
978	100	8	1000000	991	100	8	169.3751	1004	100	8	1968.81	1017	101	8	147.3882
950	116	2	1000000	950	126	2	1000000	950	136	2	196.5837	952	143	1	36.80957
943	124	2	1000000	943	133	2	1000000	943	140	2	144.3916	944	149	2	1000000
1005	111	3	165.2716	998	119	3	1000000	990	127	3	98.56201	983	134	3	25.03955
1090	174	4	1000000	1089	183	4	1000000	1089	190	4	68.62515	1088	196	3	46.78254
938	103	8	1000000	945	103	8	1000000	954	103	8	72.42229	964	103	8	64.37537
1011	142	1	19.53007	1017	148	8	115.2886	1026	151	8	62.77724	1034	153	8	105.394
991	104	2	10.38133	986	103	3	53.39931	980	105	3	96.62285	972	109	3	46.94902
1063	142	3	54.56348	1059	147	3	34.225	1055	151	3	29.32551	1047	156	2	160.9944
939	106	6	120.2993	948	103	6	34.99064	961	100	8	95.35648	971	101	8	48.62175
1005	100	8	1000000	1018	100	8	53.04191	1029	100	8	83.04427	1042	103	8	444.0559
955	119	4	97.95766	954	123	4	1000000	953	128	2	1000000	953	133	2	1000000
974	121	2	1000000	974	128	2	1000000	974	135	2	1000000	974	143	2	1000000
1032	107	2	172.1828	1023	110	3	68.12717	1013	114	3	82.32498	1006	118	3	32.76463
1085	163	4	1000000	1084	172	4	65.46612	1083	181	4	53.29325	1080	189	4	32.15382

Figure 3.7 Data collected in a .csv file

CHAPTER 4

RECOGNITION PROCESS

4.1 Introduction to SVM

SVM was first heard in 1992, introduced by Boser, Guyon and Vapnik. SVMs are a set of related supervised learning methods used for classification and regression [1]. They are based on the concept of decision planes that define decision boundaries. A decision plane is the one that separates between a set of objects having different class memberships. As in Figure 4.1, objects belong either to rectangles or hollow circles. Thus, there are two classes, one of rectangles and the other of circles. H3 does not separate the two classes, H1 does with a small margin and H2 with the maximum margin.

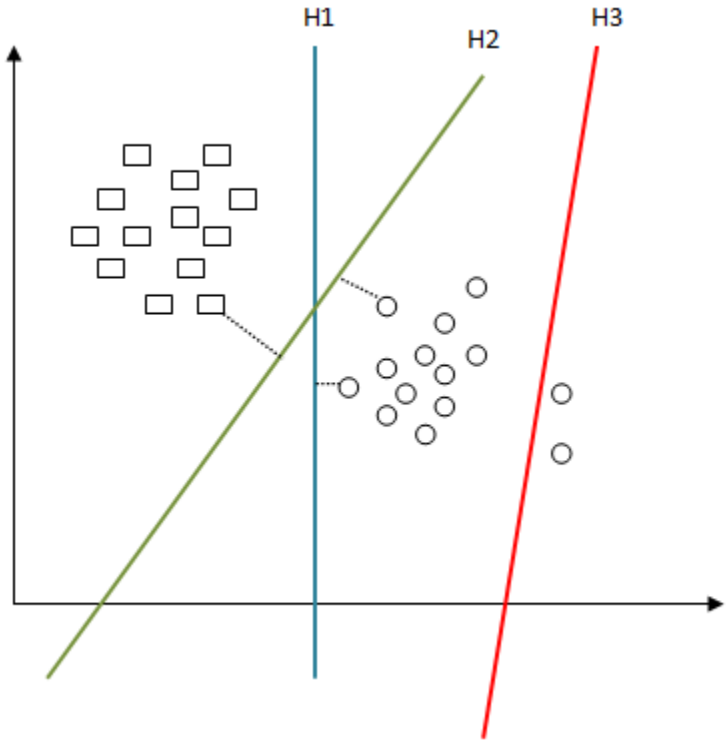

Figure 4.1 Linear Classification using decision planes

Figure 4.1 is an example of a linear classifier i.e. a classifier that separates a group of objects into their respective classes using a straight line. But most classification tasks are not that easy and require complex structures in order to correctly classify objects belonging to different classes i.e. an optimal separation. One such non-linear classification is illustrated in Figure 4.2. In this case, in order to correctly classify between objects of different classes i.e. rectangles and circles, a curve needs to be drawn rather than a straight line. Classification tasks based on drawing separating lines to distinguish between objects of different class memberships are known as hyperplane classifiers. SVMs are particularly suited to handle such tasks [31].

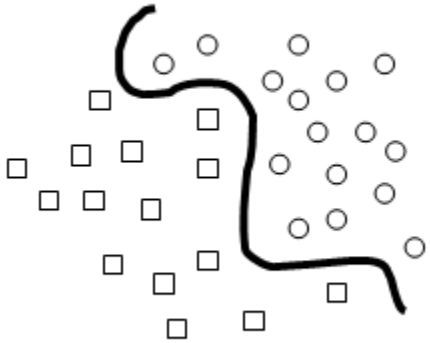

Figure 4.2 Complex classification

The basic SVM formulation is for linearly separable datasets. It can be used for non-linear datasets by indirectly mapping the nonlinear inputs into linear feature space. This mapping or re-arrangement is done using the mathematical functions also called as kernel functions of SVM. Figure 4.3 shows the basic idea behind SVM. The mapping is done in such a way that the mapped objects (right side of Figure 4.3) are linearly separable and thus, instead of constructing a complex curve (left side of Figure 4.3), all that needs to be done is to find an optimal line that can separate the objects into different classes as shown in the Figure 4.3.

SVM are binary classifiers that separate linearly any two classes by finding a hyperplane of maximum margin between the two classes. The margin means the minimal distance from the separating hyperplane to the closest data points. SVM learning machines

searches for an optimal separating hyperplane where the margin is maximal. The outcome of the SVM is based only on the data points that are at the margin and are called 'Support Vectors' [9]. This concept is illustrated in figure 4.4.

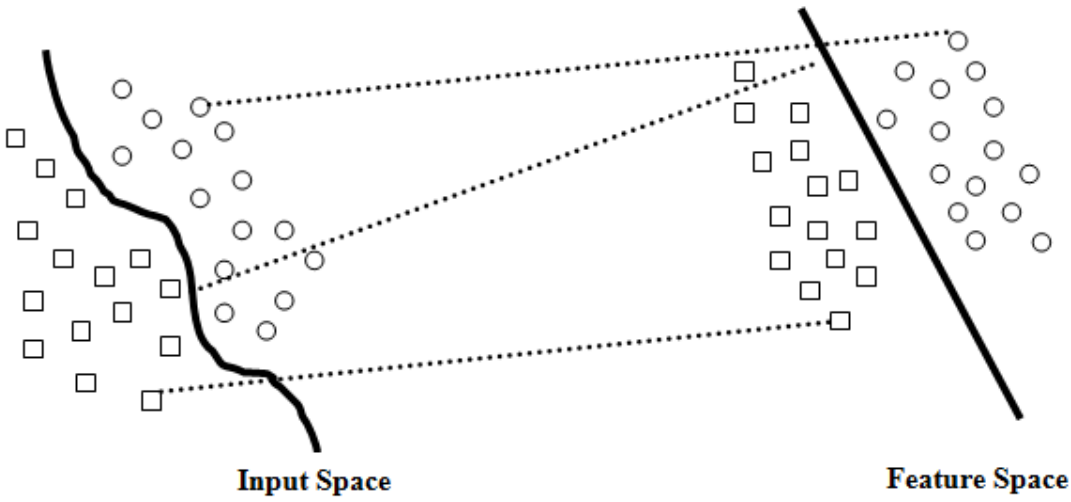

Figure 4.3 Idea behind SVM

The dashed lines in Figure 4.4 determine the mapping done from the input space to the feature space in SVM.

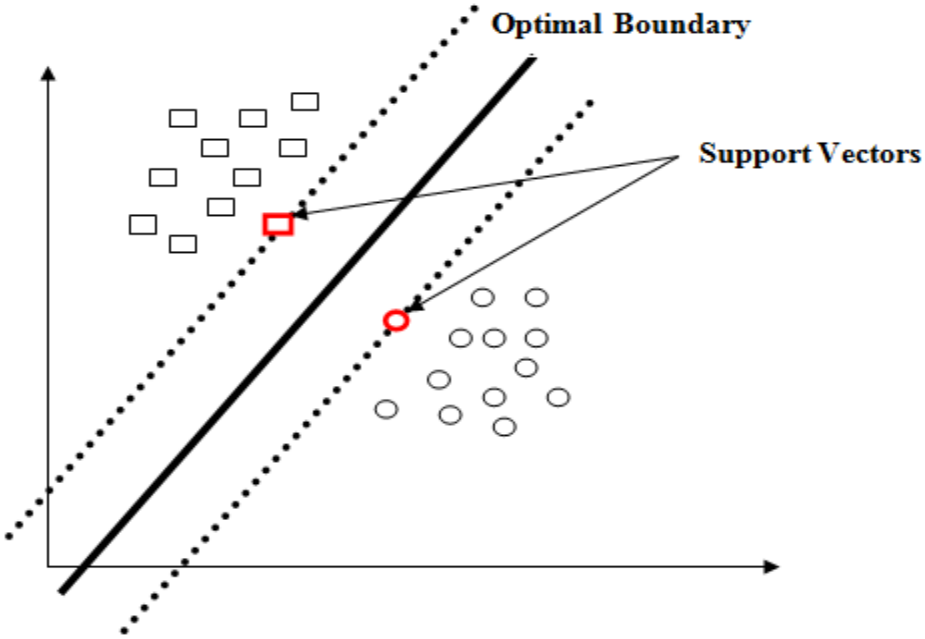

Figure 4.4 Concept of support vectors

Support vector machine classifiers have gained immense popularity in recent years providing excellent recognition results in various applications. Some of the applications of SVM are as listed below:

- Text (and hypertext) categorization
- Image classification
- Bioinformatics (protein classification, cancer classification)
- Topic drift in page ranking algorithms
- Handwritten character recognition

SVM in the present dissertation has been used for handwritten character recognition.

4.2 Recognition of Numerals

The process has been implemented on the handwritten Devanagari numeral data collected in this work. Numerals are written by ten writers. Devanagari numeral system consists of a set of ten numeral symbols as discussed in an earlier chapter. Each writer was asked to write ten samples of each numeral. Thus, 1000 samples in all are created during this work. The co-ordinate points of the input handwritten character are collected as the character is written. The input handwritten character then undergoes pre-processing and from the pre-processing phase resampled points are obtained. These resampled points are then fed to the feature extraction phase where two low level features i.e. direction angle and curvature are extracted. Together all this data i.e. x, y coordinate points, direction angle and curvature for each point of each character are stored in a .csv file. The data from this file is then fed to the next phase i.e. recognition phase. This is level one of the experimentation.

In the second level of experimentation, the data that is fed into the recognition phase is partitioned into the following six schemes:

SCHEME 1: x, y, direction angle, curvature

SCHEME 2: x, y and direction angle

SCHEME 3: x, y and curvature

SCHEME 4: direction angle and curvature

SCHEME 5: direction angle only

SCHEME 6: curvature only

In the third level of experimentation, SVM is used for the recognition process. Recognition results are obtained using four kernel functions of SVM. The four kernel functions of SVM used in this work are linear kernel, polynomial kernel, radial basis function kernel and sigmoid kernel. Brief descriptions of these kernels are given in the next section. The data collected is analyzed for each of the four kernels using each of the six schemes in which the data is partitioned. 75% of data in each of the six schemes is used for training and the rest 25% is used for testing purposes.

4.2.1 SVM Kernels

Linear kernel: Linear SVM is linearly scalable with the size of the training data set. It is given by the following formula:

$$k(x, y) = x * y$$

Where, k(x, y) is the kernel function

Polynomial kernel: It is a non-stationary kernel which is well suited for problems where the training data is normalized. It is given by the following formula:

$$k(x, y) = (\alpha x^T y = c)^d$$

Where, α is the slope, c is the constant term and d is the polynomial degree

RBF kernel: It is defined on the interval [-1, 1] and is given by the following formula:

$$k(x, y) = \exp(-\| x - y \|^2 / 2\sigma^2)$$

Sigmoid kernel: With gain κ and offset θ, the formula for a sigmoid kernel is given by:

$$k(x, y) = \tanh(\kappa(x.y) + \theta)$$

4.3 Results and Discussions

Results obtained by the recognition using SVM for all the four kernels for each of the six data schemes are depicted as below. 75% of the data is used for training the system and the rest 25% is used for testing.

a) SCHEME 1

In this scheme, all the four features i.e. x coordinate, y coordinate, direction angle and curvature are used for recognition. The results obtained by this recognition are shown in Table 4.1 and Figure 4.5 depicts these results graphically.

TABLE 4.1: Recognition accuracy with SCHEME 1

Kernel	Accuracy
Linear	**98.700**
Polynomial	**98.100**
RBF	**96.500**
Sigmoid	**10.000**

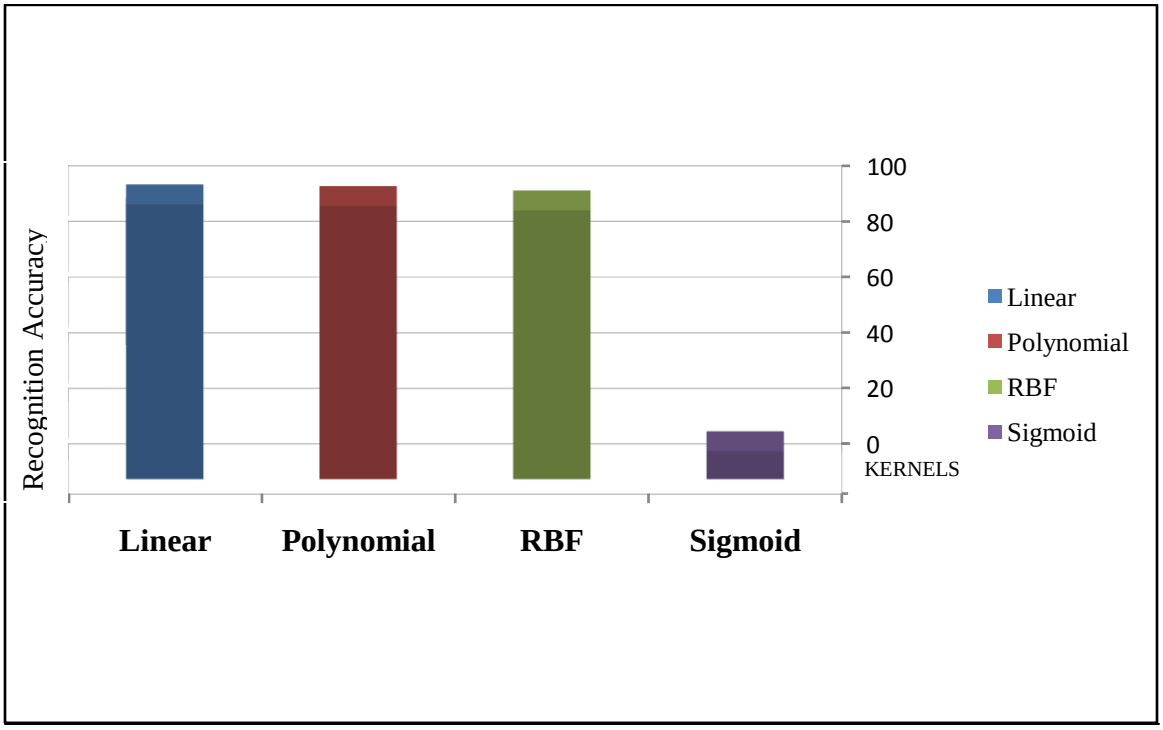

Figure 4.5 Recognition of numerals with SCHEME 1

b) SCHEME 2

In this scheme, x-y coordinate of a point along with the direction angle are considered for each point in the input handwritten character. The results obtained by this recognition are shown in Table 4.2 and Figure 4.6 graphically depicts these results.

Table 4.2: Recognition accuracy with SCHEME 2

Kernel	Accuracy
Linear	**98.900**
Polynomial	**98.100**
RBF	**98.200**
Sigmoid	**97.800**

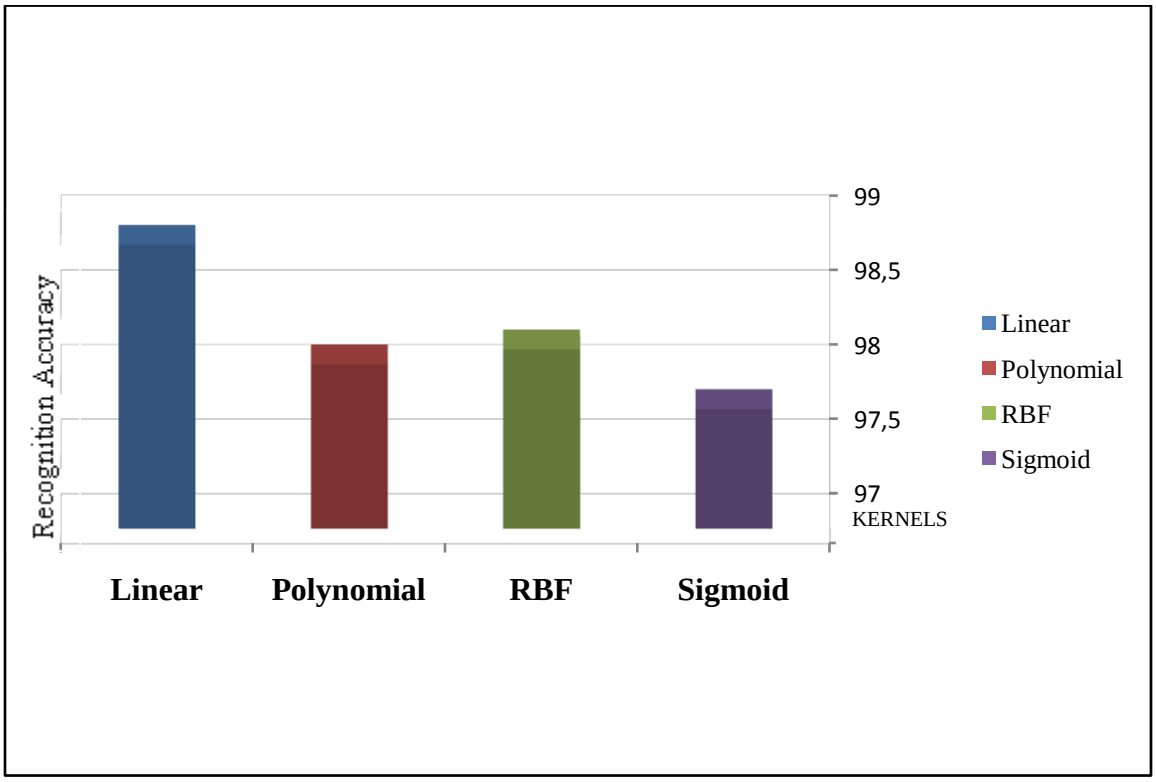

Figure 4.6 Recognition of numerals with SCHEME 2

c) SCHEME 3

In this scheme, x-y coordinate of a point along with its curvature value are considered for each point of the input handwritten character. The results obtained from this recognition are depicted in Table 4.3 and Figure 4.7.

Table 4.3: Recognition accuracy with SCHEME 3

Kernel	Accuracy
Linear	**97.400**
Polynomial	**97.400**
RBF	**96.900**
Sigmoid	**10.000**

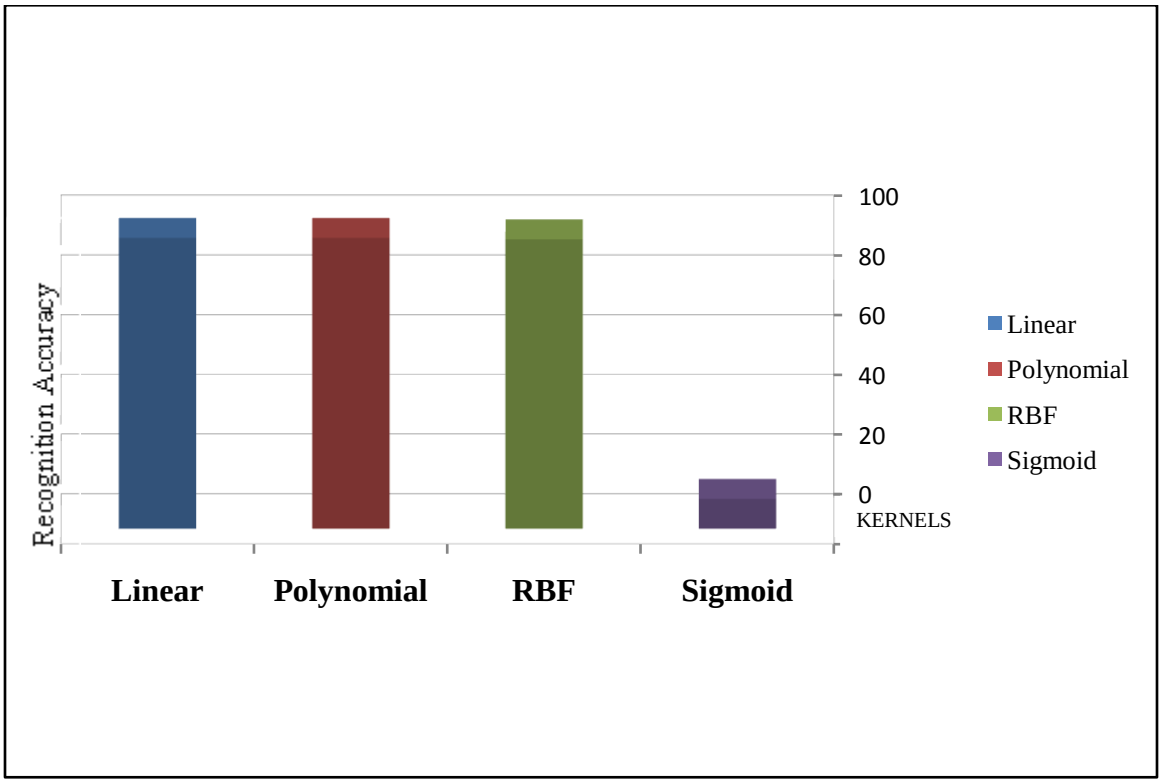

Figure 4.7 Recognition of numerals with SCHEME 3

d) SCHEME 4

In this scheme, the coordinate points are not considered. Only the direction angle and curvature values are considered from the data collected. The recognition results obtained with this scheme are illustrated in Table 4.4. Figure 4.8 depicts these results graphically.

Table 4.4: Recognition accuracy with SCHEME 4

Kernel	Accuracy
Linear	**96.400**
Polynomial	**96.400**
RBF	**96.500**
Sigmoid	**10.000**

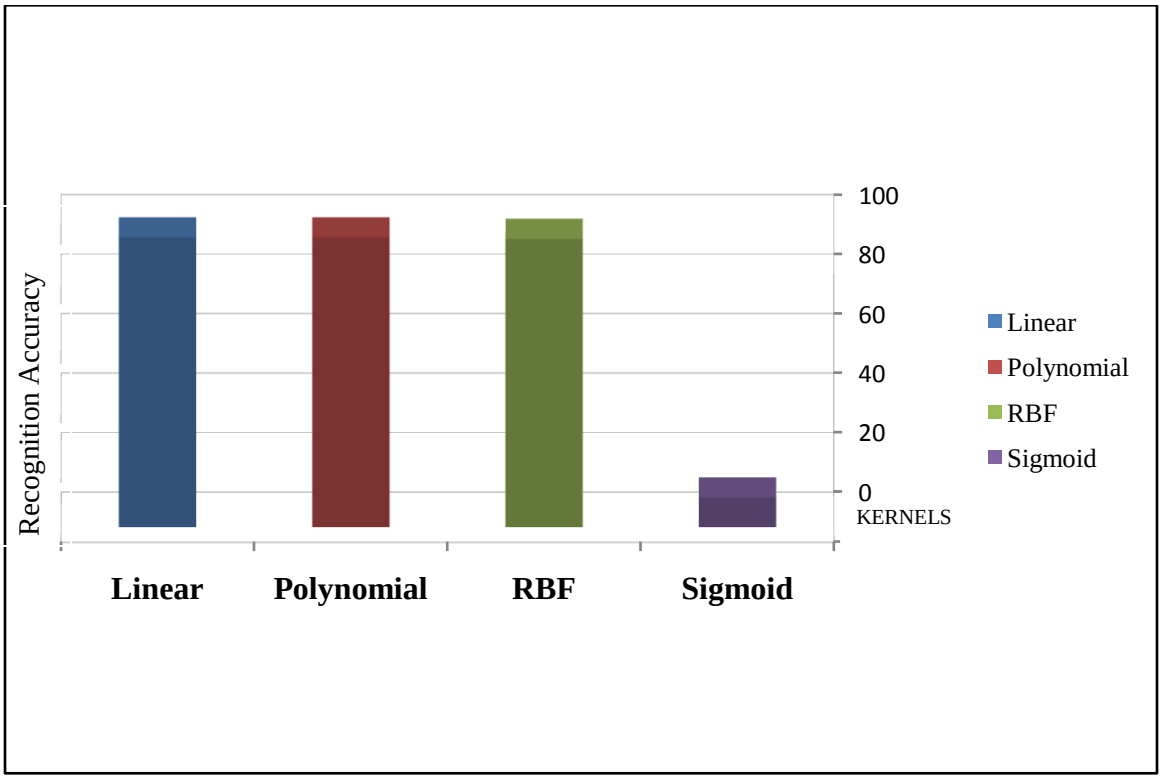

Figure 4.8 Recognition of numerals with SCHEME 4

e) SCHEME 5

In this scheme, only the direction angles are considered. Table 4.5 depicts the recognition results and Figure 4.9 illustrates the results graphically.

Table 4.5: Recognition accuracy with SCHEME 5

Kernel	Accuracy
Linear	**95.800**
Polynomial	**95.200**
RBF	**95.400**
Sigmoid	**91.700**

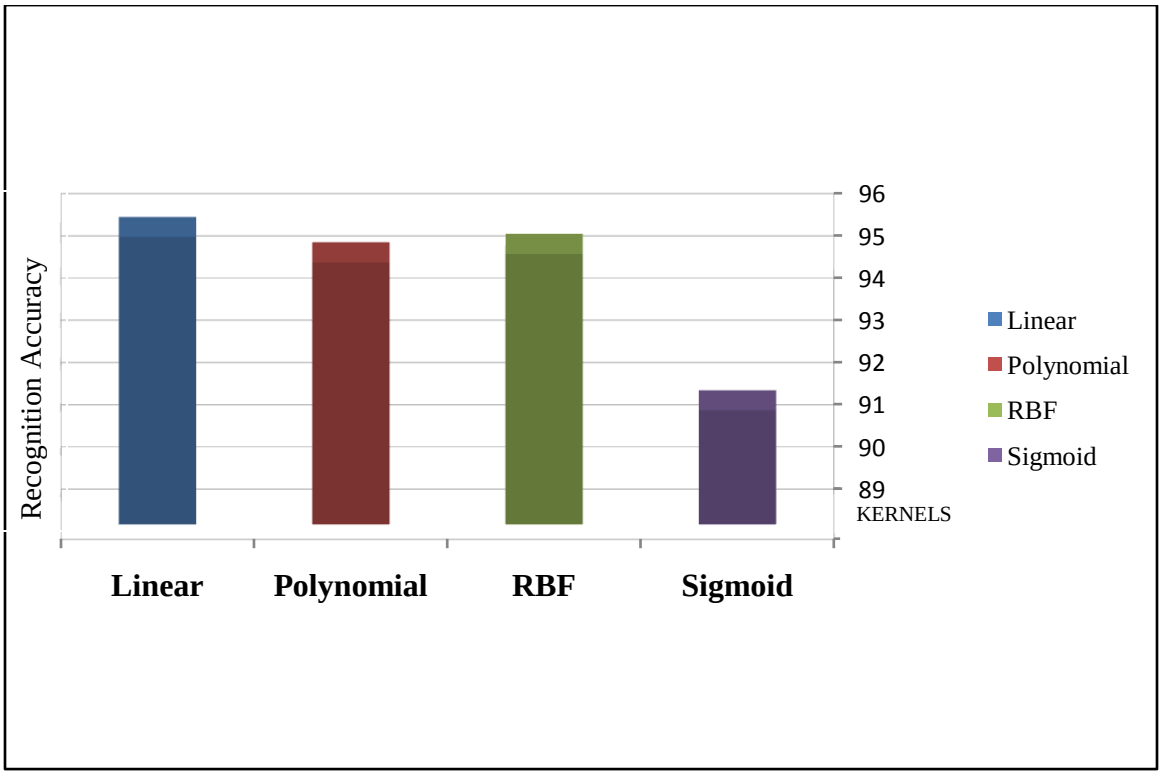

Figure 4.9 Recognition of numerals with SCHEME 5

f) SCHEME 6

In this scheme, only the curvature is considered. Table 4.6 depicts the recognition results and Figure 4.10 illustrates the results graphically.

Table 4.6: Recognition accuracy with SCHEME 6

Kernel	Accuracy
Linear	**83.800**
Polynomial	**83.700**
RBF	**72.700**
Sigmoid	**10.000**

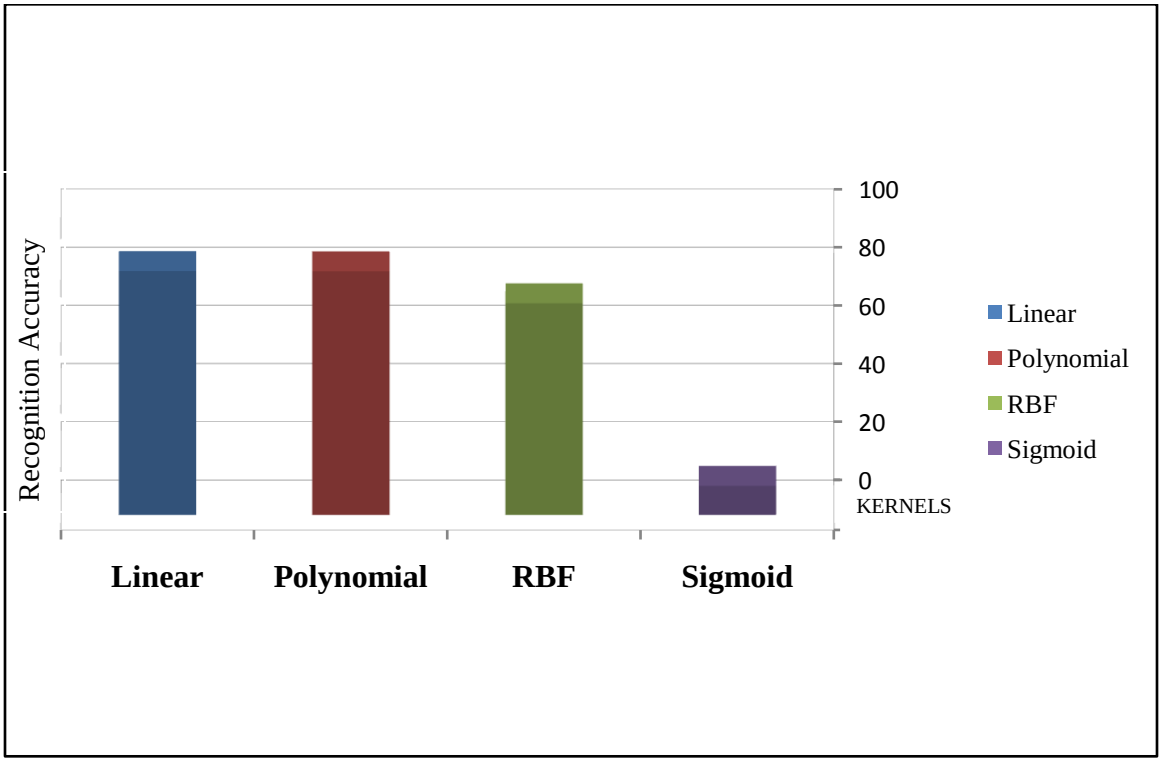

Figure 4.10 Recognition of numerals with SCHEME 6

CHAPTER 5

CONCLUSIONS and FUTURE SCOPE

The main goal of this thesis was to develop an online handwritten Devanagari numeral recognition system. This thesis describes the pre-processing, feature extraction and the recognition phase. Pre-processing and feature extraction are done prior to recognition to increase the efficiency of the character recognition system. Direction angle and curvature are the two features extracted. Recognition is done using four kernel functions of SVM by dividing the data into six schemes depending on the features extracted. Good recognition accuracies have been obtained for all the six schemes and the kernels. Results obtained are reasonably good when the linear kernel is used as compared to the other kernels. The highest accuracy shown by the linear kernel is 98.900%. The results also prove that direction angle and curvature are two very important features and enhance the recognition process. The two features showed good results even when used individually for character recognition especially the direction angles.

The present work can be extended in following directions:

- This work was based on recognition of Devanagari numerals. The work can be extended to recognize alphanumeric characters and Devanagari words in future.
- The recognition accuracies can be further improved by extracting more features from the input character.
- In the present work, direction angles are estimated using the 8-directional code method. This method can be further extended to 12-directional code method and 16-directional code method [39], in the future work. Implementing this would lead to more precision in direction angles.
- There is a lot of scope for future enhancements by implementing new preprocessing techniques such as word rotation, to aid the recognition process.
- The work can be extended by increasing the database and the number of users from which the data is collected, in the future.

REFERENCES

[1] Wikipedia, http://en.wikipedia.org/wiki.

[2] Tappert, C. C., Suen, C. Y., and Wakahara, T., 1988. Online Handwriting Recognition- A Survey. International Conference on Pattern recognition, vol. 2, pp. 1123-1132.

[3] Tappert, C. C., Suen, C. Y., and Wakahara, T., 1990. The State of the Art in Online Handwriting Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 8, pp. 787-808.

[4] Tappert, C. C, 1984. Adaptive Online Handwriting Recognition. Proceedings of International Conference of Pattern Recognition, pp. 1004-1007.

[5] Jaeger, S., Manke, S., Reichert, J., Waibel A., 2001. Online Handwriting Recognition: The Npen++ Recognizer. International Journal of Document Analysis and Recognition, vol. 3, no. 3, pp. 169-180.

[6] Kumar, A. and Bhattacharya, S., 2010. Online Devanagari Isolated Character Recognition for the iPhone using Hidden Markov Model. Proceedings of the IEEE Students' Technology Symposium, pp. 300-304.

[7] Suen, C. Y., Koerich, A. L. and Sabourin, R., 2003. Lexicon-Driven HMM decoding for large vocabulary handwriting recognition with multiple character models. International Journal of Document Analysis and Recognition, vol 6, no. 2, pp. 126-144.

[8] A. Sharma, "Online Handwritten Gurmukhi Character recognition", PhD. Thesis, Thapar University. 2009.

[9] Rajashekararadhya, S. V. and Ranjan, P. V., 2009. Support Vector Machine Based Handwritten Numeral Recognition of Kannada Script. IEEE International Advance Computing Conference, pp. 381-386.

[10] Joshi, N., Sita, G., Ramakrishnan, A. G., Deepu, V., and Madhvanath, S., 2005. Machine Recognition of Online Handwritten Devanagari Characters. Proceedings of the Eighth International Conference of Document Analysis and Recognition, vol. 2, pp. 1156-1160.

[11] Slanik, P. and Govindaraju, V., 2001. Equivalence of Different Methods for Slant and Skew Corrections in Word Recognition Applications. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 3, pp. 323-326.

[12] Mondal, T., Bhattacharya, U., Parui, S. K. and Das, K., 2010. Online Handwriting Recognition of Indian Scripts- the First Benchmark. Proceedings of twelfth International Conference on Frontiers in handwriting recognition, pp. 200-205.

[13] Huang, B. Q., Zhang, Y. B., Kechadi, M-T., 2007. Preprocessing Techniques for Online Handwriting Recognition. Proceedings of Seventh International Conference on Intelligent Systems Design and Applications, pp. 793-800.

[14] Jayaraman, A., Sekhar, C. C. and Chakravarthy, V. S., 2007. Modular Approach to Recognition of Strokes in Telugu Script. Ninth international Conference on Document Analysis and Recognition, vol. 1, pp. 501-505.

[15] Verma, B., LU, j., Ghosh, M. and Ghosh, R., 2004. A feature extraction Technique for Online Handwriting Recognition. IEEE International Conference on Neural Networks, vol. 2, pp. 1337-1341.

[16] Prasanth, L., Babu, V. J., M. D., Sharma, R., 2007. Elastic Matching of Online Handwritten Tamil and Telugu Scripts using Local Features. International Conference on Document Analysis and Recognition, vol. 2, pp. 1028-1032.

[17] Bhattacharya, U., Gupta, B. K. and Parui, S. K., 2007. Direction Code Based Features for Recognition of Online Handwritten Characters of Bangla. Proceedings of Ninth International Conference on Document Analysis and Recognition, vol. 1, pp. 58-62.

[18] Sharma, A., Kumar, R. and Sharma, R. K., 2008. Online Handwritten Gurmukhi Character Recognition using Elastic Matching. Proceedings of Congress on Image and Signal Processing, vol. 2, pp. 391-396.

[19] Swethalakshmi, H., Jayaraman, A., Chakravarthy, V. S. and Sekhar, C. C.. Online Handwritten Character Recognition o Devanagari and Telugu Characters using Support Vector machine, pp. 1-6.

[20] Mohiuddin, S., Bhattacharya, U. and Parui, S. K., 2011. Unconstrained Bangla Online Handwriting Recognition based on MLP and SVM. Proceedings of the Joint Workshop on Multilingual OCR and Analysis for Noisy Unstructured Text Data, Article. 11, pp. 1-6.

[21] Jang, B. K. and Chin, R. T., 1992. One-Pass Parallel Thinning: Analysis, Properties and Quantitative Evaluation. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 11, pp. 1129-1140.

[22] Elnagar, A., Al-Kharousi, F. and Harous, S., 1997. Recognition of Handwritten Hindi Numerals using Structural Descriptors. IEEE International Conference on Computational Cybernetics and Simulation, vol. 2, pp. 311-314.

[23] Ahmad, A. R., Khalid, M., Gaudin, C. V. and Poisson, E., 2004. Online Handwriting Recognition using Support Vector machine. IEEE Region 10 Conference, vol. 1, pp. 311-314.

[24] El Emary, I. M. M. and Hammad, M. M., 2005. On-line Structural approach for Recognizing Hand-Written Indian Numbers. ICGST-CVIP Journal, vol. 5, issue. 7, pp. 21-26.

[25] Bharath, A. and Madhvanath, S., 2007. Hidden markov model for Online Handwritten Tamil Word recognition. Proceedings of International Conference on Document Analysis and Recognition, vol. 34, issue. 4, pp. 670-682.

[26] Elanwar, R. I., Rashwan, M. A. and Mashali, S. A., 2007. Simultaneous Segmentation and Recognition of Arabic Characters in an Unconstrained Online Cursive Handwritten Document. Proceedings of World Academy of Science, Engineering and Technology, vol. 1, no. 4, pp. 203-206.

[27] Shanthi, N. and Duraiswamy, K., 2010. A Novel SVM- Based Handwritten Tamil Character Recognition System. Pattern Anal Applic, vol. 13, no. 2, pp. 173-180.

[28] Hanmandlu, M., Grover, J., Madar, V. K. and Vasikarla, S., 2007. Input Fuzzy Modeling for the Recognition of Handwritten Hindi Numerals. Proceedings of International Conference on Information technology, pp. 208-213.

[29] Huang, B. Q., Du, C. J., Zhang, Y. B. and Kechadi, M-T., 2006. A Hybrid HMM-SVM Method or Online Handwriting Symbol Recognition. Proceedings of Sixth International Conference on Intelligent System Design and Applications, vol. 1, pp. 887-891.

[30] Jain, A. K., Duin, R. P. W. and Mao, J., 2000. Statistical Pattern Recognition: A Review. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, pp. 4-37.

[31] Bellegarda, E. J., Bellegarda, J. R., Namahoo, D. and Nathan K. S., 1993. A probabilistic framework for online handwriting recognition. Proceedings of IWFHR III, pp. 225-234.

[32] Bilane, P., Youssef, E., Eter, B., Sarraf, C. and Constantin, J., 2007. Simplified Point to Point Correspondence of the Euclidean Distance for Online Handwriting Recognition.

IEEE International Conference on Signal Processing and Communications, pp. 1011-1014.

[33] Alleau, F., Poisson, E., Guadian, C. V. and Callet, P. L., 2003. TDNN With Masked Inputs. Proceedings of ICICS-PCM, vol. 2, pp. 989-993.

[34] Bharath, A. and Madhvanath, S., 2010. On the Significance of Stroke Size and Position for Online Handwritten Devanagari Word Recognition: An Empirical Study. International Conference on Pattern recognition, pp. 2033-2036.

[35] Beigi, H. S. M., 1996. Pre-Processing the Dynamics of On-line Handwriting data, Feature Extraction and Recognition. International Workshop on Frontiers of Handwriting recognition, pp. 1-4.

[36] Connell, S. D., Sinha, R. M. K. and Jain, A. K., 2000. Recognition of Unconstrained On-Line Devanagari Characters. IEEE International Conference on Pattern recognition, vol. 2, pp. 368-371.

[37] Bahlmann, C., Haasdonk, B. and Burkhart, H., 2002. Online Handwriting Recognition with SVM- A Kernel Approach. Proceedings of the eighth International Workshop on Frontiers in handwriting Recognition, pp. 49-54.

[38] Mahdy, Y. B. and El-Melegy, M. T., 1996. Encoding Patterns for Efficient Classification by Nearest Neighbor Classifiers and Neural Networks with Application to Handwritten Hindi Numeral Recognition. Proceedings of ICSP, vol. 2, pp. 1362-1365.

[39] Ding, Y., Kimura, F. and Miyake, Y., 2004. Slant Estimation for Handwritten Words by Directionally Refined Chain Code. University of Groningen Archive, pp. 1-10.

LIST OF PUBLICATIONS

"Online handwriting Recognition of Hindi Numerals using SVM". Deepika Wadhwa, Karun Verma. International Journal of Computer Applications, Accepted, June 2012.