Recognition of Online Handwritten Gurmukhi Strokes using Support Vector Machine

A Thesis

Submitted in partial fulfillment of the requirements for the award of the degree of

Master of Technology

Submitted by

Rahul Agrawal

(Roll No. 601003022)

Under the supervision of

Dr. R. K. Sharma

Professor

School of Mathematics and Computer Applications

Thapar University

Patiala

School of Mathematics and Computer Applications

Thapar University

Patiala – 147004 (Punjab), INDIA

June 2012

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, **"Recognition of Online Handwritten Gurmukhi Strokes using Support Vector Machine"**, in partial fulfillment of the requirements for the award of degree of Master of Technology in **Computer Science and Applications** submitted in School of Mathematics and Computer Applications of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of Dr. R.K. Sharma and refers other researcher's work which are duly listed in the reference section.

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

(Rahul Agrawal)

Roll No.: 601003022

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

12.6.12

(Dr. R. K.Sharma)
Professor, SMCA
Thapar University,Patiala

Countersigned:

(Dr. S.S. Bhatia)
Head,
School of Mathematics & Computer Applications
Thapar University,
Patiala

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

ABSTRACT

Pen-based interfaces are becoming more and more popular and play an important role in human-computer interaction. This popularity of such interfaces has created interest of lot of researchers in online handwriting recognition. Online handwriting recognition contains both temporal stroke information and spatial shape information. Online handwriting recognition systems are expected to exhibit better performance than offline handwriting recognition systems. Our research work presented in this thesis is to recognize strokes written in Gurmukhi script using Support Vector Machine (SVM). The system developed here is a writer independent system.

First chapter of this thesis report consist of a brief introduction to handwriting recognition system and some basic differences between offline and online handwriting systems. It also includes various issues that one can face during development during online handwriting recognition systems. A brief introduction about Gurmukhi script has also been given in this chapter In the last section detailed literature survey starting from the 1979 has also been given.

Second chapter gives detailed information about stroke capturing, preprocessing of stroke and feature extraction. These phases are considered to be backbone of any online handwriting recognition system.

Recognition techniques that have been used in this study are discussed in chapter three. In this chapter Support Vector Machine is discussed and two cross validation techniques namely holdout and *k*-fold have been discussed.100 words that are used in this study are also given in this chapter.

Chapter 4 contains the results that have been calculated after applying Support Vector Machine in MATLAB for three different writers. These results have been calculated for each partitioning techniques. For holdout 70%, 60% and 50% training sets have been used for stroke recognition. In *k*-fold technique 10-fold has been used. Both of these partitioning

techniques have been used for both non preprocessed and preprocessed strokes. Also results for recognition of 1000, 2000 and 3000 preprocessed strokes have been shown in this chapter. Last chapter of this work concludes all the work that has been done and gives some directions in which more work can be done to improve the recognition system.

ACKNOWLEDGEMENTS

My sincere thanks to all the people around me who helped me in completing this thesis work.

First, I wish to thank **Dr. R. K. Sharma** (Professor) of School of Mathematics and Computer Applications, Thapar University, Patiala for giving me an opportunity to work under his guidance. His continued support, guidance and vision helped me to complete this thesis. It has been a pleasure working under his guidance.

I truly appreciate cooperation and support received from my friends Mayank Gupta, Ajay Sharma, Poonam Sharma, Nainsi Gupta, Preeti Singhal, Rajan Walecha and Virendra Mishra during this work.

I also express my sincere gratitude to all the faculty members at **THAPAR UNIVERSITY** for equipping me with the best of knowledge and providing me top class facilities and infrastructure.

Date: 11th June 2012

Place: Thapar University, Patiala.

(Rahul Agrawal)

LIST OF FIGURES AND GRAPHS

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviations	Expanded Forms
PC	Personel Computer
HWAI	Handwritten Address Interpretation System
HMM	Hidden Markov Model
RMS	Root Mean Square
SVM	Support Vector Machine
MATLAB	Matrix Laboratory
SRM	Structural Risk Minimization
SVC	Support Vector Classifier
SVR	Support Vector Regeressor

CONTENTS

CHAPTER 1

INTRODUCTION

Whenever we talk about the interaction between human beings and computers the devices which help in interaction are monitor, keyboard, printer, mouse and any pointing device. But nowadays the concept has totally changed. Day by day computers are available in more and more compact form, which are more convenient for the user. In earlier days, user took a lot of time to enter data in the computer through keyboard and we had to accommodate keyboards in the smaller computers. Same is for the output devices that were of huge size. But now both input and output processes can be done on a single compact device, where a user gives his input with the help of an electronic pen and found the output simultaneously. This whole thing makes it very convenient for the user to operate these systems. But sometimes, whenever the data is in textual format, it becomes problematic for the user to use an electronic pen. So for providing solution for such kind of problems handwriting recognition feature is introduced which makes it very much easier for the user to input a character without using a keyboard.

Handwriting recognition is the most exciting and empirical research for image processing and pattern recognition. There are many applications that are included in this handwriting recognition like reading aid for blind, bank cheques and conversion of any handwritten document into text form. There is an extensive contribution given by the handwritten character recognition in the advancement of high tech process and it also improves the interface section which is held between a human beings and a computer in a number of applications. Several research works have been focusing on different methodologies in an attempt to reduce the processing time while simultaneously improving recognition accuracy.

The classification of handwriting recognition can be done in two major categories.
1. Offline handwriting recognition - Under such handwriting recognition, the writing is

available as an image, it can be done so with the help of scanner, which captures the writing optically.

2. Online handwriting recognition - Under such handwriting recognition, the two dimensional coordinates of successive points are represented as a function of time and the order of strokes made by the writer are also available. The online methods have been shown to be superior to their offline counterparts in recognizing handwritten characters due to the temporal information available with the former.

Nowadays, for devices like personal digital assistant or tablets, online handwriting recognition has become a useful feature. With this feature the demand of tablets increase as per the rate of their use has increase.

The presence of online handwriting recognizer for Devanagiri, Gurmukhi, Bangla and other Asian scripts shall provide a natural way of communication between users and computers and it will increase the usage of personal digital assistant or tablet PCs in Indian languages. Also, some of the Asian scripts such as Devanagiri, Gurmukhi, Bangla and Tamil share many similarities and therefore advances made for one script with respect to online handwriting recognition could be useful for other such similar scripts (Sharma, 2009).

1.1 Handwriting recognition system

Handwriting recognition system is the activity through which user transmit the handwritten data into the computer through the sources of scanner, touch screens and other devices, where computer interpret that input into its language. The mode of such input can be paper documents, photograph etc.

As we have discussed in the above paragraph that handwriting recognition system deals with the acquiring of data through various methods, so on this acquiring only the classification of handwriting recognition system depends to use online character recognition systems and offline character recognition systems. As it include a set of characters, the numerous writing style variations and the constraints on the writing have a great influence on recognition methods.

1.1.1 Offline character recognition system

Offline character recognition includes a process for this transmission of input into the machine. Firstly it includes recognition of text from originally written paper. Further for this system paper document has to be digitized for getting two dimensional images. Features for

recognition are then first enhanced and then extracted from the bitmap image by means of digital image processing. This type of recognition is called offline recognition. So here it is understood that in offline character recognition whatever the text is written is not recognized at the same time when it is produced, but after the completion of writing task. So as per the researchers, offline recognition does not provide the facility of real time activity. Text is converted automatically in offline handwriting recognition in the form of an image into some letter codes which can be use by computer on any time. The data obtained by this form is regarded as a static representation of handwriting. Several problem areas exist where offline handwriting recognizers can be of use because of large quantities of handwritten data.

1.1.2 Online character recognition system

Under online handwriting recognition system the whole task is dependent on the application which we are using for the transmission of input into the electronic tablets. In online handwriting recognition system, text is input with special equipment and it is recognized in real time. Input media is usually a tablet that can capture information of the location and the motion of a pen or some other pointing device moving on its surface. Writing can be done with any ordinary pen or a pen like device specially designed for its purpose. Location and pressure data are frequently sent to the computer which performs the recognition task.

Online handwriting recognition assumes that a device is connected to the computer and is available to the user. The device converts the user's writing motion into a sequence of signals and sends the information to the computer. The most common form of the device is a tablet digitizer. A tablet consists of a plastic or electronic pen and a pressure or electro static sensitive writing surface on which the user forms one's handwriting with the pen. When moving a pen, the digitizer is able to detect information like x and y coordinates of a point, the state of whether the pen touches the surface or not. The information is sent to the connected computer for recognition process as given in Figure 1.1 (Oh, 2001). A "stroke" in online data is defined as sequence of sampled points from the pen-down state to the pen-up state of the pen, and the completed writing of a character consists of a sequence of one or more strokes. A" digital ink" is the display of the strokes on the computer screen. By digital ink, the user can see what he or she writes on the tablet and it provides interactivity between the user and the computer.

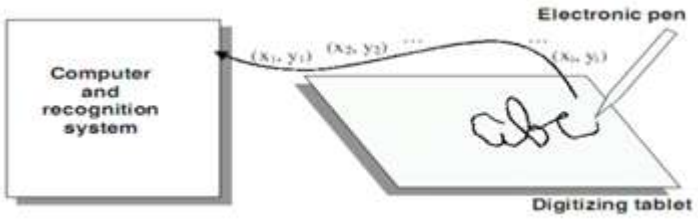


Figure 1.1: A tablet digitizer, input sampling and communication to the computer

Online handwriting system in real time collects through handwriting pad, not only the position of stroke, but also the dynamic information of velocity, writing pressure, *etc*. A typical online handwriting system should include four main parts:

1. Data acquisition: It is the first phase in online handwriting recognition that collects the sequence of coordinate points of the moving pen.

2. Preprocessing: Preprocessing phase in handwriting recognition is applied to remove noise or distortions present in input text due to hardware and software limitations. In online handwriting recognition, preprocessing includes five common steps, namely, size normalization and centering, interpolating missing points, smoothing, slant correction and re sampling of points.

3. Feature extraction: Feature extraction is essential for efficient data representation and for further processing. Also, high recognition performance could be achieved by selecting suitable feature extraction method.

4. Recognition: According to extracted features, compared with reference stroke, decision can be made through some kind of decision rules.

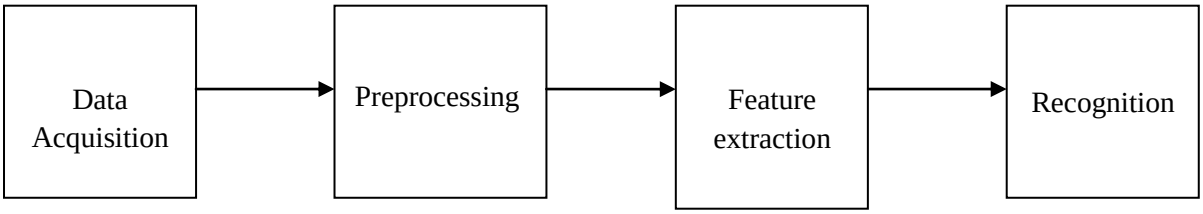


Figure 1.2: Phases of online handwriting recognition

1.1.3 Advantages of online handwriting recognition over offline handwriting recognition

- Online handwriting recognition means that the machine recognizes the writing while the user writes. The term real time or dynamic has been used in place of online. Offline handwriting recognition, by contrast, is performed after the writing is completed. It can be performed days, months, or years later.
- An advantage of online devices is that they capture the dynamic information of the writing. This information consists of the number of strokes, the direction of the writing for each stroke, and the speed of the writing each stroke. Online transducers capture the trace of the handwriting or line drawing as a sequence of coordinate points. By contrast, offline conversion of scanner data to line drawings usually requires costly and imperfect preprocessing to extract contours and to thin.
- Another advantage of online over offline data is the availability of the stroke segmentation and order of writing.
- In dynamic handwritten data, the trace of the writer's pen is stored as a sequence of points sampled at equally spaced time intervals. The information captured for each sample is the (x, y) coordinates of the pen on the digitizing tablet. While this pen trace could be used to construct a static image of the writing, thus allowing traditional optical character recognition techniques to be applied.
- Another advantage of online recognition is interactivity .In an editing application, for example, the writing of an editing symbol can cause the display to change appropriately. Also, recognition errors can be corrected immediately.
- Another advantage is adaptation. When the user sees that some of the characters are not being accurately recognized, they can alter their drawing to improve recognition. Thus, the user adapts to the recognition system.

1.2 Introduction to Gurmukhi script

The word 'Gurmukhi' literally means "from the mouth of the guru". Gurmukhi script is used primarily for the Punjabi language, which is the world's 14^{th} most widely spoken language. The people speaking Punjabi are not only confined to north Indian states such as Punjab,

Haryana and Delhi but are spread over all parts of the world. There is rich literature in this language in the form of scripture, books, poetry. It is, therefore, important to develop online handwriting recognition for such a rich and widely used language which may find many practical uses in various areas. Some of the major properties of the Gurmukhi script are:

Modern Gurmukhi has 41 consonants (vianjans), 9 vowel symbols (laga or matras), 2 symbols for nasal sounds, 1 symbol which duplicates the sound of any consonant, 3 subjoined forms of the consonants R (Ra) , H (Ha) and ੲ (Va).

Consonants

The Gurmukhi alphabet contains thirty five distinct letters. Table 1.1 shows first three letters are unique because they form the basis for vowels. Apart from era, these characters are never used on their own (Gurmukhi information).

Table 1.1: Unique vowel characters

ੳ	ਅ	ੲ
Ura	Era	Iri

In Table 1.2 character set of Gurmukhi is shown.

Table1.2: Character set of Gurmukhi script

ਸ	ਹ	ਕ	ਖ	ਗ	ਘ
Sussa Sa	Haha Ha	Kukka Ka	Khukha Kha	Gugga Ga	Ghugga Gha
ਙ	ਚ	ਛ	ਜ	ਝ	ਞ
Ungga Nga	Chucha Ca	Chhuchha Cha	Jujja Ja	Jhujja Jha	Yanza Nya
ਟ	ਠ	ਡ	ਢ	ਣ	ਤ
Tainka Tta	Thutha Ttha	Dudda Dda	Dhudda Ddha	Nahnha Nna	Tutta Ta
ਥ	ਦ	ਧ	ਨ	ਪ	ਫ

Thutha Tha	Duda Da	Dhuda Dha	Nunna Na	Puppa Pa	Phupha Pha
ਬ	ਭ	ਮ	ਯ	ਰ	ਲ
Bubba Ba	Bhubba Bha	Mumma Ma	Yaiyya Ya	Rara Ra	Lulla La
ਵ	ੜ				
Vava Va	Rahrha Rra				

In Table 1.3, six consonants created by placing a dot (bindi) at the foot (pair) of the consonant are shown.

Table 1.3: Six special consonants in Gurmukhi

ਸ਼	ਖ਼	ਗ਼	ਜ਼	ਫ਼	ਲ਼
Shusha pair bindi Sha	Khukha pair bindi Khha	Gugga pair bindi Ghha	Zuzza pair bindi Za	Fuffa pair bindi Fa	Lulla pair bindi Lla

Vowels

Gurmukhi follows similar concepts to other Brahmi scripts and as such, all consonants are followed by an inherent sound. This inherent vowel sound can be changed by using dependent vowel signs which attach to a baring consonant. In some cases, dependent vowel signs cannot be used at the beginning of a word or syllable for instance and so an independent vowel character is used instead. Table 1.4 shows list of dependent vowels.

Table 1.4: Dependent vowels list

◌	◌ਾ	◌ਿ	◌ੀ	◌ੇ	◌ੈ
Mukta *a*	Kanna *aa*	Sihari *i*	Bihari *ii*	Lavan *ee*	Dulavan *ai*

◌ੁ	◌ੂ	◌ੋ	◌ੌ		
Onkar *u*	Dulankar *uu*	Hora *oo*	Kanaura *au*		

Dotted circles represent the barer consonant. Vowels are always pronounced after the consonant they are attached to. Thus, sihari is always written to the left, but pronounced after the character on the right. List of independent vowels is shown in Table 1.5.

Table 1.5: Independent vowels list

ਅ	ਆ	ਇ	ਈ	ਏ	ਐ
A	*Aa*	*I*	I*i*	E*e*	*Ai*
ਉ	ਊ	ਓ	ਅੌ		
U	*Uu*	*Oo*	A*u*		

A word in Gurmukhi script can be partitioned into three horizontal zones. The upper zone denotes the region above the head line, where vowels reside, while the middle zone represents the area below the head line where the consonants and some sub-parts of vowels are present. The middle zone is the busiest zone. The lower zone represents the area below middle zone where some vowels and certain half characters lie in the foot of consonants. These zones are shown in Figure 1.3 (Sharma, 2009) for the Gurmukhi word "Gurmukhi".

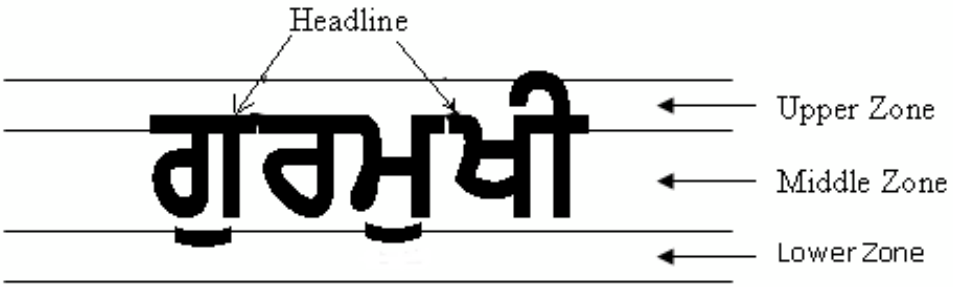


Figure 1.3: Three zones and headline in Gurmukhi word

1.3 Issues in recognition

Initially, handwriting recognition does not appear to be a difficult problem. A recognition system should just choose the correct answer, usually the one that most resembles the written one, from a limited set of characters. Unfortunately, this approach faces a number of difficulties. The most important problem in handwriting recognition is the large variation in personal writing styles. There are also a lot of variations within the writing style of one person. These variations depend for example on the context of the writing, writing equipment, writing situation, and the mood of the writer. The writing style may also change with time or practice. The performance of the automatic recognition system thus depends heavily on how well the different personal writing styles and their variations are modeled. A recognition system should be insensitive to meaningless variations and still be able to distinguish different but sometimes very similar looking characters.

1.3.1 Styles of handwriting: printed vs. cursive

The difficulty of handwriting recognition varies greatly with different writing styles. Figure 1.4 (Bellegarda e*t al.*, 1994) illustrates different writing styles in English. The writing style of the first three lines is commonly referred to as printed or discrete handwriting, in which the writer is told to write each character within a bounding box or to separate each character.

The writing style of the fourth line is commonly referred to as pure cursive or connected handwriting, in which the writers are told to connect all of the lower case characters within a word. Most people write in a mixed style, a combination of printed and cursive styles, similar to the writing on the fifth line.

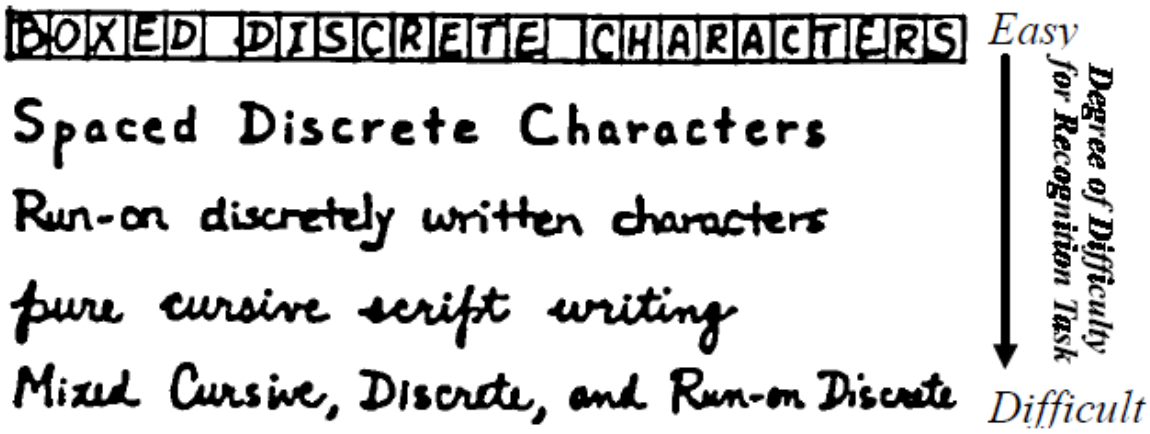


Figure 1.4 Types of writing styles

Both printed and cursive handwriting recognition are difficult tasks because of the great amount of variability present in the online handwriting signal. The variability is present both in time and signal space. Variability in time refers to variation in writing speed, while variability in signal space refers to the shape changes of the individual characters. It is rare to find two identically written characters. The difficulty of recognizing handwriting lies in constructing accurate and robust models to accommodate the variability in time and feature space.

In addition to these two types of variability in time and signal space, cursive handwriting has another type of variability in time which makes this task even more difficult. This additional type of variability is due to the fact that no clear inter-character boundaries (where one character starts or ends) exist. In printed handwriting, a pen-lift defines these boundaries between characters. However, in cursive handwriting the pen lift cues simply do not exist. Cursive-style handwriting recognition is more difficult because the recognizer has to perform the error-prone step of character segmentation, either explicitly or implicitly.

1.3.2 Writer-dependent vs. writer-independent

A writer-independent system is capable of recognizing handwriting from users, whose writing is not used for training the system. Humans are capable of writer-independent recognition. The writer-independent systems are more difficult to construct because of the variability in handwriting across writers. For writer-dependent tasks, the system is only required to learn a few handwriting styles. On the other hand, for writer-independent tasks, the system must learn invariant and generalized characteristics of handwriting.

1.3.3 Closed-vocabulary vs. open-vocabulary

Vocabulary is also a major factor in determining how difficult a handwriting recognition task is. Closed-vocabulary tasks refer to recognition of words from a predetermined dictionary. Open-vocabulary tasks refer to recognition of any words without the constraint of being in a dictionary.

Closed-vocabulary tasks are easier than open-vocabulary ones because only certain sequences of letters are possible when limited by a dictionary.

Closed-vocabulary tasks using a small dictionary are especially easy because:

1. A small vocabulary size can mean a smaller number of word pairs that can be confused.

2. A small vocabulary size enables the direct modeling of individual words, whereas a large vocabulary size needs the modeling of letters. With the usage of letters for large vocabulary tasks, the search space of all possible sentences is usually much larger due to an increase in the number of nodes in the search graph. When letters are used for modeling instead of words, the number of nodes is $m \times n$ instead of n where n is the number of words, and m is the average number of letters per word. As the vocabulary size increases, the occurrence of out-of-vocabulary words is less frequent. Thus, the performance of the large vocabulary tasks is approximately the same as of the performance of the open-vocabulary tasks.

1.4 Literature Survey

The basic idea behind doing this literature survey is to understand the inception of online handwriting recognition for different scripts and to get an idea of different recognition techniques adopted by different researchers from all over the world.

In 1915 U.S. patent a handwriting recognition user interface with stylus. After that Tom Dimond introduced Stylator tablet for computer input and handwriting recognition, but after a few years in 1961 a better tablet named RAND tablet was invented and became popular for computer recognition of script handwriting in 1962. After that an advanced system called GRAIL system was introduced in 1969 for handwriting recognition with electronic ink display. It takes input through gesture commands. In early 1980s, retail handwriting-recognition systems pencept and CIC both offered PC computers for the consumer market using a tablet and handwriting recognition instead of a keyboard and mouse. In 1989 portable handwriting recognition computer GRiDPad was introduced. First handwritten address interpretation system(HWAI) was deployed by united states postal service in 1997 and latest in 2007 first automatic writer recognition system CEDAR-FOX was launched in market.

This literature survey has been done chronologically and advancements in the online as well as offline handwriting recognition system have been reviewed.

Farag (1979) used chain codes and markov chains for word-based recognition. A recognition rate of 100% is reported for ten cursive words and one writer.

Character recognition is reviewed in Suen *et al.* (1980) saw three new emphases and directions are crucial to future breakthroughs in handprint recognition. These are:

- Development of a standard set of handprint models which could be comfortably followed by human beings to produce samples easily recognized by machines.
- Systematic selection of distinctive features and optimal combination of recognition techniques to strengthen the discriminant power of machines.
- Application of grammatical, linguistical, and contextual information to resolve ambiguities of characters which have similar shapes.

Tappert (1982) employed dynamic programming and compared the input data with letter prototypes point by point. Each data point is characterized by two parameters, the vertical position and the tangent of the pen path at the current point. Experiments involved careful writing and user-dependent databases of letter prototypes. An average recognition rate of 97% is reported for three users writing 30 words each. It should be noted that author stresses the need for careful writing and user-defined prototypes.

Brown and Ganapathy (1983) used feature vectors and an estimate of the length of the word to represent the word characteristics. The recognition is based on the extracted feature vectors using the k-nearest neighbor method. In the test, the recognition domain consisted of 43 words. Three users wrote ten samples, each containing 22 words. The recognizer was trained on data of one of the users and tested on data of the other two. Recognition rates ranged from 63.2% to 80.3%.

Impedovo (1984) proposed a method in which curves of unknown characters are matched against those of prototype characters. The curves matched are usually functions of time, such as coordinates, angular variations of the pen path and curvatures. Curve matching is a signal processing method.

Kurtzberg (1987) proposed a method to recognize unconstrained handwritten discrete symbols based on elastic matching against a set of prototypes generated by individual writers. He also introduced feature analysis with elastic matching to eliminate unlikely prototypes.

A tutorial by Rabiner (1989) presents introduction to HMMs. In this tutorial, use of HMMs has been explained with respect to speech and cursive handwriting. The reference of these tutorials has been noted in most of the research work in HMMs after 1990.

Noubound and Plamondon (1991) used structural approach to recognize online handwritten characters. They presented a real-time constraint-free hand printed character recognition system based on a structural approach. After the preprocessing operation, a chain code is extracted to represent the character. The classification is based on the use of a processor dedicated to string comparison.

Veltman and Prasad (1994) used HMM to isolated online handwritten characters and achieved an average error rate of 6.9% over fully unconstrained alphabet consisting of the lowercase English alphabet.

Powalka *et al.,* (1994) developed a word-based recognizer which used a very limited set of features consisting of a sequence of ascenders and descenders and an estimate of word length. A fuzzy logic based matching algorithm is used. Average recognition rates obtained for a 200 word lexicon was 60.6% .Handwriting of 18 users was evaluated, each writing 200 words.

Bontempi and Marcelli (1994) presented a method based on a genetic algorithm used as the engine of a learning system to produce prototypes of the characters, and on a string matcher to perform the classification. The learning mechanism, provided by a genetic algorithm, allowed the system to have both a writer independent core and an adaptation scheme to finely tune the recognizer to the writer's style. The system was tested over 10 subjects, different from the ones used to obtain the training set, who were required to write, in different times, two sets of numerals, uppercase and lowercase letters. A recognition rate of 83.4% was obtained, with a reject rate of 14.7% and an error rate of 1.9%.after adapting writer, the system was able to exhibit a recognition rate of 94.8% on that writer. By running again the second experiment with the new system, a recognition rate of 92.7% was obtained, with a rejection rate of 6.5% and an error rate of 0.8%, showing that, by adapting to that writer, the system increased its overall performance.

Duneau and Dorizzi (1994) presented a system dedicated to the recognition of English cursive words drawn on a digitizing tablet. This system uses an analytical approach in the sense that it tries to localize the letters of the word to be recognized. The system in a monoscriptor context has recognition rate min-average-max of 89.8-93.2-95.6% respectively,

and is the case of multi-scripter context rate was 93.5% for 15432 prototypes and 91.4% for 5625 prototypes, but the system is too slow and heavy during multi-scripter context, and in the case of omniscriptor recognition rate min-average-max of 60.8-75.8-92.2%.

Scattolin (1995) developed an online handwritten recognition system based on nearest neighbor classifier. He further added weights to model and found improvement in recognition, training rate was 93.7% with 99.79% reliability, and testing rate was 81.35% with 98.35% reliability.

Hu *et al.* (1996) used HMM in writer independent online handwriting recognition system using invariant and segmental features.

Rigoll *et al.* (1996) compared continuous and discrete density HMMs for cursive handwriting recognition. They obtained 70% word recognition rate for a challenging large vocabulary, writer-independent sentence input task.

Senior and Nathan (1997) used HMM with respect to writer adaption system. The initial performance of the writer-independent model varies from writer to writer. The average word error rate is 28.7%, but several writers have around 10% errors, and one has 60% errors.

Li and Yeung (1997) presented an approach to online handwritten alphanumeric character recognition based on sequential handwriting signals. In this approach, an online handwritten character is characterized by a sequence of dominant points in strokes and a sequence of writing directions between consecutive dominant points. The overall recognition rate is 91.0%, with 7.9% substitution rate and 1.1% rejection rate, and the average time for processing one data file of 20 numerals is about 6 seconds while that for 52 English letters is about 18 seconds.

Cho (1997) presented three sophisticated neural network classifiers to solve complex pattern recognition problems that include multiple multilayer perceptron classifier, HMM multi layer perceptron hybrid classifier and structure adaptive self-organizing map classifier. This work was related to unconstrained handwritten numerals.

Kimura *et al.* (1997) presented a two-stage hierarchical system consisting of a statistical pattern recognition module and artificial neural network to recognize a large number of categories including similar category sets. This work was related to hand printed Kanji

characters. The correct recognition rates of the samples are 99.09% and 98.59% for the training data and the test data, respectively.

Wakahara and Odaka (1997) proposed a distant tolerant stroke matching method that uses stroke based affine transformation. Stroke based affine transformation transforms each stroke of an input pattern to yield the best match with reference patterns as a kind of flexible shape matching. They presented experimental results for kanji characters and obtained recognition rate of 98.4% in case of data freely written in square style and 96% for test data written in fast and cursive handwriting style.

Chan and Yeung (1999) proposed a structural approach for recognizing online handwriting. Their approach achieved reasonable speed, fairly high accuracy and sufficient tolerance to variations. At the same time, it maintained a high degree of reusability and hence facilitates extensibility. The recognition rates are 98.60% for digits, 98.49% for uppercase letters, 97.44% for lowercase letters, and 97.40% for the combined set. When the rejected cases are excluded from the calculation, the rates can be increased to 99.93%, 99.53%, 98.55% and 98.07%, respectively.

Spitz (1999) worked on shape based word recognition. The process relies on the transformation of text images into character shape codes, and on special lexical that contains information on the shape of words. Ambiguity is reduced by template matching using exemplars derived from surrounding text, taking advantage of the local consistency of font, face and size as well as image quality. This work describes the effects of lexical content, structure and processing on the performance of a word recognition engine.

Zhou *et al.* (1999) described a new kind of neural network, namely, quantum neural network and also presented its application to the recognition of handwritten numerals. Quantum neural network combined the advantages of neural modeling and fuzzy theoretic principles. An effective decision fusion system is proposed and a high reliability of 99.10% has been achieved.

Hu *et al.* (2000) used HMMs for writer independent online handwriting recognition system using combination of point oriented and stroke oriented features. Error rates for the character recognition test results are accurate to within less than 1% with 95% confidence.

Nakai *et al.* (2001) used sub stroke approach that recognizes online handwritten kanji characters using HMM. An experimental comparison between sub stroke HMM and character HMM was carried out on the 1,016 Japanese educational kanji recognition task. These results show that there is no big differences between the recognition rate of sub stroke HMM and character HMM.

Jaeger *et al.* (2001) presented the online handwriting recognition system based on a multistate time delay neural network, a hybrid architecture combining features of neural networks and HMMs. Two main features of a multistate time delay neural network were its time-shift invariant architecture and the nonlinear time alignment procedure.

Koblayashi *et al.* (2001) used template matching and proposed an algorithm called re parameterized angle variations which make explicit use of trajectory information where the time evolution of pen co-ordinates plays a crucial role. This choice appears to be reasonable since the top seven recognition rate was 96% without giving rise to too much computational burden.

Connell and Jain (2001) demonstrated a method of online character recognition which focuses on a representation of characters that models the different writing styles found in the training set. These models are then used by a decision classifier which yields good class determination. A method of automatically identifying writing styles and modeling these writing styles using templates has also been proposed by them. They were able to obtain an 86.9% classification accuracy for a 36-class set of alphanumeric characters with a throughput of over 8 characters/s on a 296 MHz Sun UltraSparc.

Connell and Jain (2002) approach was to make writer adaptation from writer independent writing style models to identify the styles present in a particular writer's training data. They used HMM in this approach. Writer adaptation achieves an average reduction in error rate of 9.2% over the writer independent rate for the case-independent word recognition task.

Brakensiek *et al.* (2002) describe a writer-independent online handwriting recognition system which is comparing the effectiveness of several confidence measures. Their recognition system for single German words is based on HMMs using a dictionary. They compare the ratio of rejected words to misrecognized words using four different confidence measures: They use a large online handwriting database of several writers consists of cursive script samples of 166 different writers. The training of the writer independent system is performed

using about 24400 words of 145 writers. Testing is carried out with 2071 words of 21 different writers. The recognition results are determined using an increasing threshold τ and using the baseline system without rejection a word recognition rate of 87.0% (1801 words are recognized correctly) is achieved testing the entire test-set of 2071 words. The presented results refer to a single word recognition rate using a dictionary of about 2200 German words.

Shimodiara *et al.* (2003) proposed a novel handwriting recognition interface for wearable computing where users write characters continuously without pauses on a small single writing box. They used HMM as a recognition method. The proposed method demonstrated promising performance with 69.2% of handwriting sequences being correctly recognized when different stroke order was permitted, and the rate was improved up to 88.0% when characters were written with fixed stroke order.

Gunter and Bunke (2004) presented methods for the creation of classifier ensembles based on feature selection algorithms using HMM. The performance of the new ensemble method was 1.5% higher than the best combination of the base classifiers, 2.94 % higher than the classical ensemble methods, and 4.48 % higher than the best base classifier.

Schlapbach and Bunke (2004) used HMM based recognizers that identify and verify writers. In an identification experiment the 96.56% of the writer out of a set of 100 writers is correctly identified. Second, in a verification experiment using over 8,600 text lines from 120 writers an equal error rate of about 2.5% is achieved.

Funanda *et al.* (2004) used HMM in online handwriting recognition that reduces memory usage and further improves the recognition rate. Self-Organizing Map (SOM) density tying reduced the dictionary size to $1/7$ of the original size with a recognition rate of 90.45%, only slightly less than the original recognition rate of 91.51%. their additional feature increased recognition capability to 91.34%.

A good introduction related to problems in structural and syntactical methods is given by Basu *et al.* (2005).

Shintani *et al.* (2005) developed a small scale four-layered neural network model for simple character recognition, which can recognize the patterns transformed by affine conversion. 4200 learning patterns and 1200 noise patterns (noise patterns are changed for each iteration.) were presented to the network. The root mean square (rms) error of the output is less than 0.005

Jindal *et al.* (2006) have proposed a solution for segmenting horizontally overlapping lines. Whole document has been divided into strips and proposed algorithm has been applied for segmenting horizontally overlapping lines and associating small strips to their respective lines. The results reveal that the algorithm is almost 99% perfect when applied to the Gurmukhi script.

Chain code has been used in finger print recognition by Paul *et al.* (2007).

Jindal *et al.* (2008) have discussed various structural features used for recognizing degraded printed Gurmukhi script documents containing touching characters and heavy printed characters. For classification purpose they have used k-NN, SVM classifiers. It was observed that maximum recognition accuracy of 83.60% at $k = 1$ is obtained when all the structural features have been used. Similarly, it was observed that maximum accuracy of 91.54% using SVM classifier was found.

Sharma *et al.* (2008) discussed a process that recognizes characters in two stages. First stage recognizes the strokes, in second stage; character is evaluated on the basis of recognized strokes. For 60 writers and a set of 41 Gurmukhi characters, they have obtained recognition rate as 90.08%.

Sharma *et al.* (2009) proposed a new step as rearrangement of recognized strokes in online handwriting recognition procedure. The rearrangement of recognized strokes includes: strokes identification as dependent and major dependent strokes; the rearrangement of strokes with respect to their positions; the combination of strokes to recognize character. They have achieved an overall recognition rate as 81.02% in online handwritten cursive handwriting for a set of 2576 Gurmukhi dictionary words.

Kumar *et al.* (2011) have attempted grading of writers based on offline Gurmukhi characters written by them. Grading has been accomplished based on statistical measures of distribution of points on the bitmap image of characters. The gradation features used for classification are based on zoning, which can uniquely grade the characters. In this work, one hundred different Gurmukhi handwritten data sets have been used for grading the handwriting. They have used k-NN, HMM and Bayesian decision making classifiers for classification. They have taken samples of Gurmukhi characters from one hundred different writers. In the process of grading, they have found the writer with best handwriting, based on the characters taken in

the data set. They have calculated average score of all writers obtained by them when the five features and three classifiers are used.

Based on this literature survey, it has been observed that most of the researchers have done recognition in online as well as offline Handwriting recognition systems using HMM, k-nearest neighbor, SVM, EM, Naive Bayes methods.

It has been observed from the literature survey that most of the work has been done for English, Japanese, Chinese, Arabic, Urdu, Devnagiri and Tamil. The recognition systems for online Gurmukhi script are still in initial stages and the most of the work for Gurmukhi script is for offline recognition.

This thesis is an attempt to develop an online handwriting recognition system for Gurmukhi script using SVM which is one of the most important methodologies in data mining (Ward and Kuklinski, 1988).

CHAPTER 2

DATA COLLECTION, PREPROCESSING AND FEATURE SELECTION

As it is already discussed in chapter 1, the important phases in an online handwriting recognition system before recognition process are data collection, preprocessing and feature computation. Section 2.1 concentrate on data collection, section 2.2 includes preprocessing and various phases in preprocessing, and section 2.3 discusses feature computation. All of these phases are dependent on each other and the accuracy of the recognition is affected by the performance of these phases.

2.1 Data capturing

In this phase, the data has been collected in the form of co-ordinates. These co-ordinates are input by mouse or writing pads by using stylus/pen. The sequential positions of the stroke are stored dynamically.

The Samples of i^{th} stroke $S_i = \{(X_{io}, Y_{i0}), (X_{i1}, Y_{i1}), \dots, (X_{io}, Y_{i0}), \dots, (X_{in}, Y_{in})\}$ expressed by the time sequence of co-ordinates as Figure 2.1 represents the generation of the data for a sample of a Gurmukhi characters in which (X_0, Y_0) and (X_n, Y_n) denote the initial and the final point of the generated time sequence of co-ordinates for the particular sample.

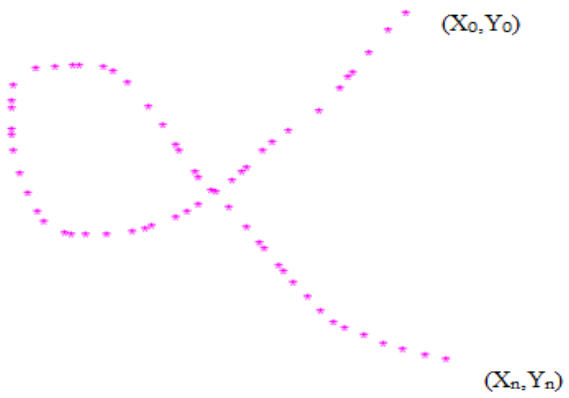


Figure 2.1: Stroke of a Gurmukhi character

2.2 Algorithms used in preprocessing

Due to hardware and software limitations, the input stroke contains noise and distortions. These imperfections in stroke have to be removed before recognition. This can be removed by applying various methods collectively known as preprocessing. Preprocessing phase for Gurmukhi character consists of following stages as shown in Figure 2.2 (Sharma, 2009).

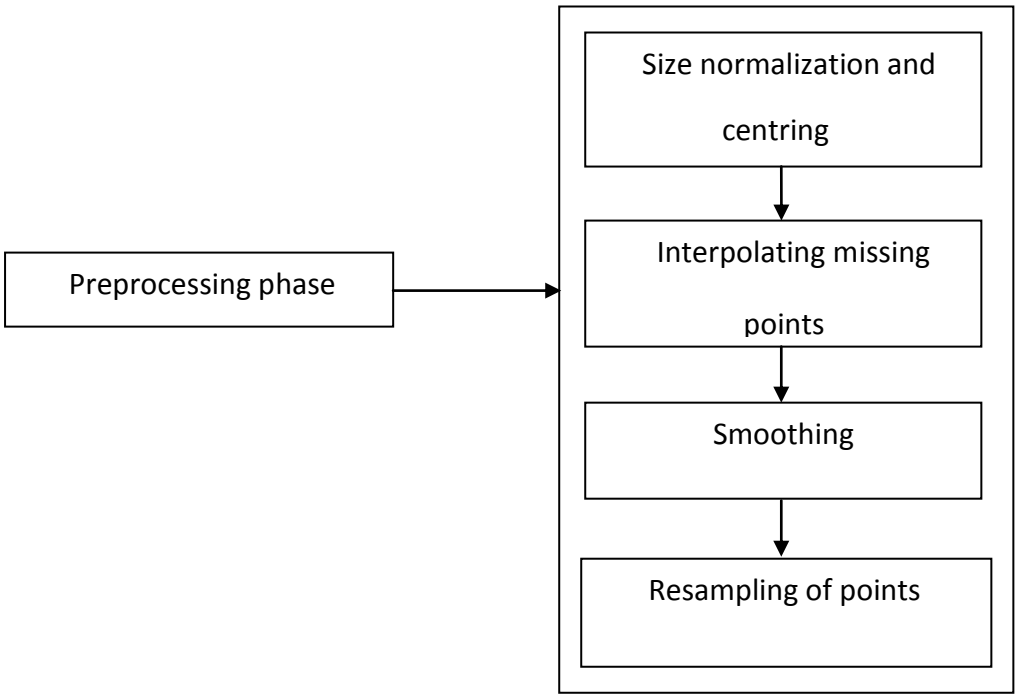


Figure 2.2: Various stages of preprocessing

1. Size Normalization and Centering
2. Interpolating Missing Points
3. Smoothing
4. Re sampling Of Points

In the Normalization phase, stroke of Gurmukhi character is resized on window of 300 × 300 and moved to the center of the window, if it is written along the boundary of writing window centering is required.

The missing points present in a stroke due to variation in speed of writing by the user can be calculated using various techniques such as Bezier and B-Spline. This phase is called interpolation phase. Piecewise Bezier interpolation has been used because it helps to

interpolate points among fixed number of points. After this phase, number of points in the stroke written by the user will increase.

Smoothing is required in an online handwriting recognition system because of the presence of noise in the handwriting. This noise comes in the stroke due to individual handwriting style and the hardware limitations. The algorithm for this phase uses average of a point with its neighbors to perform smoothing.

Resampling is required to keep the points in a stroke equidistant as far as possible and to a specific number. The number of points in actual stroke is generally large and varies greatly due to high variation in writing speed. A fixed, lesser number and equally spaced points are selected for recognition process.

2.3 Feature selection

Points generated after preprocessing phase are used as a feature for recognition; these points are always fixed for each stroke and are equidistant as far as possible. In this thesis number of points in a stroke is fixed to 64. As discussed in earlier chapters, an online handwriting recognition system consists of mainly Data Collection, preprocessing, feature selection and recognition. In this chapter the first three phases of online handwriting systems are briefly described. The next chapter covers the most important phase of an online handwriting recognition system, *i.e.*, recognition. This recognition has been done using Support Vector Machine (SVM).

CHAPTER 3

GURMUKHI STROKE RECOGNITION

In this chapter recognition process used in our work has been discussed which is based on recognition of online handwritten Gurmukhi strokes. Before recognition, an input handwritten stroke is preprocessed as discussed in chapter 2. The preprocessed handwritten stroke is then recognized using SVM. Section 3.1 discusses about Support Vector Machine (SVM). Recognition using MATLAB is shown in section 3.2.

Different strokes of Gurmukhi characters from three different writers are collected using the iBall electromagnetic digital pen Tablet which is having 1024 levels of pressure sensitivity, resolution 400 lpi, 200 rps report rate, and have ± 0.01 inch accuracy.

Each of the 3 writers has written same 100 words of Gurmukhi script. These words are shown in Table 3.1.This Table also shows number of strokes used to write each word by a particular writer.

Table 3.1: The 100 words used for recognition and written by three writers

Serial number	Gurmukhi word	Number of strokes		
		Writer 1	Writer 2	Writer 3
1	ਅਣਕਿਆਸੀ	12	12	13
2	ਅਣਖ	8	7	9
3	ਅਣਦੇਖਿਆ	12	11	13
4	ਅਣਪਛਾਤੇ	10	9	11
5	ਅਣਪੜ੍ਹ	9	8	10
6	ਅੰਤਰ	4	4	5
7	ਅੰਤਰਰਾਸ਼ਟਰੀ	11	11	12
8	ਅਤਿਵਾਦੀ	8	7	8
9	ਅਤੇ	3	3	4
10	ਅਥਾਰਟੀ	9	8	9

11	ਇਜ਼ਹਾਰ	10	9	9
12	ਇੱਜ਼ਤ	8	8	9
13	ਇਟਲੀ	9	9	9
14	ਇੰਡੀਅਨ	11	9	11
15	ਇੰਤਜ਼ਾਮ	12	11	12
16	ਇਤਿਹਾਸ	10	9	10
17	ਇਤਿਹਾਸਕ	11	10	11
18	ਇੱਥੇ	9	9	9
19	ਇਥੋਂ	9	9	9
20	ਇਨਕਾਰ	9	8	9
21	ਅਕਾਲੀ	8	8	9
22	ਅਖ਼ਬਾਰ	10	9	10
23	ਅਖਵਾਉਂਦਾ	11	10	12
24	ਅਖਾਣਾਂ	10	9	11
25	ਅੰਗ	5	4	5
26	ਅੰਗਹੀਣ	9	9	10
27	ਅੰਗਰੇਜ਼ੀ	11	10	11
28	ਅਗਲੇ	8	8	9
29	ਅਗਵਾਈ	9	8	10
30	ਅਗਾਂਹਵਧੂ	11	11	13
31	ਉੱਪਰ	6	6	6
32	ਉਪਰਲੀ	10	10	10
33	ਉਪਲਬਧ	12	11	12
34	ਉਪਾਵਾਂ	9	8	8
35	ਉੱਭਰੇ	6	6	6
36	ਉਮਰ	5	5	6
37	ਉਮੀਦਵਾਰਾਂ	10	10	12
38	ਉਰਦੂ	7	6	6
39	ਉਲਟ	7	6	7
40	ਊਸ਼ਣ	8	9	8
41	ਅਸੀਂ	5	5	7
42	ਅਸੀਮ	6	7	8

43	ਅੱਸੂ	7	6	8
44	ਅਹਿਸਾਸ	9	8	9
45	ਅਹੁਦਾ	6	5	6
46	ਅਹੁਦੇ	6	5	7
47	ਅਕਸਰ	6	5	6
48	ਅਕਸਰ	5	5	9
49	ਅਕਤੂਬਰ	9	7	9
50	ਅੰਕਲ	7	7	8
51	ਅੰਗੂਠੇ	10	7	9
52	ਅੱਗੇ	6	5	7
53	ਅਚੰਭਾ	6	5	7
54	ਅਚਾਨਕ	7	5	6
55	ਅੱਜ	4	4	6
56	ਅਜਬ	5	5	7
57	ਅੰਜਾਮ	8	7	9
58	ਅਜਿਹਾ	7	6	8
59	ਅਜਿਹੀ	7	6	8
60	ਅਜਿਹੇ	8	5	8
61	ਉਠਾਉਣ	9	9	9
62	ਉਠਾਇਆ	10	9	11
63	ਉਡਾਣ	7	7	7
64	ਉਡੀਕ	5	5	6
65	ਉਡੀਕਦੇ	8	7	8
66	ਉਤਸਕ	7	6	7
67	ਉਤਸ਼ਾਹ	9	8	9
68	ਉਤਸ਼ਾਹਿਤ	11	9	11
69	ਉਤਪਾਦਕਤਾ	11	10	11
70	ਉਤਰ	4	4	5
71	ਆਈ	6	6	7
72	ਆਏ	6	6	7
73	ਆਸ	6	5	6
74	ਆਸਥਾ	9	7	9

75	ਆਸ਼ਰਮ	9	9	11
76	ਆਕਾਸ਼	8	7	9
77	ਆਗੂਆਂ	9	9	12
78	ਆਜ਼ਾਦ	8	7	9
79	ਆਜ਼ਾਦੀ	9	8	9
80	ਆਟਾ	5	4	6
81	ਸਹਿਤ	6	5	6
82	ਸਹਿਮਤੀ	8	8	9
83	ਸ਼ਹਿਰ	7	6	7
84	ਸ਼ਹਿਰੋਂ	9	8	8
85	ਸ਼ਹੀਦ	7	6	7
86	ਸਹੁੰ	6	5	6
87	ਸਹੂਲਤ	10	9	10
88	ਸ਼ੱਕ	11	11	13
89	ਸਕਣਾ	8	7	9
90	ਸਕੱਤਰ	12	11	13
91	ਸਕੱਤਰੇਤ	11	9	11
92	ਸ਼ਕਤੀ	9	8	11
93	ਸ਼ਕਤੀਆਂ	4	4	6
94	ਸਕਦਾ	11	11	13
95	ਸਕਦੀ	7	7	9
96	ਸਕਦੀਆਂ	3	3	5
97	ਸਕਦੇ	9	8	10
98	ਸ਼ਕਲ	9	9	10
99	ਸਕਾਂਗੀ	8	8	9
100	ਸਕਿਓਰਿਟੀ	9	9	9

The words are stored as a collection of strokes in an xml format and the snapshot of an xml file is shown below in Figure 3.1. There is a single xml file for each word and the strokes of which consists each word is stored as a collection of points and only the stroke are considered for recognition.

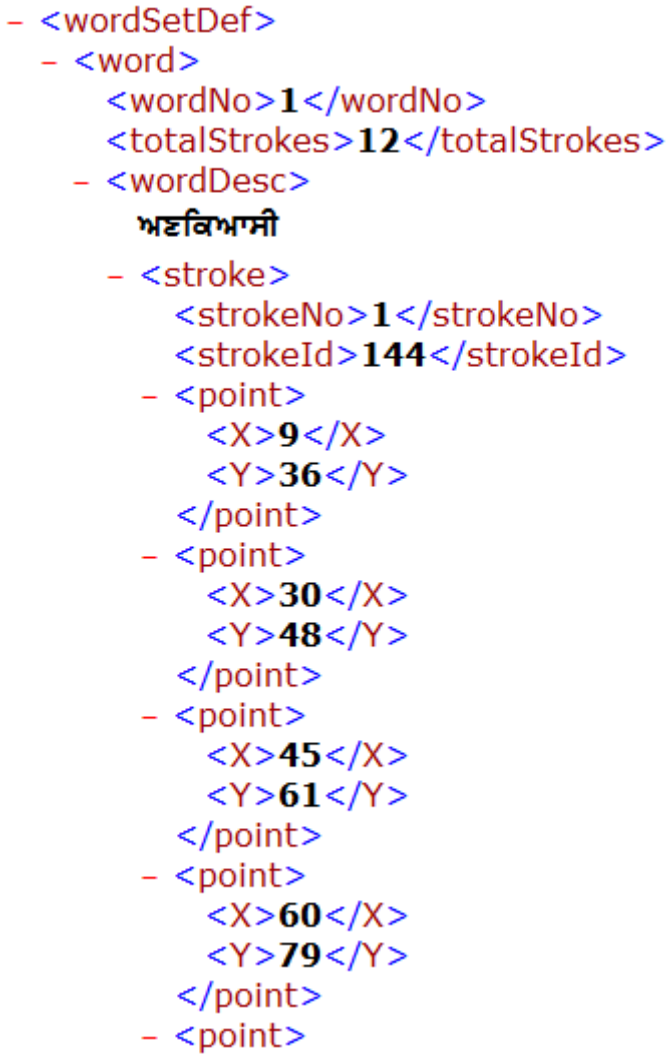


Figure 3.1: An xml file for ਅਣਕਿਆਸੀ

There are total 2428 strokes and each of this stroke is assigned stroke IDs as its unique identification number. These strokes and there corresponding IDs with their respective zones are presented in Table 3.2 to Table 3.5.

Table 3.2: Unique lower zone strokes with their stroke IDs

S.NO	STROKE ID	SYMBOL	DESCRIPTION
1)	101	ੱ	Onkar
2)	102	ਂ	Bindi
3)	103	਼	Ra
4)	104	੍	Ha

5)	105	꣇	Va
6)	106	꣆	Dulenkar
7)	107	ꣅ	Halanq

Table 3.3: Unique upper zone strokes with their stroke IDs

S.NO	STROKE ID	SYMBOL	DESCRIPTION
1)	121		Upper bar
2)	122		Lawan
3)	123		Dalawan
4)	124		Hora
5)	125		Ghanoura bar
6)	126		Adhak
7)	127		Bindi
8)	128		Tippi
9)	129		Associate bar
10)	130		Dalawan Side

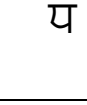
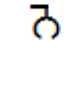
Table 3.4: Unique middle zone strokes with their stroke IDs

S.NO	STROKE ID	SYMBOL	DESCRIPTION
1)	141		Urah
2)	142		Urah

3)	143		Urah
4)	144		Aarah
5)	145		Aarah
6)	146		Eedi
7)	147		Eedi
8)	148		Eedi
9)	149		Eedi
10)	150		Eedi
11)	151		Sassa
12)	152		Sassa
13)	153		Sassa
14)	154		Sassa
15)	155		Haaha
16)	156		Haaha

17)	157		Kakka
18)	158		Kakka
19)	159		Khakha, Pappa
20)	160		Khakha,
21)	161		Khakha, Pappa
22)	162		Vertical Bar
23)	163		Khakha, Dhadha
24)	164		Gagga, Rara
25)	165		Gagga, Rara
26)	166		Ghagga
27)	167		Ghagga
28)	168		Aeyaa
29)	169		Aeyaa

30)	170		Chachaa
31)	171		Chachaa
32)	172		Chhachha
33)	173		Chhachha
34)	174		Jajja
35)	175		Jajja
36)	176		Jhajja
37)	177	Y	Jhajja
38)	178		Jhajja
39)	179		Neyaa, Vawa
40)	180		Neyaa, Vawa
41)	181		Vawa
42)	182		Tenqa
43)	183		Thathaa

44)	184		Thathaa
45)	185		Dadda
46)	186		Dadda
47)	187		Dhadhaa
48)	188		Dhadhaa
49)	189		Nahnaa
50)	190		Nahnaa
51)	191		Tatta
52)	192		Tatta
53)	193		Dadhaa
54)	194		Dadhaa
55)	195		Dadhaa
56)	196		Nanhaa
57)	197		Nanhaa
58)	198		Nanhaa

59)	199	।	Nanhaa half vertical bar
60)	200	੶	Faffa
61)	201	ੵ	Faffa
62)	202	ੲ	Babba
63)	203	ੳ	Babba
64)	204	ੴ	Bhabhaa
65)	205	ੵ	Bhabhaa
66)	206	ਮ	Mamma
67)	207	੫	Yaeya
68)	208	ਯ	Yaeya
69)	209	੫	Yaeya
70)	210	ਲ	Lalla
71)	211	V	Lalla
72)	212	ੜ	Rara
73)	213	।	Rara

74)	214		Rara
75)	215		Cehari
76)	216		Behari
77)	217	૧	One
78)	218	૨	Two
79)	219	૩	Three
80)	220	૪	Four
81)	221	૫	Five
82)	222	૬	Six
83)	223	૭	Seven
84)	224	૮	Eight
85)	225	૯	Nine
86)	226	0	Zero
87)	227		Mamma
88)	228		Jhajja

89)	229		Rara
90)	230		Cehari
91)	231		Mamma
92)	232		Jajja
93)	233		Jajja

In this chapter, we have focused on SVM that we are using for stroke level recognition using MATLAB.

3.1 Support vector machine

According to Vapnik (1995), a Support vector Machine (SVM) is a concept in statistics and computer science for a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis. The standard SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the input. SVM in its basic form implement two class classifications. 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). Its formulation approximates SRM principle by maximizing the margin of class separation, the reason for it to be known also as large margin classifier. The basic SVM formulation is for linearly separable datasets. It can be used for non-linear datasets by indirectly mapping the nonlinear inputs into to linear feature space where the maximum margin decision function is approximated. The mapping is done by using a kernel function. A support vector machine includes Support Vector Classifier (SVC) and Support Vector Regressor (SVR).

Support Vector Classifier: For a two-class linearly separable learning task, the aim of SVC is to find a hyper-plane that can separate two classes of given samples with a maximal margin which has been proved able to offer the best generalization ability. Generalization ability refers to the fact that a classifier not only has good classification performance (*e.g.*, accuracy) on the training data, but also guarantees high predictive accuracy for the future data from the

same distribution as the training data. Intuitively, a margin can be defined as the amount of space, or separation, between the two classes as defined by a hyper-plane. Geometrically, the margin corresponds to the shortest distance between the closest data points to any point on the hyper-plane. Figure 3.2 illustrates a geometric construction of the corresponding optimal hyper-plane under the above conditions for a two-dimensional input space.

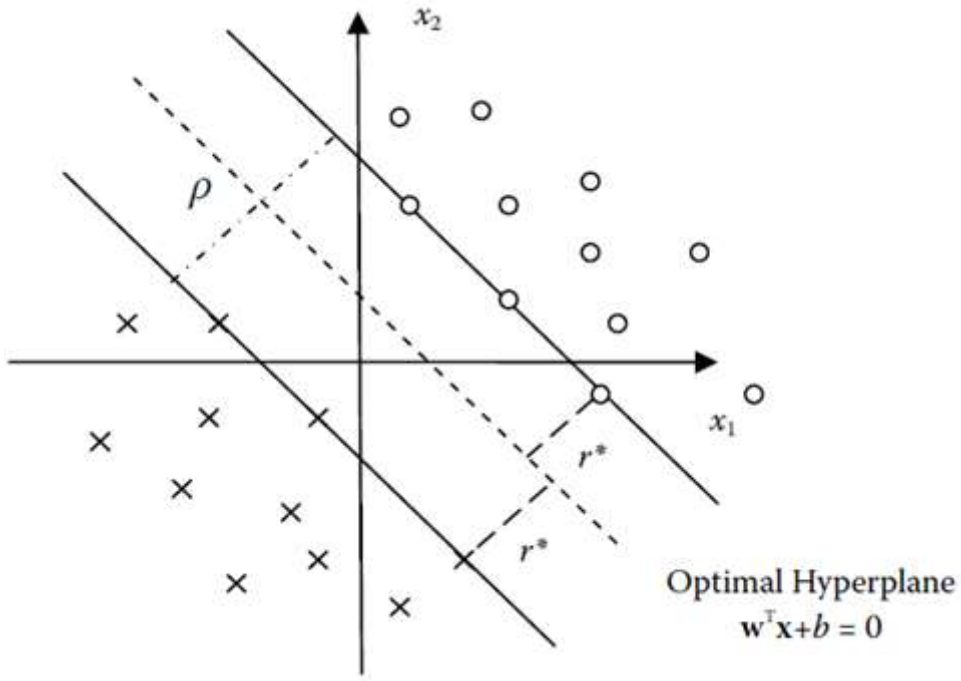


Figure 3.2: Illustration of the optimal hyper-plane in SVC for a linearly separable case

Let w and b denote the weight vector and bias in the optimal hyper-plane, respectively. The corresponding hyper-plane can be defined as

$W^T X + b = 0$

Consequently, SVC aims to find the parameters **w** and b for an optimal hyper-plane in order to maximize the margin of separation that is determined by the shortest geometrical distances from the two classes, respectively, thus SVC is also called maximal margin classifier. Maximal margin SVC, including SVR, represents the original starting point of the SVM algorithms. However, in many real-world problems, it may be too rigid to require that all points are linearly separable, especially in many complex nonlinear classification cases. When the samples cannot be completely linearly separated, the margins may be negative. In these cases, the feasible region of the primal problem is empty, and thus the corresponding

dual problem is an unbounded objective function. This makes it impossible to solve the optimization problem. To solve these inseparable problems, we generally adopt two approaches. The first one is to relax the rigid inequalities and thus lead to so-called soft margin optimization. Another method is to apply the kernel trick to liberalize those nonlinear problems.

Support Vector Regressor: SVM can also be applied to regression problems by the introduction of an alternative loss function (Smola, 1996). The loss function must be modified to include a distance measure. The traditional least-squares estimator may not be quite feasible in the presence of outliers, resulting in the regressor to perform poorly when the underlying distribution of the additive noise has a long tail.

LibSVM and SVMlight are two of the most famous software about the implementation of SVM algorithms. One of the initial drawbacks of the SVM is its costly computational complexity in the training phase, which leads to inapplicable algorithms in the large datasets. However, this problem is being solved with great success. One approach is to break a large optimization problem into a series of smaller problems, where each problem only involves a couple of carefully chosen variables so that the optimization can be done efficiently. The process iterates until all the decomposed optimization problems are solved successfully. The optimal separating hyper-plane can be determined without any computations in the higher dimensional feature space by using kernel functions in the input space. Commonly used kernels include:

1. **Linear Kernel:** $K(x, y) = x.y$

2. **Radial Basis Function (Gaussian) Kernel:** $\mathrm{K(x, y)} = \exp(-\|\mathrm{x} - \mathrm{y}\|^2/2\sigma^2)$

3. **Polynomial Kernel:** $\mathrm{K(x, y)} = (\mathrm{x.y} + 1)^{\mathrm{d}}$

3.2 Recognition using MATLAB

MATLAB is a high performance language for technical computing, created by The MathWorks. It features a family of add-on application-specific solutions called toolboxes *(i.e.,* comprehensive collections of functions) that extend the MATLAB environment to solve

particular classes of problems. In this research work MATLAB version 7.10.0.499(R2010a) with bioinformatics toolbox is used for recognition of Gurmukhi stroke. Bioinformatics toolbox offers an integrated software environment for genome and proteome analysis. The key feature of the basic categories in the bioinformatics toolbox is a general-purpose technical computing language and development environment that is widely used in scientific and engineering applications. The bioinformatics toolbox for MATLAB is a library of functions that adds bioinformatics capabilities to the MATLAB environment. It contains a number of set of functions from which a statistical learning set of functions is used. This set of functions is used to classify and identify features in data sets, set up cross-validation experiments, and compare different classification methods. In this work two methods holdout and *k*-fold cross validation has been used and are discussed in next sections.

3.2.1 Hold out cross validation for training and testing

The holdout method is the simplest kind of cross validation. The data set is separated into two sets, called the training set and the testing set. The errors it makes are accumulated as before to give the mean absolute test set error, which is used to evaluate the model. The advantage of this method is that it is usually preferable to the residual method and takes no longer to compute. However, its evaluation can have a high variance. The evaluation may depend heavily on which data points end up in the training set and which end up in the test set, and thus the evaluation may be significantly different depending on how the division is made. In MATLAB crossvalind function is used. This function is used to generate cross validation indices. Prototype of this cross validation function is crossvalind('HoldOut', N, P). It returns logical index vectors for cross-validation of N observations by randomly selecting P*N (approximately) observations to hold out for the evaluation set. P must be a scalar between 0 and 1. P defaults to 0.5 when omitted, corresponding to holding 50% out.

3.2.2 *k*-Fold cross-validation for Training and Testing

k-fold cross validation is one way to improve over the holdout method. The data set is divided into k subsets, and the holdout method is repeated k times. Each time, one of the k subsets is used as the test set and the other $k - 1$ subsets are put together to form a training set. Then the average error across all k trials is computed. The advantage of this method is that it matters less how the data gets divided. Every data point gets to be in a test

set exactly once, and gets to be in a training set $k-1$ times. The variance of the resulting estimate is reduced as *k* is increased.

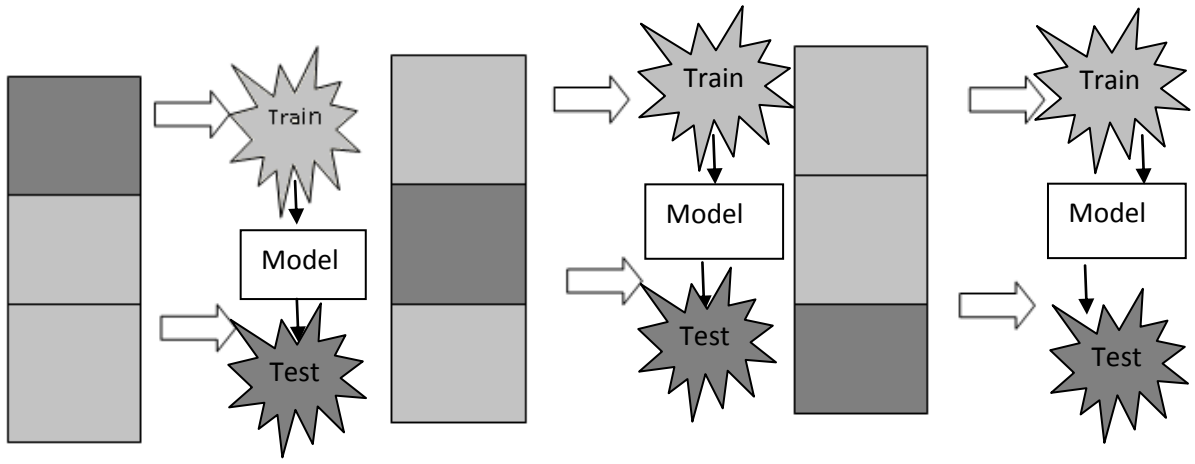


Figure 3.3: Functioning of 3-fold cross validation

In MATLAB crossvalind('Kfold', N, *k*) function is used for Gurmukhi stroke recognition. It returns randomly generated indices for a *k*-fold cross-validation of N observations. Indices contain equal (or approximately equal) proportions of the integers 1 through *k* that define a partition of the N observations into *k* disjoint subsets. Repeated calls return different randomly generated partitions. *k* defaults to 5 when omitted. In *k*-fold cross validation, $k-1$ folds are used for training and the last fold is used for evaluation. This process is repeated $k-1$times, leaving one different fold for evaluation each time.

In the next chapter the result after applying two partitioning strategies of cross validation in SVM is applied on the data collected from three different writers consisting 2428 strokes is discussed in detail.

CHAPTER 4

RESULTS AND DISCUSSIONS

This chapter contains the results of the experiments carried out for recognition of Gurmukhi strokes written by 3 writers. Each writer has written 100 words in Gurmukhi script. These words are stored in an xml format as shown in chapter 3. Table 3.1 shows 100 words written by each writer. Each word is divided in three zones namely lower zone, middle zone and upper zone. Unique IDs for each stroke of a particular zone is given in Table 3.2, Table 3.3 and Table 3.4.

Table 4.1, given below, shows the frequency of occurrences of each stroke in 100 words written by three writers. Each stroke is identified by a unique ID in the Table.

Table 4.1: Stroke frequency table for each writer

Stroke ID	Writer 1	Writer 2	Writer 3
101	40	40	40
102	19	20	20
104	-	-	1
105	2	-	-
121	91	26	96
122	15	15	15
124	2	2	2
126	10	10	10
127	12	11	12
128	12	12	12
129	5	6	10
141	-	-	23
142	20	23	-
144	53	55	-
145	2	-	56

146	32	11	34
147	14	-	14
148	9	23	9
149	10	23	9
151	37	3	40
152	5	-	-
154	17	94	-
155	20	-	20
156	-	20	-
157	23	1	26
158	4	26	1
159	-	-	2
161	18	23	17
162	85	48	144
163	9	9	9
164	26	-	34
165	10	35	1
170	2	-	2
171	-	2	-
172	1	-	1
173	1	1	-
174	11	12	12
179	5	-	6
181	1	6	-
182	1	9	-
183	3	-	3
184	-	3	-
185	4	4	-
186	-	-	4
189	11	5	11
191	25	25	1
192	-	-	28

193	14	-	12
194	2	16	-
197	12	12	12
199	45	6	58
202	4	4	4
204	2	-	2
205	2	2	-
212	3	1	1
213	1	1	1
215	27	19	28
216	27	27	26
226	4	43	6
229	-	9	-
Total	810	743	875

In the recognition process we have used SVM in MATLAB. SVM is applied to both non preprocessed and preprocessed data. Recognition is done for each writer separately as well as for the combined data of all the writers. Writer 1 has used 810 strokes, writer 2 has used 743 strokes and writer 3 has used 875 strokes for writing 100 words in Gurmukhi scripts as shown in Table 3.1. Recognition has also been done using 1000, 2000 and 3000 strokes of preprocessed data for each writer.

Recognition has been done for only those strokes that are written by writer 1 only and are present in the strokes written by writer 2 and writer 3 also. Section 4.1 gives the recognition accuracy for non preprocessed data. Subsections of section 4.1 shows recognition accuracy using different partitioning techniques like holdout cross validation and *k*-fold cross validation as discussed in chapter 3. A subsequent section 4.2 gives the recognition accuracy for preprocessed data using holdout and *k*-fold cross validation techniques. Section 4.3 gives the result for recognition using 1000, 2000 and 3000 preprocessed strokes.

4.1 Recognition results using non preprocessed data

In this section the results of the recognition for non preprocessed data are given. Two techniques holdout and *k*-fold cross validation for each writer in MATLAB has been used. In

holdout cross validation 3 strategies have been used to partition the data into training and testing sets. Also 10-fold method is used for generating training and testing sets.

4.1.1 Results using holdout cross validation

Three types of partitions (70% training, 60% training and 50% training) have been used for recognition of non preprocessed data. These partitions are used for recognition of stroke IDs, for individual writer. Also recognition is done on data collected from each writer.

After applying these partitioning techniques, the results are shown in Table 4.2, Table 4.3 and Table 4.4. In these Tables recognition accuracy for each writer has been shown for every stroke id and the recognition accuracy for the combined data is shown in the last column. The blank entries in the Table 4.2, Table 4.3 and Table 4.4 are due to the absence of stroke, *i.e.* stroke is not written by the writer.

Table 4.2: Stroke wise recognition accuracy for each writer with 70% training

Stroke ID	Writer 1 (W1)	Writer 2 (W2)	Writer 3 (W3)	W1+W2+W3
101	0.9975	0.9920	1	0.9926
102	0.9901	1	1	0.9975
105	0.9975	-	-	0.9992
121	0.9802	0.9625	0.9635	0.9688
122	0.9950	0.9920	1	0.9901
124	0.9951	0.9946	0.9954	0.9975
126	0.9975	0.9946	0.9954	0.9942
127	1	1	1	1
128	0.9679	0.9812	0.9909	0.9836
129	0.9975	0.9812	0.9909	0.9918
142	0.9802	0.9866	-	0.9836
144	0.9877	0.9839	-	0.9901
145	0.9975	-	0.9908	0.9762
146	0.9556	0.9839	0.9521	0.9622
147	0.9852	-	0.9817	0.9901
148	0.9678	0.9651	0.9886	0.9836
149	0.9852	0.9705	0.9908	0.9803

151	0.9431	0.9973	0.9612	0.9556
152	0.9926	-	-	0.9984
154	0.9777	0.8928	-	0.9507
155	0.9679	-	0.9749	0.9811
157	0.9480	1	0.9726	0.9745
158	0.9926	0.9544	1	0.9819
161	0.9704	0.9571	0.9748	0.9737
162	0.9604	0.9893	0.9749	0.9350
163	0.9876	0.9759	0.9863	0.9893
164	0.9605	-	0.9566	0.9523
165	0.9728	0.9383	0.9977	0.9737
170	0.9926	-	0.9977	0.9984
172	1.0	-	0.9977	0.9984
173	0.9975	0.9973	-	0.9967
174	0.9827	0.9893	0.9909	0.9827
179	0.9802	-	0.9886	0.9959
181	0.9975	0.9893	-	0.9967
182	0.9827	0.9732	-	0.9836
183	0.9901	-	0.9977	0.9967
185	0.9926	0.9812	-	0.9951
189	0.9802	0.9866	0.9863	0.9893
191	0.9604	0.9437	0.9977	0.9638
193	0.9704	-	0.9795	0.9819
194	0.9951	0.9491	-	0.9893
197	0.9679	0.9839	0.9863	0.9852
199	0.9282	0.9920	0.9429	0.9638
202	0.9901	0.9973	0.9886	0.9951
204	0.9975	-	0.9977	0.9992
205	0.9975	0.9973	-	0.9942
212	1	0.9973	1	0.9975
213	0.9950	0.9973	0.9931	0.9942
215	0.9752	0.9759	0.9726	0.9622

216	0.9926	0.9786	0.9658	0.9729
226	0.9926	0.9196	0.9886	0.9696

From Table 4.2 it has been observed that the average recognition accuracy for writer 1 is 98.25%, for writer 2 is 97.80% and for writer 3 is 98.52%. So it can be concluded that writer 3 is the most consistent writer as compared to other writers. An overall accuracy of 98.33% for all writers has been observed.

Table 4.3: Stroke wise recognition accuracy for each writer with 60% training

Stroke ID	**Writer 1 (W1)**	**Writer 2 (W2)**	**Writer 3 (W3)**	**W1+W2+W3**
101	1	0.9955	0.9962	0.9952
102	0.9938	0.9978	1	0.9973
105	0.9979	-	-	0.9993
121	0.9753	0.9709	0.9733	0.9575
122	0.9979	0.9888	1	0.9911
124	0.9979	0.9978	0.9962	0.9986
126	0.9959	0.9911	0.9981	0.9836
127	0.9979	1	1	0.9904
128	0.9773	0.9844	0.9924	0.9925
129	0.9979	0.9888	0.9905	0.9877
142	0.9877	0.9933	-	0.9733
144	0.9876	0.9888	-	0.9664
145	0.9979	-	0.9886	0.9877
146	0.9485	0.9776	0.9600	0.9836
147	0.9835	-	0.9886	0.9815
148	0.9691	0.9553	0.9867	0.9589
149	0.9856	0.9553	0.9905	0.9986
151	0.9423	0.9978	0.9600	0.9541
152	0.9938	-	-	0.9788
154	0.9794	0.9217	-	0.9794
155	0.9650	-	0.9581	0.9774
157	0.9588	0.9978	0.9714	0.9692

158	0.9814	0.9262	1	0.9315
161	0.9753	0.9485	0.9810	0.9883
162	0.9527	0.9843	0.9676	0.9671
163	0.9876	0.9843	0.9810	0.9561
164	0.9464	-	0.9619	0.9925
165	0.9671	0.9330	0.9962	0.9993
170	0.9979	-	0.9962	0.9788
172	1	-	1	0.9993
173	0.9979	0.9955	-	0.9774
174	0.9773	0.9933	0.9886	0.9911
179	0.9835	-	0.9771	0.9938
181	0.9959	0.9866	-	0.9931
182	0.9938	0.9709	-	0.9938
183	0.9856	-	0.9962	0.9918
185	0.9897	0.9888	-	0.9890
189	0.9814	0.9911	0.9771	0.9740
191	0.9506	0.9397	0.9981	0.9842
193	0.9546	-	0.9810	0.9897
194	0.9876	0.9642	-	0.9829
197	0.9794	0.9799	0.9876	0.9836
199	0.9115	0.9821	0.9485	0.9623
202	0.9856	0.9821	0.9829	0.9918
204	0.9938	-	0.9962	0.9986
205	0.9959	0.9933	-	0.9966
212	1.0	0.9978	1	0.9952
213	0.9979	1	0.9962	0.9979
215	0.9732	0.9776	0.9733	0.9609
216	0.9753	0.9754	0.9829	0.9746
226	0.9794	0.9508	0.9867	0.9664

From Table 4.3 it has been observed that the average recognition accuracy for writer 1 is 98.09%, for writer 2 is 97.81% and for writer 3 is 98.51%. So it can be concluded that writer

3 is the most consistent writer as compared to other writers. An overall accuracy of 98.24% for all writers has been observed.

Table 4.4: Stroke wise recognition accuracy for each writer with 50% training

Stroke ID	Writer 1 (W1)	Writer 2 (W2)	Writer 3 (W3)	W1+W2+W3
101	0.9982	0.9962	0.9967	0.9947
102	0.9947	1	0.9804	0.9976
105	0.9982	-	-	0.9994
121	0.9523	0.9502	0.9429	0.9577
122	0.9912	0.9885	0.9984	0.9912
124	0.9965	0.9981	1	0.9965
126	0.9965	0.9904	0.9967	0.9924
127	0.9982	1	1	0.9994
128	0.9859	0.9923	0.9853	0.9853
129	0.9965	0.9847	0.9886	0.9918
142	0.9806	0.9866	-	0.9906
144	0.9823	0.9885	-	0.9953
145	0.9947	-	0.9902	0.9830
146	0.9558	0.9751	0.9608	0.9647
147	0.9753	-	0.9739	0.9912
148	0.9541	0.9540	0.9739	0.9765
149	0.9718	0.9406	0.9706	0.9706
151	0.9187	0.9962	0.9445	0.9448
152	0.9876	-	-	0.9959
154	0.9682	0.8602	-	0.9559
155	0.9577	-	0.9739	0.9748
157	0.9435	0.9962	1	0.9736
158	0.9912	0.9215	1	0.9765
161	0.9682	0.9559	0.9673	0.9706
162	0.9417	0.9713	0.9657	0.9448
163	0.9788	0.9770	0.9886	0.9847
164	0.9505	-	0.9641	0.9571
165	0.9683	0.9521	0.9886	0.9759

170	0.9965	-	0.9951	0.9924
172	0.9982	-	0.9967	0.9929
173	0.9965	1	-	0.9994
174	0.9823	0.9866	0.9788	0.9783
179	0.9912	-	0.9886	0.9900
181	0.9965	0.9866	-	0.9830
182	0.9965	0.9713	-	0.9924
183	0.9876	-	0.9935	0.9965
185	0.9894	0.9904	-	0.9959
189	0.9717	0.9847	0.9869	0.9894
191	0.9647	0.9521	0.9984	0.9689
193	0.9664	-	0.9869	0.9788
194	0.9894	0.9713	-	0.9847
197	0.9700	0.9828	0.9869	0.9812
199	0.9240	0.9885	0.9346	0.9589
202	0.9806	0.9981	0.9886	0.9953
204	0.9929	-	0.9869	0.9965
205	0.9982	0.9904	-	0.9947
212	0.9929	1	0.9984	0.9977
213	0.9947	0.9962	0.9951	0.9965
215	0.9541	0.9732	0.9690	0.9559
216	0.9770	0.9520	0.9821	0.9759
226	0.9753	0.9636	0.9918	0.9636

From Table 4.4 it has been observed that the average recognition accuracy for writer 1 is 97.81%, for writer 2 is 97.59% and for writer 3 is 98.27%. So it can be concluded that writer 3 is the most consistent writer as compared to other writers. An overall accuracy of 98.21% for all writers has been observed.

4.1.2 Results using 10-fold cross validation

10-fold cross validation has been used for recognition of non preprocessed data, for individual writers. Also recognition is done on data combined from all writers.

After applying this cross validation technique, the results are shown in Table 4.5. In Table 4.5 recognition accuracy for each writer has been shown for every stroke id and the recognition accuracy for the combined data is shown in the last column.

Table 4.5: Stroke wise recognition accuracy for each writer with 10-fold cross validation

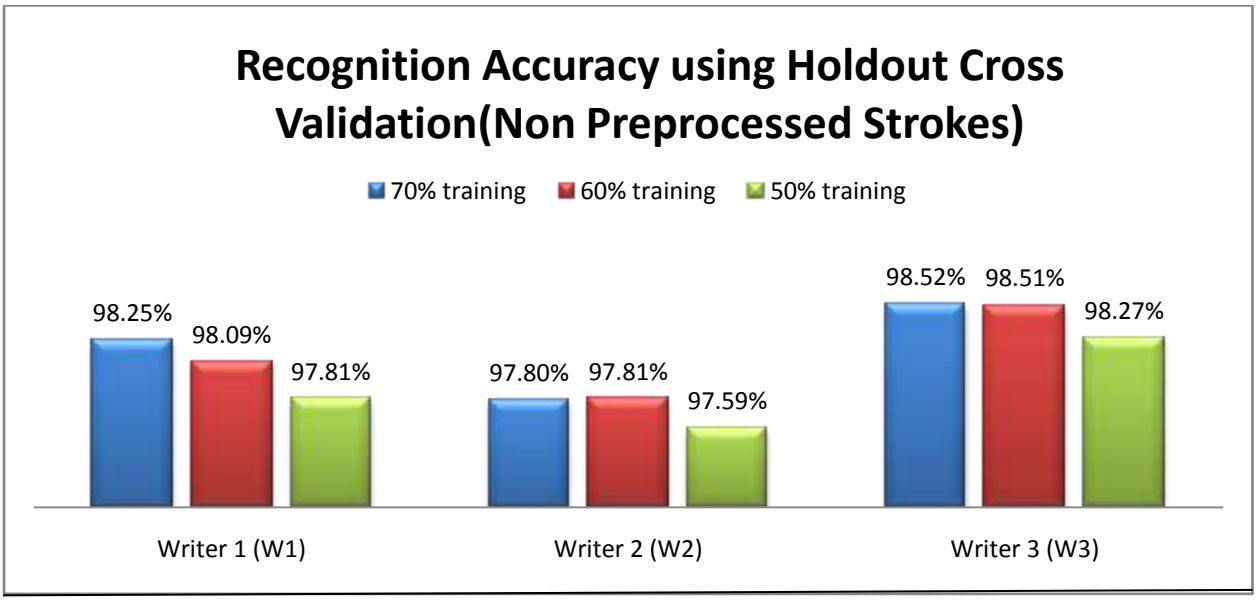
Stroke ID	**Writer 1 (W1)**	**Writer 2 (W2)**	**Writer 3 (W3)**	**W1+W2+W3**
101	0.9975	0.9946	0.9954	0.9963
102	0.9926	1	1	0.9984
105	0.9975	-	-	0.9992
121	0.9778	0.9679	0.9772	0.9636
122	0.9975	0.9920	1	0.9918
124	0.9988	0.9973	0.9989	0.9979
126	0.9988	0.9960	0.9954	0.9955
127	0.9988	1	1	0.9996
128	0.9852	0.9920	0.9932	0.9881
129	0.9963	0.9893	0.9852	0.9914
142	0.9790	0.9880	-	0.9889
144	0.9852	0.9906	-	0.9914
145	0.9963	-	0.9943	0.9856
146	0.9605	0.9853	0.9589	0.9700
147	0.9914	-	0.9874	0.9910
148	0.9864	0.9746	0.9874	0.9835
149	0.9778	0.9759	0.9886	0.9736
151	0.9407	0.9987	0.9658	0.9595
152	0.9914	-	-	0.9971
154	0.9827	0.9009	-	0.9565
155	0.9728	-	0.9726	0/9807
157	0.9605	-	0.9692	0.9689
158	0.9938	0.9545	-	0.9831
161	0.9728	0.9625	0.9760	0.9747
162	0.9506	0.9906	0.9760	0.9457
163	0.9864	0.9853	0.9886	0.9889

164	0.9580	-	0.9658	0.9712
165	0.9741	0.9438	-	0.9733
170	0.9975	-	0.9932	0.9971
172	-	-	-	0.9975
173	-	-	-	0.9979
174	0.9802	0.9893	0.9920	0.9803
179	0.9877	-	0.9932	0.9942
181	-	0.9853	-	0.9955
182	0.9951	0.9772	-	0.9918
183	0.9926	-	0.9943	0.9963
185	0.9864	0.9920	-	0.9942
189	0.9642	0.9920	0.9852	0.9759
191	0.9617	0.9545	-	0.9753
193	0.9741	-	0.9783	0.9856
194	0.9963	0.9639	-	0.9889
197	0.9802	0.9799	0.9863	0.9712
199	0.9395	0.9946	0.9600	0.9630
202	0.9877	0.9946	0.9932	0.9953
204	-	-	0.9954	0.9975
205	0.9975	0.9946	-	0.9934
212	-	-	-	0.9951
213	-	-	-	0.9979
215	0.9741	0.9826	0.9749	0.9692
216	0.9864	0.9759	0.9897	0.9770
226	0.9852	0.9652	0.9897	0.9716

From Table 4.5 it has been observed that the average recognition accuracy for writer 1 is 98.16%, for writer 2 is 98.06% and for writer 3 is 98.53%. So it can be concluded that writer 3 is the most accurate writer as compared to other writers. An overall accuracy of 98.45% for all writers has been observed. In Table 4.5 blank entries signifies either the absence of stroke or due to the reason that recognition process cannot be applied to that stroke because that stroke occurs only one time. It is not possible for 10-fold cross validation process to divide it

in training and testing sets. From all of the above experimental results it can be concluded that for all three partitioning techniques for holdout and for 10-fold cross validation.

Graph 4.1 shows recognition accuracy for three different writers using holdout cross validation for non preprocessed strokes. Three different portioning strategies have been used for generating testing and training sets.



Graph 4.1: Recognition accuracy of 3 writers for non preprocessed strokes

From Graph 4.1 it can be concluded that writer 3 is the most persistent writer amongst all writers because for all the three partitioning strategies writer 3 has highest recognition accuracy for non preprocessed strokes.

4.2 Recognition results using preprocessed data

The strokes discussed in above section are non-preprocessed and in this section, preprocessed strokes are now used for recognition using the same techniques as discussed in the above section. The results of preprocessing technique are strokes with 64 equidistant points.

4.2.1 Results using holdout cross validation

For preprocessed data also, three types of partitions (70% training, 60% training and 50% training) have been used for recognition. These partitions are used for recognition of stroke IDs, for individual writer and also for data combined from all writers.

After applying these partitioning techniques, the results are shown in Table 4.6, Table 4.7 and Table 4.8. In these Tables recognition accuracy for each writer has been shown for every stroke and the recognition accuracy for the combined data is shown in the last column.

Table 4.6: Stroke wise recognition accuracy for each writer with 70% training for preprocessed strokes

Stroke ID	**Writer 1 (W1)**	**Writer 2 (W2)**	**Writer 3 (W3)**	**W1+W2+W3**
101	0.9284	0.9517	0.9703	0.9433
102	0.9950	1.0	1	0.9992
105	0.9975	-	-	0.9992
121	0.9604	0.9544	0.9653	0.9770
122	0.9975	0.9651	0.9748	0.9975
124	0.9975	0.9920	0.9977	0.9930
126	0.9852	0.9893	0.9909	0.9893
127	0.9975	1.0	0.9954	1
128	1	1.0	1	1
129	0.9950	0.9893	0.9772	0.9918
142	0.9877	0.9946	-	0.9967
144	0.9802	0.9866	-	0.9811
145	0.9975	-	0.9886	0.9860
146	0.9926	0.9920	0.9703	0.9868
147	0.9704	-	0.9680	0.9844
148	0.9802	0.9946	0.9931	0.9926
149	0.9926	0.9812	0.9840	0.9877
151	0.9827	0.9973	0.9932	0.9877
152	0.9975	-	-	0.9992
154	0.9876	0.9732	-	0.9918
155	0.9877	-	0.9886	0.9942
157	0.9802	0.9973	0.9863	0.9885
158	0.9951	0.9759	1	0.9918
161	0.9901	0.9705	0.9817	0.9770
162	0.8391	0.9330	0.8950	0.9221

163	0.9703	0.9759	0.9703	0.9885
164	0.9457	-	0.9726	0.9803
165	0.9852	0.9598	0.9977	0.9786
170	0.9975	-	0.9954	0.9975
172	1.0	-	1	1
173	0.9975	0.9973	-	0.9984
174	0.9975	0.9946	0.9954	0.9975
179	0.9950	-	0.9909	0.9926
181	1.0	0.9920	-	0.9984
182	1.0	0.9893	-	0.9942
183	1.0	-	1	0.9992
185	0.9951	0.9866	-	0.9942
189	0.9926	0.9920	0.9817	0.9951
191	0.9802	0.9705	0.9931	0.9770
193	0.9778	-	0.9749	0.9819
194	0.9975	0.9812	-	0.9901
197	0.9778	0.9705	0.9840	0.9762
199	0.8936	0.9705	0.9566	0.9466
202	0.9901	1.0	0.9954	0.9967
204	0.9975	-	0.9977	0.9992
205	0.9975	1.0	-	0.9975
212	0.9975	1.0	1	0.9975
213	1.0	1.0	0.9931	0.9967
215	0.9926	0.9732	0.9977	0.9910
216	0.9876	0.9893	0.9795	0.9794
226	0.9975	0.9705	0.9932	0.9868

From Table 4.6 it has been observed that the average recognition accuracy for writer 1 is 98.38%, for writer 2 is 98.33% and for writer 3 is 98.47%. So it can be concluded that writer 3 is the most consistent writer as compared to other writers. An overall accuracy of 98.73% for all writers has been observed.

Table 4.7: Stroke wise recognition accuracy for each writer with 60% training for preprocessed strokes

Stroke ID	Writer 1 (W1)	Writer 2 (W2)	Writer 3 (W3)	W1+W2+W3
101	0.9280	0.9464	0.9543	0.9507
102	0.9876	0.9978	0.9905	0.9973
105	0.9979	-	-	0.9993
121	0.9134	0.9620	0.9657	0.9486
122	1.0	0.9643	0.9752	0.9836
124	0.9979	0.9888	0.9943	0.9966
126	0.9856	0.9777	0.9886	0.9829
127	0.9979	1.0	1	0.9993
128	1.0	1.0	1	1
129	0.9979	0.9710	0.9695	0.9897
142	0.9815	0.9978	-	0.9973
144	0.9876	0.9911	-	0.9890
145	0.9979	-	0.9867	0.9808
146	0.9876	1.0	0.9924	0.9829
147	0.9546	-	0.9714	0.9808
148	0.9897	0.9933	0.9962	0.9911
149	0.9938	0.9732	0.9886	0.9918
151	0.9897	0.9955	0.9829	0.9822
152	0.9979	-	-	0.9993
154	0.9918	0.9732	-	0.9863
155	0.9959	-	0.9867	0.9883
157	0.9773	0.9978	0.9829	0.9870
158	0.9959	0.9776	1	0.9877
161	0.9835	0.9709	0.9676	0.9822
162	0.8889	0.9664	0.9238	0.9433
163	0.9753	0.9776	0.9543	0.9870
164	0.9567	-	0.9733	0.9801
165	0.9938	0.9598	0.9981	0.9818
170	0.9979	-	0.9981	0.9979

172	1.0	-	1	1
173	0.9979	0.9978	-	0.9986
174	0.9876	1.0	0.9981	0.9973
179	0.9856	-	0.9943	0.9959
181	1.0	0.9978	-	0.9973
182	1.0	0.9776	-	0.9945
183	0.9979	-	1	1
185	0.9897	0.9843	-	0.9918
189	0.9918	0.9866	0.9962	0.9911
191	0.9733	0.9732	0.9943	0.9808
193	0.9897	-	0.9771	0.9877
194	0.9979	0.9642	-	0.9938
197	0.9794	0.9844	0.9886	0.9712
199	0.9074	0.9754	0.9313	0.9557
202	0.9959	1.0	0.9981	0.9986
204	1.0	-	0.9943	0.9993
205	0.9979	0.9978	-	0.9986
212	0.9959	1.0	0.9981	0.9986
213	0.9959	1.0	1	0.9973
215	0.9959	0.9821	0.9962	0.9836
216	0.9856	0.9955	0.9867	0.9829
226	0.9959	0.9306	0.9905	0.9870

From Table 4.7 it has been observed that the average recognition accuracy for writer 1 is 98.44%, for writer 2 is 98.28% and for writer 3 is 98.46%. So it can be concluded that writer 3 is the most promising writer as compared to other writers. An overall accuracy of 98.69% for all writers has been observed.

Table 4.8: Stroke wise recognition accuracy for each writer with 50% training preprocessed strokes

Stroke ID	**Writer 1 (W1)**	**Writer 2 (W2)**	**Writer 3 (W3)**	**W1+W2+W3**
101	0.9224	0.9579	0.9396	0.9477
102	0.9806	0.9923	0.9935	0.9965

105	0.9982	-	-	0.9994
121	0.9576	0.9444	0.9706	0.9601
122	0.9982	0.9732	0.9739	0.9818
124	1.0	0.9828	0.9951	0.9988
126	0.9859	0.9751	0.9837	0.9859
127	1.0	0.9943	0.9984	1
128	1.0	1.0	1	1
129	0.9912	0.9713	0.9755	0.9730
142	0.9929	0.9943	-	0.9965
144	0.9841	0.9962	-	0.9806
145	0.9947	-	0.9804	0.9835
146	0.9682	0.9885	0.9902	0.9835
147	0.9523	-	0.9739	0.9800
148	0.9859	0.9866	0.9918	0.9912
149	0.9947	0.9847	0.9902	0.9888
151	0.9947	0.9981	0.9886	0.9871
152	1.0	-	-	0.9988
154	0.9894	0.9789	-	0.9894
155	0.9894	-	0.9935	0.9859
157	0.9735	0.9962	0.9918	0.9847
158	0.9894	0.9579	1	0.9882
161	0.9929	0.9617	0.9510	0.9706
162	0.8463	0.9502	0.8725	0.9321
163	0.9770	0.9540	0.9722	0.9882
164	0.9541	-	0.9739	0.9683
165	0.9912	0.9636	0.9967	0.9900
170	0.9982	-	0.9984	0.9971
172	1.0	-	1	0.9988
173	0.9965	0.9981	-	0.9976
174	0.9965	0.9981	0.9951	0.9941
179	0.9894	-	1	0.9935
181	0.9982	0.9904	-	0.9982

182	1.0	0.9847	-	0.9953
183	0.9965	-	0.9967	0.9988
185	0.9947	0.9808	-	0.9935
189	0.9965	0.9943	0.9853	0.9941
191	0.9788	0.9483	0.9935	0.9806
193	0.9647	-	0.9788	0.9759
194	0.9982	0.9770	-	0.9918
197	0.9823	0.9751	0.9820	0.9771
199	0.8746	0.9732	0.9444	0.9322
202	0.9965	0..9981	0.9951	0.9965
204	0.9982	-	0.9967	0.9994
205	0.9982	1.0	-	0.9982
212	0.9929	1.0	1	0.9988
213	0.9947	1.0	0.9967	0.9982
215	0.9823	0.9713	0.9918	0.9865
216	0.9841	0.9885	0.9772	0.9777
226	0.9982	0.9483	0.9902	0.9783

From Table 4.8 it has been observed that the average recognition accuracy for writer 1 is 98.26%, for writer 2 is 97.79% and for writer 3 is 98.29%. So it can be concluded that writer 3 is the most promising writer as compared to other writers. An overall accuracy of 98.68% for all writers has been observed.

4.2.2 Results using 10-fold cross validation

For preprocessed data another technique for 10-fold cross validation is used instead of holdout cross validation. The recognition is done for individual writer as well as clustered writer.

After applying this cross validation technique, the results are shown in Table 4.9. In Table 4.9 recognition accuracy for each writer has been shown for every stroke ID and the recognition accuracy for the combined data is shown in the last column.

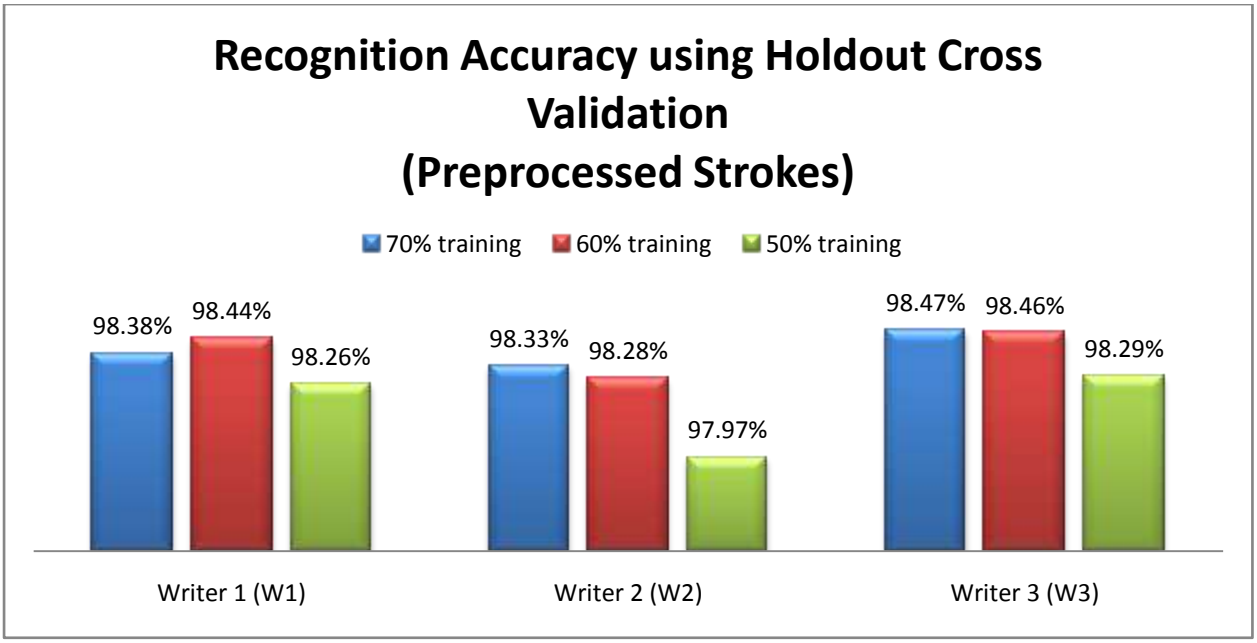
Table 4.9: Stroke wise recognition accuracy for each writer with 10-fold cross validation

Stroke ID	Writer 1 (W1)	Writer 2 (W2)	Writer 3 (W3)	W1+W2+W3
101	0.9370	0.9598	0.9715	0.9595
102	0.9926	0.9987	1	0.9979
105	0.9975	-	-	0.9992
121	0.9630	0.9692	0.9703	0.9721
122	1	0.9746	0.9726	0.9888
124	0.9963	0.9933	0.9954	0.9955
126	0.9926	0.9826	0.9909	0.9889
127	0.9988	1	1	1
128	1	1	1	1
129	0.9951	0.9799	0.9806	0.9897
142	0.9926	0.9973	-	0.9955
144	0.9889	0.9866	-	0.9893
145	0.9975	-	0.9852	0.9897
146	0.9889	0.9960	0.9840	0.9918
147	0.9790	-	0.9749	0.9744
148	0.9926	0.9893	0.9932	0.9926
149	0.9975	0.9893	0.9932	0.9934
151	0.9877	0.9960	0.9852	0.9897
152	1	-	-	0.9988
154	0.9938	0.9732	-	0.9918
155	0.9901	-	0.9909	0.9897
157	0.9877	-	0.9909	0.9881
158	0.9951	0.9772	-	0.9897
161	0.9951	0.9719	0.9760	0.9776
162	0.8938	0.9612	0.9598	0.9298
163	0.9765	0.9813	0.9852	0.9761
164	0.9667	-	0.9817	0.9790
165	0.9877	0.9746	-	0.9905
170	0.9975	-	0.9966	0.9975
172	-	-	-	0.9992

173	-	-	-	0.9984
174	0.9963	0.9987	0.9977	0.9975
179	0.9914	-	0.9977	0.9947
181	-	0.9960	-	0.9984
182	-	0.9839	-	0.9942
183	0.9975	-	0.9989	0.9988
185	0.9951	0.9880	-	0.9955
189	0.9938	0.9960	0.9932	0.9918
191	0.9728	0.9652	-	0.9790
193	0.9827	-	0.9795	0.9897
194	0.9951	0.9813	-	0.9926
197	0.9802	0.9853	0.9840	0.9827
199	-	0.9826	0.9406	0.9665
202	0.9951	0.9987	0.9989	0.9975
204	-	-	0.9977	0.9996
205	0.9963	1	-	0.9975
212	-	-	-	0.9984
213	-	-	-	0.9988
215	0.9988	0.9786	0.9977	0.9885
216	0.9889	0.9880	0.9920	0.9889
226	0.9975	0.9746	0.9874	0.9873

From Table 4.9, it has been observed that the average recognition accuracy for writer 1 is 98.75% for writer 2 is 98.48% and for writer 3 is 98.73%. So it can be concluded that writer 3 is the most accurate writer as compared to other writers. An overall accuracy of 98.92% for all writers has been observed.

Graph 4.2 shows recognition accuracy for three different writers using SVM using three different partitioning strategies for holdout cross validation.



Graph 4.2: Recognition accuracy of three writers for preprocessed strokes

From Graph 4.2 it has been observed that writer 2 is the most inconsistent writer amongst all writers, writer 1 is slightly better than writer 2, and writer 3 is the best one. So here it can be determined that there is minimum variation in handwriting of writer 3.

4.3 Recognition results using preprocessed data on specific number (1000, 2000 and 3000) of strokes

In this section, the recognition process is applied to specific number of stroke for each writer. In this study 1000, 2000 and 3000 preprocessed strokes are considered from each writer. These preprocessed strokes were taken randomly from the stroke database. The results are calculated only for those stroke IDs present in 1000 strokes. The recognition results of these strokes are given in Table 4.10, Table 4.11 and Table 4.12 for each writer respectively. In the Table 4.10, Table 4.11 and Table 4.12, T_{s1000}, T_{s2000} and T_{s3000} represents 1000, 2000 and 3000 strokes respectively.

Table 4.10: Recognition result of 1000, 2000 and 3000 preprocessed strokes for writer 1

Stroke ID	T_{s1000}	T_{s2000}	T_{s3000}
101	0.9680	0.9620	0.9593

102	1	1	1
103	0.9940	0.9980	0.9977
104	1	0.9985	1
121	0.9690	0.9785	0.9783
122	1	0.9995	0.9997
123	0.9990	0.9970	0.9997
124	0.9970	0.9930	0.9993
125	0.9870	0.9980	0.9900
126	0.9910	0.9980	0.9933
127	1	1	0.9993
128	1	1	1
129	0.9950	0.9540	0.9657
142	0.9975	0.9980	0.9990
144	0.9980	0.9965	0.9973
146	0.9990	0.9980	0.9983
147	0.9810	0.9725	0.9880
148	0.9980	0.9970	0.9970
149	0.9980	0.9935	0.9960
151	0.9980	0.9980	0.9968
154	0.9870	0.9825	0.9897
155	0.9980	0.9980	0.9988
156	0.9960	0.9835	0.9912
157	0.9930	0.9965	0.9980
161	0.9860	0.9880	0.9880
163	0.9680	0.9635	0.9687
164	0.9970	0.9965	0.9947
165	0.9780	0.9814	0.9827
167	0.9960	0.9975	0.9973
170	0.9990	0.9995	0.9997
171	0.9610	0.9835	0.9750
173	0.9960	0.9971	0.9987
174	1	1	1

181	0.9960	0.9835	0.9923
183	0.9990	0.9970	0.9993
184	0.9970	0.9965	0.9960
185	0.9890	0.9905	0.9820
189	0.9940	0.9985	0.9977
191	0.9690	0.9710	0.9808
193	0.9977	0.9955	0.9993
194	0.9670	0.9815	0.9827
197	0.9720	0.9780	0.9791
201	0.9930	0.9925	0.9817
202	0.9990	0.9985	0.9990
205	0.9980	0.9965	0.9963
212	0.9900	0.9870	0.9827
215	0.9970	0.9995	0.9950
216	0.9970	0.9975	0.9957

Table 4.10 shows stroke IDs present in 1000 strokes and recognition accuracy of each stroke ID present in 1000, 2000 and 3000 strokes for writer 1. For 1000, 2000 and 3000 strokes 99.12%, 99.08% and 99.16% accuracy has been achieved respectively. Table 4.11 demonstrates the results of recognizing 1000, 2000 and 3000 preprocessed strokes for writer 2.

Table 4.11: Recognition result of 1000, 2000 and 3000 preprocessed strokes for writer 2

Stroke ID	T_{s1000}	T_{s2000}	T_{s3000}
101	0.9620	0.9740	0.9613
102	0.9990	0.9990	0.9993
104	1	1	1
121	0.9800	0.9835	0.9827
124	0.9900	0.9925	0.9937
125	0.9940	0.9935	0.9897
126	0.9920	0.9780	0.9797

127	0.9990	0.9990	0.9993
128	1	1	1
129	0.9940	0.9810	0.9773
130	0.9890	0.9975	0.9933
142	0.9993	0.9995	0.9993
144	0.9980	0.9980	0.9987
146	0.9970	0.9985	0.9977
147	0.9930	0.9870	0.9907
148	0.9910	0.9940	0.9960
149	0.9970	0.9955	0.9960
156	0.9990	0.9820	0.9853
158	0.9600	0.9620	0.9863
161	0.9880	0.9885	0.9893
163	0.9710	0.9450	0.9430
165	0.9810	0.9765	0.9783
171	0.9830	0.9975	0.9900
173	0.9930	0.9990	0.9953
174	1	0.9990	0.9990
178	0.9940	0.9960	0.9970
181	0.9890	0.9690	0.9947
182	0.9730	0.9845	0.9833
184	0.9990	0.9970	0.9970
185	0.9920	0.9840	0.9840
188	0.9960	0.9940	0.9970
189	1	0.9995	0.9990
191	0.9730	0.9790	0.9857
194	0.9290	0.9670	0.9844
197	0.9630	0.9600	0.9840
199	0.9930	0.9960	0.9967
201	0.9970	0.9945	0.9950
202	1	0.9985	0.9997
205	0.9950	0.9970	0.9970

207	1	1	1
212	0.9890	0.9840	0.9803
213	0.9770	0.9850	0.9797
215	0.9970	0.9940	0.9947
216	0.9940	0.9955	0.9943
226	0.9720	0.9760	0.9890
227	0.9960	0.9970	0.9970
229	0.9840	0.9790	0.9923

From Table 4.11 it has been observed that for 3000 preprocessed strokes, writer 2 has attained better recognition accuracy as compare to 1000 and 2000 strokes. For 1000, 2000 and 3000 preprocessed strokes in case of writer 2 the recognition accuracy is 98.83%, 98.82% and 99.02% respectively. Similarly Table 4.12 also shows the result of writer 3 for 1000, 2000 and 3000 preprocessed strokes.

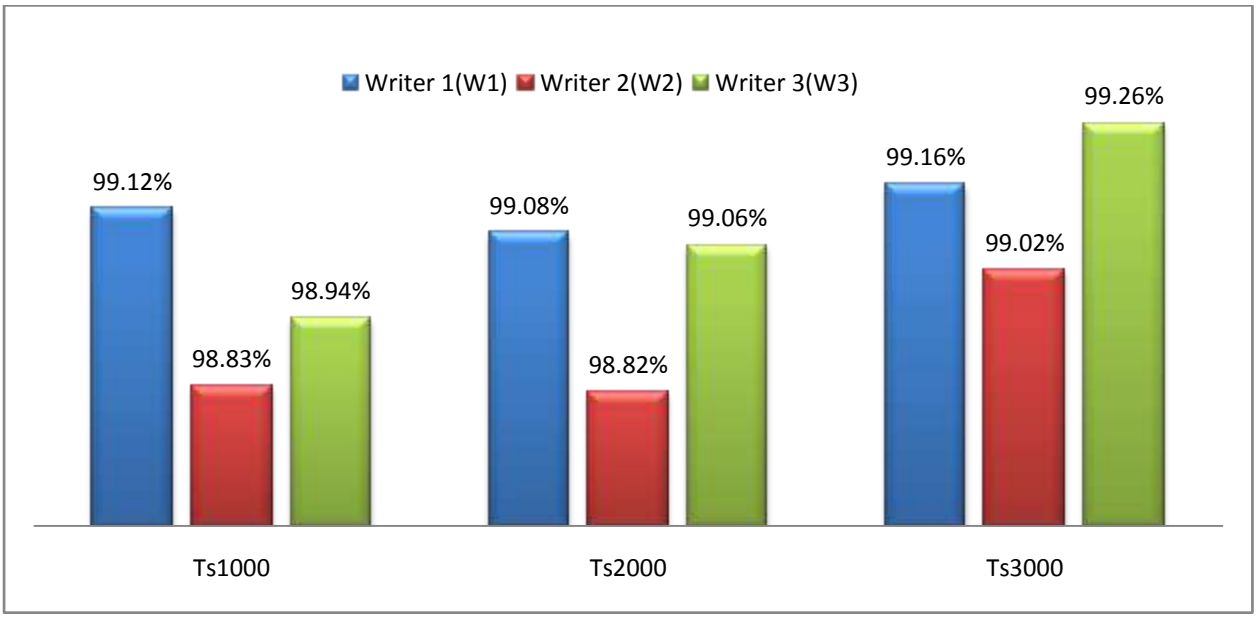
Table 4.12: Recognition result of 1000, 2000 and 3000 preprocessed strokes for writer 3

Stroke ID	T_{s1000}	T_{s2000}	T_{s3000}
101	0.9540	0.9615	0.9640
102	0.9990	1	1
103	1	1	0.9997
104	0.9987	0.9985	0.9990
121	0.9760	0.9885	0.9880
122	0.9640	0.9755	0.9810
124	0.9940	0.9955	0.9977
125	0.9970	0.9945	0.9920
126	0.9980	0.9885	0.9917
127	0.9960	0.9965	0.9983
128	1	1	1
129	0.9610	0.9595	0.9563
130	0.9920	0.9905	0.9940
141	1	0.9990	1

145	0.9900	0.9900	0.9913
146	0.9750	0.9785	0.9843
148	0.9780	0.9870	0.9917
149	0.9900	0.9885	0.9920
151	0.9970	0.9985	0.9993
155	0.9930	0.9960	0.9967
157	0.9890	0.9895	0.9940
161	0.9800	0.9700	0.9843
163	0.9820	0.9865	0.9787
164	0.9700	0.9785	0.9793
165	0.9960	0.9965	0.9980
167	0.9940	0.9955	0.9970
170	1	1	0.9997
174	0.9980	0.9980	0.9993
178	0.9970	0.9980	0.9990
179	0.9900	0.9930	0.9937
183	0.9950	0.9940	0.9980
186	0.9950	0.9965	0.9977
187	0.9980	0.9995	1
189	0.9900	0.9920	0.9947
192	0.9910	0.9975	0.9947
193	0.9860	0.9910	0.9923
197	0.9840	0.9800	0.9847
200	0.9950	0.9950	0.9973
202	0.9980	0.9980	0.9987
213	0.9760	0.9725	0.9890
214	0.9860	0.9915	0.9947
215	1	0.9990	0.9990
216	0.9920	0.9875	0.9910
226	0.9920	0.9955	0.9977
227	0.9990	0.9995	0.9993

For writer 3 recognition accuracy of 1000, 2000 and 3000 preprocessed strokes 98.94%, 99.06% and 99.26% respectively. Also it has been found that the recognition accuracy of 3000 preprocessed strokes is maximum.

Graph 4.3 shows recognition accuracy of 1000, 2000 and 3000 preprocessed strokes for three different writers using 10-fold cross validation technique.



Graph 4.3: Recognition accuracy of three writers for 1000, 2000 and 3000 preprocessed strokes

For each writer, recognition using 3000 preprocessed strokes gives maximum accuracy as compared to 1000 and 2000 preprocessed strokes. Out of the three writers, writer 3 gives highest accuracy for 1000, 2000 and 3000 preprocessed strokes.

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

On-line handwriting recognition started in the sixties, as discussed in literature survey. The first generation of tablets became available during this period. These tablets were less accurate and unreliable and the time for computing was very large. This resulted in less accuracy for an online handwriting recognition system. As advancements were made in the hardware there arises a need of better handwriting recognition system because people need pen-paper like interface for input a text on a computer. A lot of work has been done in languages like English, Japanese, Chinese, Arabic, Urdu, Devnagiri and Tamil till now. This thesis focuses on developing a stroke recognition system for Gurmukhi script using SVM.

100 words from three different writers are considered for this study. Each of these words are divided into strokes and each stroke is given a unique ID. These strokes are divided into three zones namely Upper Zone, Middle Zone and Lower Zone. Writer 1 writes 810 strokes, writer 2 writes 743 strokes and writer 3 writes 875 strokes for the same 100 words.

We used SVM to perform Gurmukhi stroke recognition for both non preprocessed and preprocessed strokes. Two different partitioning techniques holdout and 10-fold cross validation have been used for generating training and testing sets. We have done recognition 70%, 60% and 50% training sets in holdout cross validation for both non preprocessed and preprocessed data.

For 70%, 60% and 50% training set of non preprocessed stroke in holdout cross validation method, writer 3 attain better accuracy of 98.52%, 98.51% and 98.27% respectively. For 70%, 60% and 50% training set of preprocessed stroke in holdout cross validation method also, writer 3 has better accuracy 98.47%, 98.46% and 98.29% respectively. For 10-fold cross validation writer 3 attains 98.73% for non preprocessed strokes and in the case of preprocessed stroke writer 3 got 98.53% accuracy. We have considered 1000, 2000 and 3000 preprocessed strokes and examined the recognition accuracy for each writer. For writer 1, writer 2 and writer 3 recognition accuracies 99.16%, 99.02% and 99.26% is achieved with 3000 preprocessed strokes which are higher as compared to 1000 and 2000 preprocessed strokes. Writer 3 attains better recognition accuracy 99.26% amongst all the writers for 1000,

2000 and 3000 preprocessed strokes. From this, we can conclude that writer 3 has achieved maximum accuracy for Gurmukhi strokes in each situation. So it can be concluded that writer 3's handwriting is better writer as compare to others. In this study, it has been observed that SVM gives good recognition result for non preprocessed strokes and also recognition accuracy rate depends on the writer and his writing style.

However, there are still many areas for improvement in this work. These are listed below**:**

- The work from stroke level recognition can be extended to character and word level recognition.
- The stroke database can be increased and examined for getting more recognition results.
- Kernel level functions in SVM can be used and accuracy rate can be examined.
- Another tool in SVM like statisctica, LibSVM and SVMlight *etc.* can also be used.
- Some more features and preprocessing techniques can be introduced for further recognition results.

REFERENCES

1. Artieres, T. and Gallinari, P., 2002. Stroke level HMMs for online handwriting recognition. Proceedings of IWFHR, pp. 227-232.
2. Basu, M., Bunke, H. and Bimbo, A. D., 2005. Guest editors' introduction to the special section on syntactic and structural pattern recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 7, pp. 1009-1012.
3. Bontempi, B. and Marcelli, A., 1994. A genetic learning system for on-line character recognition, Proceedings of 12th IAPR International Conference on Pattern Recognition, vol. 2, pp. 83-87.
4. Brakensiek, A., Kosmala, A., and Rigoll G., 2002. Evaluation of Confidence Measures for On-Line Handwriting Recognition, Springer-Berlin Heidelberg, pp. 507–514.
5. Brown M. K. and Ganapathy S., 1983. Preprocessing techniques for cursive script word recognition, Pattern Recognition, vol. 16, pp. 447-458.
6. Chan, K. F. and Yeung, D. Y. 1999. Recognizing on-line handwritten alphanumeric characters through flexible structural matching, Pattern Recognition, vol. 32, no. 7, pp. 1099-1114.
7. Cho, S., 1997. Neural-network classifiers for recognizing totally unconstrained handwritten numerals, IEEE Transactions on Neural Networks, vol. 8, no. 1, pp. 43-53.
8. Connell, S. D. and Jain A. K., 2001. Template-based online character recognition, Pattern Recognition, vol. 34, no. 1, pp. 1-14.
9. Connell, S. D. and Jain, A. K., 2002. Writer adaptation for online handwriting recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 3, pp. 329-346.
10. Duneau, L. and Dorizzi, B., 1994. Online cursive script recognition: a system that adapts to an unknown user, Proceedings of International Conference on Pattern Recognition, vol. 2, pp. 24-28.

11. Farag, R. F., 1979. Word-level recognition of cursive script, IEEE Transactions on Computers, vol. 28, no. 2, pp. 172-175.
12. Fujisawa, H., 2001. Forty years of research in character and document recognition-an industrial perspective, Pattern Recognition, vol. 41, no. 8, pp. 2435-2446.
13. Funanda, A., Muramatsu, D., Matsumoto, T., 2004. The reduction of memory and the improvement of recognition rate for HMM on-line handwriting recognition, Proceedings of IWFHR, pp. 383-388.
14. Gunter, S. and Bunke, H., 2004. Combination of three classifiers with different architectures for handwritten word recognition, Proceedings of IWFHR, pp. 63-68.
15. Jindal, M. K., Sharma, R. K. and Lehal, G. S., 2006. Segmentation of Horizontally Overlapping lines in Printed Gurmukhi Script, Advanced Computing and Communications, 2006. ADCOM. International Conference, pp. 226-229.
16. Jindal, M. K., Sharma, R. K. and Lehal, G. S., 2008. Structural Features for Recognizing Degraded Printed Gurmukhi Script, Fifth International Conference on Information Technology: New Generations, pp. 668-673.
17. Hu, J., Brown, M. K. and Turin, W., 1996. HMM based on-line handwriting recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 10, pp. 1039-1045.
18. Hu, J., Lim, S. G. and Brown, M. K., 2000. Writer independent on-line handwriting recognition using an HMM approach, Pattern Recognition, vol. 33, no. 1, pp. 133-147.
19. Impedovo S., 1984. Plane curve classification through Fourier descriptors: An application to Arabichand-written numeral recognition, In Proceedings of 7th International Conference on Pattern Recognition, pp. 1069-1072.
20. 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.
21. Kimura, Y., Wakahara, T. and Odaka, K., 1997. Combining statistical pattern recognition approach with neural networks for recognition of large-set categories, Proceedings of International Conference on Neural Networks, vol. 3, pp. 1429-1432.
22. Kobayashi, M., Masaki, S., Miyamoto, O., Nakagawa, Y., Komiya, Y. and Matsumoto, T., 2001. Reparametrized Angle Variations algorithm for online handwriting Recognition, International Journal of Document Analysis and Recognition, vol. 3, no. 1, pp. 181-191.

23. Kumar, M., Jindal, M. K. and Sharma, R. K., 2011. Classification of Characters and Grading Writers in Offline Handwritten Gurmukhi Script, International Conference on Image Information Processing. pp. 1-4.
24. Kurtzberg J. M. and Tappert C. C., 1982. Segmentation procedure for handwritten symbols and words. IBM Tech, Disclosure Bull, vol. 25, pp. 3848-3852.
25. Kurtzberg, J. M., 1987. Feature analysis for symbol recognition by elastic matching, IBM Journal of Research and Development, vol. 31, no. 1, pp. 91-95.
26. Li, X. and Yeung, D.Y., 1997. Online handwritten alphanumeric character recognition using dominant points in strokes, Pattern Recognition, vol. 30, no. 1, pp. 31-44.
27. Kim, H. Y. and Kim, J. H., 2001. Hierarchical random graph representation of handwritten character and its application to Hangul recognition, Pattern Recognition, vol. 34, no. 2, pp. 187-201.
28. Mori, S., Suen, C. Y. and Kamamoto, K., 1992. Historical review of OCR research and development, Proceedings of IEEE, vol. 80, no. 7, pp. 1029-1058.
29. Nakai, M., Akira, N., Shimodiara, H., Sagayama, S., 2001. Substroke approach to hmm-based online Kanji handwriting recognition, Proceedings of International Conference on Document Analysis and Recognition, pp. 491-495.
30. Nouboud, F. and Plamondon, R., 1991. A structural approach to online character recognition: System design and applications, International Journal of Pattern Recognition and Artificial Intelligence, vol. 5, no. 1/2, pp. 311-335.
31. Paul, R., Nasif, M. S. and Farhad, S. M., 2007. Fingerprint recognition by chain coded string matching technique, Proceedings of International Conference on Information and Communication Technology, pp. 64-67.
32. Plamondon, R. and Srihari, S. N., 2000. On-line and off- line handwritten character recognition: A comprehensive survey, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, pp. 63-84.
33. Powalka, R. K., Sherkat, N., and Whitrow, R. J., 1994. The Use of Word Shape Information for Cursive Script Recognition, Fourth International Workshop on Frontiers of Handwriting Recognition, Taiwan, pp. 67-76.
34. Rabiner, L. R., 1989. A tutorial on hidden Markov models and selected applications in speech recognition, Proceedings of IEEE, vol. 77, no. 2, pp. 257-286.
35. Rigoll, G., Kosmala, A., Rottland, J. and Neukirchen, C., 1996. A comparison between continuous and discrete density hidden Markov models for cursive

handwriting recognition, Proceedings of International Conference on Pattern Recognition, vol. 2, pp. 205-209.

36. Scattolin, P., 1995, Recognition of handwritten numerals using elastic matching, Master's thesis, Concordia University, Canada.
37. Schomaker, L., 1993. Using stroke or character based self-organizing maps in the recognition of online, connected cursive script. Pattern Recognition, vol. 26, no. 3, pp. 443-450.
38. Schplachbach, A. and Bunke, H., 2004. Using HMM based recognizers for writer identification and verification, Proceedings of IWFHR, pp. 167-172.
39. Senior, A. and Nathan, K., 1997. Writer adaptation of a HMM handwriting recognition system, Proceedings of International Conference on Acoustics, Speech, and Signal Processing, vol. 2, pp. 1447-1450.
40. Sharma A., 2009. Online handwritten Gurmukhi character recognition, Ph.D. thesis, School of Mathematics and Computer Applications, Thapar University, Patiala.
41. Sharma, A., Sharma, R. and Sharma, R. K., 2008. Online Handwritten Gurmukhi Character Recognition Using Elastic Matching, Congress on Image and Signal Processing, pp. 391-396.
42. Sharma, A., Sharma, R. and Sharma, R. K., 2009. Rearrangement of Recognized Strokes in Online Handwritten Gurmukhi Words Recognition, 10th International Conference on Document Analysis and Recognition, pp. 1241-1245.
43. Shimodaira, H., Sudo, T., Nakai, M., Sagayama, S., 2003. Online overlaid-handwriting recognition based on substroke HMMs, Proceedings of International Conference on Document Analysis and Recognition, pp. 1043-1047.
44. Shintani, H., Akutagawa, M., Nagashino, H., Kinouchi, Y., 2005. Recognition mechanism of a neural network for character recognition, Proceedings of Engineering in Medicine and Biology 27th Annual Conference, pp. 6540-6543.
45. Spitz, A.L., 1999. Shape-based word recognition, International Journal of Document Analysis and Recognition, vol. 1, pp. 178-190.
46. Suen, C. Y., Berthod, M., and Mori, S., 1980. Automatic Recognition of Hand printed Character-the State of the Art, Proceedings of IEEE, vol. 68, no. 4, pp. 469-487.
47. Tappert, C.C., Suen, C.Y. and Wakahara, T., 1990. The state of art in online handwriting recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12 no. 8, pp. 787-808.

48. Veltman, S. R. and Prasad, R., 1994. Hidden markov models applied to online handwritten isolated character recognition, IEEE Transactions on Image Processing, vol. 3, no. 3, pp. 314-318.
49. Wakahara, T. and Odaka, K., 1997. Online cursive kanji character recognition using stroke-based affine transformation. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, no. 12, pp. 1381-1385.
50. Ward, J.R. and Kuklinski, T., 1988. A model for variability effects in handwriting with implications for design of handwriting character recognition systems, IEEE Transactions on systems, Man, and Cybernetics, vol. 18, no. 3, pp. 438-451.
51. Zhou, J., Gan, Q., Krzyzak, K., Suen, C.Y, 1999. Recognition of handwritten numerals by quantum neural network with fuzzy features, International Journal of Document Analysis and Recognition, vol. 2, pp. 30-36.