AN OFF-LINE CHARACTER RECOGNITION SYSTEM FOR FREE STYLE HANDWRITING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

NAFİZ ARICA

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE

DEGREE OF

MASTER OF SCIENCE

IN

THE DEPARTMENT OF COMPUTER ENGINEERING

SEPTEMBER 1998

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Tayfur Öztürk
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Fatoş Yarman Vural
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Prof. Dr. Fatoş Yarman Vural
Supervisor

Examining Committee Members

Prof. Dr. Fatoş Yarman Vural _____

Assoc. Prof. Dr. Faruk Polat (Chairman) _____

Assoc. Prof. Dr. Kemal Leblebicioğlu _____

Assoc. Prof. Dr. İsmail Hakkı Toroslu _____

Assist. Prof. Dr. Halit Oğuztüzün _____

ABSTRACT

AN OFF-LINE CHARACTER RECOGNITION SYSTEM FOR FREE STYLE HANDWRITING

Arica, Nafiz

MS., Department of Computer Engineering

Supervisor: Prof. Dr. Fatoş Yarman Vural

September 1998, 113 pages

In this study, a new scheme is proposed for off-line handwritten connected character recognition problem, which uses a sequence of segmentation and recognition algorithms. The proposed system assumes no constraint in writing style, size or variations. First, a segmentation method finds the character segmentation paths by combining the gray scale and binary information. Then, a new set of features is extracted from the segments. The parameters of the feature set are adjusted, dynamically. The features are fed to a high order Hidden Markov Model (HMM), as a preliminary recognition step. Finally, in order to confirm the segmentation paths and recognition results, a recognition based segmentation method is presented. The proposed scheme is tested on various international and local handwritten digit and character databases. It is observed that the proposed system achieves very high recognition rates compared to the available systems in the literature.

Keywords: Handwritten Character Recognition, Hidden Markov Models, Segmentation

ÖZ

SERBEST STİL EL YAZISI İÇİN ÇEVRİM DIŞI KARAKTER TANIMA SİSTEMİ

Arica, Nafiz

Yüksek Lisans, Bilgisayar Mühendisliği Bölümü

Tez Yöneticisi: Prof. Dr. Fatoş Yarman Vural

Eylül 1998, 113 sayfa

Bu çalışmada *El Yazısı Tanıma* problemi için yeni bir algoritma geliştirilmiştir. Bu algoritma bir dizi ayrıştırma ve tanıma yöntemini kullanırken, yazı stili ve çeşitliliği üzerinde hiçbir varsayım gerektirmez. Önce bir ayrıştırma yöntemi yazıyı karakterlerine böler. İlk aşamada bu karakterlerin çoğu doğru bulunmakla birlikte daha sonraki aşamalarda düzeltilmesi gereken bazı hatalar da vardır. Şimdiye kadar yapılan çalışmalardan farklı olarak bu ayrıştırma yöntemi gri seviyeli ve ikili görüntü üzerindeki bilgileri bir arada kullanır. İkinci aşamada her bir ayrıştırma aralığındaki mustakbel karakter üzerinde bir set öznitelik çıkarılır. Önerilen öznitelik uzayı iki boyutlu görüntüyü tek boyutlu sembollere dönüştürür. Bu yapı Saklı Markov Modeli ile geliştirilen tanıma problemi için çok uygundur. Elde edilen öznitelikler yüksek dereceli bir Saklı Markov Modeli aracılığı ile önce öğrenilir sonra da tanınır. Son olarak ise ayrıştırma ve tanıma aşamalarındaki hatalar tanıma bazlı bir ayrıştırma yöntemi ile düzeltilir. Önerilen yöntem bazı uluslararası ve yerel el yazısı veri tabanlarında denenmiş ve yüksek başarı oranı elde edilmiştir.

Anahtar Kelimeler: El Yazısı Tanıma,Saklı Markov Modelleri, Ayrıştırma

ACKNOWLEDGMENTS

First of all, I would like to thank Prof. Dr. Fatoş Yarman Vural for her valuable contributions, corrections and discussions.

Also, I would like to thank the members of the Image Processing Laboratory for their encouragement and support during the preparation of this thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Over the past decade, electronic document management systems have been of great benefit to society. Software tools such as word processors, computer aided design (CAD) packages, drawing programs, and mark-up languages are extensively used in the generation, storage and retrieval of documents in a format understood by computers. In this format a document can be easily edited, copied on papers, or may be distributed electronically across world-wide networks. Additional processing tools are available when the document is in a computer readable or understandable format. These tools include keyword or pattern searching of lengthy documents, applying optimization algorithms or simulations on things such as electronic circuit designs, or improving the visual quality of the pages of a book or a photograph by removing noise that could be the result of years of decay.

On the other hand, when the documents are on paper, none of the above tasks can be accomplished by the use of computers. There is a bulk amount of documents on papers that are to be processed for various reasons. It has been estimated that approximately 250 billion USD per year is spent, worldwide, on keying information from paper documents -and this is for keying only 1% of the available documents [8]. Most of this cost is in human labour. When the process of data entry is automated, significant cost savings can be realized. In addition, the percentage of data that is brought on line can be drastically increased. In all of the applications, the major goal is to extract the information contained in the documents and to store them in a computerized format. This is the motivation of electronic document analysis systems.

Optical Character Recognition (OCR) is the most crucial part of Electronic Document Analysis Systems. The solution lies in the intersection of the fields of pattern recognition, image and natural language processing. Although there has been a tremendous research effort, the state of the art in the OCR has only reached the point of partial use in recent years. Nowadays, cleanly printed text in documents with simple layouts can be recognized reliably by off-the-shelf OCR software. There is only limited success in handwriting recognition, particularly for isolated and neatly hand-printed characters and words for limited vocabulary. However, in spite of the intensive effort of more than thirty years, the recognition of free style and cursive handwriting (as in the example of Figure 1.1) continues to remain in the research arena.

Over the last few years, the number of research laboratories and companies involved in research on handwriting recognition has continually increased. At the same time, commercial products have become available. This new stage in the evolution of handwriting processing results from a combination of several elements: improvements in recognition algorithms, the use of complex systems integrating several kinds of information, the choice of limited application domains, and availability of the new technologies such as high quality, high speed scanners and inexpensive powerful CPUs.

The difficulty of the free style handwriting recognition is not only because of the great amount of variations involved in the shape of characters, but also, because of the overlapping and the interconnection of the neighboring characters. In addition to the pecularities of an author's idioscript, which mean that one writer can be identified among thousands, there are the peculiarities of writing in different situations, with different media and for different purposes. Sometimes, it may not be possible to extract the characters from the whole word. Furthermore, when observed in isolation, characters are often ambiguous and require context to minimize the recognition errors.

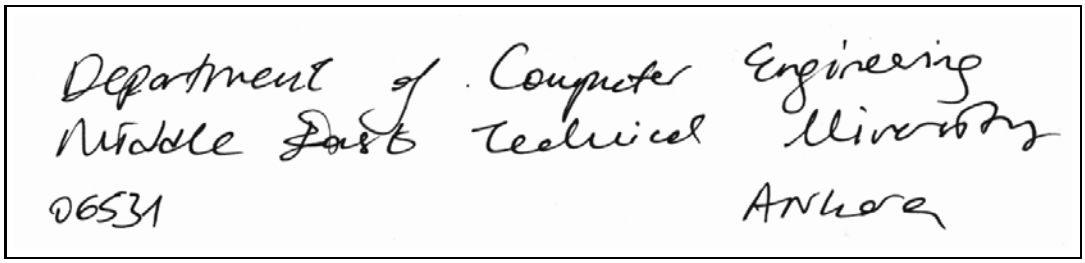


Figure 1.1: An example of handwriting

In order to reach the ultimate goal of fluent machine reading many subproblems have

been investigated applying some constraints to the recognition system. One constraint may be imposed on the data acquisition equipment, by using a digitizer as in the on-line recognition problem, where the system captures the data during the writing and the recognition task is performed concurrently. This approach avoids complicated pre-processing operations and captures the temporal information. On the other hand, the input in the off-line recognition systems is a pre-written document image that is converted into a bit pattern data through an optical scanner. The data is contaminated with various sources of noise. Another constraint is the restriction of the system to the writing of a specific user. In this case, the system is "writer dependent". It is possible to make the system learn accurately the writing style of the user and have a satisfactory performance in recognizing a specific writing. Thirdly, the size of the vocabulary can be limited. The system looks for the whole word in the lexicon, which best fits the unknown word. Thus, it avoids the recognition errors due to imperfect segmentation and increases the recognition rates. Lastly, some constraints can be applied on the writing style using pre-printed guidelines or boxes on forms or envelopes.

Therefore, the methods and the success of the recognition rates depend on the level of constraints on handwriting. Considering the roman script, the difficulty is less for handwriting produced as a sequence of separate characters than for cursive script. For other writing systems, character recognition is hard to achieve, as in the case of Chineese characters, which is characterized by complex shapes and a huge number of symbols. In the case of Arabic characters, the situation gets even more complicated due to the cursive concatenation of very similar shapes, ligatures and extensive use of dots.

The characteristics, which constrain hand writting may be combined in order to define application domains for which the results of automatic processing are satisfactory. The resulting commercial products have proved that handwriting processing can be integrated into working environments. Most efforts have been devoted to mail sorting, bank check reading, forms processing in administration and insurance. These applications are of great economic interest, each of them concerning millions of documents.

Mail sorting is a good example of the evolution in the domain, where the number of writers is unconstrained. In the early stages, only ZIP code was recognized. Then cities (and states as in the U.S.) were processed, implying the recognition of several types of handwriting: hand printed, cursive, or a mixture of the both. The use of the

redundancy between the ZIP code and the city name, as well as redundancy between numeral and literal amounts in bank checks, shows that combining several sources of information improves the recognition rates. Today, the goal is to read the full address, down to the level of the information used by the individual carrier. This necessitates precisely extracting the writing lines, manipulating a very large vocabulary and using contextual knowledge as the syntax of addresses (such as in the case of reading the literal amount of checks, the use of syntactic rules improves the recognition).

In this thesis, we focus on two major character recognition problems and generate a complete scheme for each of them:

- *Writer independent isolated character recognition :* A Hidden Markov Model (HMM) based recognition scheme is developed and tested on several databases including the ones from National Institute of Standards and Technology (NIST). The results are better than the previously reported ones.

- *Writer independent cursive script recognition :* The above HMM based recognition scheme is embedded into a robust segmentation algorithm which is applied before and after HMM recognition. The last step of segmentation confirms the initial segmentation using the recognition results. This algorithm is tested on locally generated and a public domain datasets obtained from Internet.

The major contributions of this study can be summarized as follows:

- A new set of one-dimensional discrete, constant length features is introduced to represent two-dimensional shape information for HMM (Hidden Markov Model) based recognition. The proposed feature set embeds the two dimensional information into a sequence of one-dimensional codes, selected from a code book. It provides a consistent normalization among distinct classes of shapes, which is very convenient for HMM based shape recognition schemes. The normalization process consists of some parameters which are dynamically estimated in the training stage of HMM. Chapter 3 describes the isolated character recognition system with the proposed normalization and feature extraction methods.

- Although the theory of HMM is well established, the implementation involves a large number of assumptions and restrictive conditions. Therefore, the recognition results are very much dependent on the implementation issues. A number of

properties are investigated about HMM based handwriting recognition, including the effects of different methods of initial parameter estimation, the relationship between the number of trainable parameters and the size of the training set, the effects of varying the model size and topology. These implementation issues are explained in Chapter 4.

- Segmentation of the cursive word into characters is the most crucial part of the character based cursive word recognition systems. The proposed cursive word recognition system uses a sequence of segmentation and recognition algorithms. First, a new segmentation method performs the segmentation of the whole word into strokes which corresponds mostly to a character or a portion of a character. In the proposed method, the initial segmentation regions are determined in the boundary-extracted image. In each segmentation region, a multistage di-graph is constructed by using both the gray scale and binary image. A search algorithm finds the shortest path from top to bottom, which corresponds to the imperfect character boundaries. Then, each segment is recognized by HMM with an output of classification probability. Finally, in order to confirm the preliminary segmentation and HMM recognition results, a recognition based segmentation method is presented. In Chapter 5, a complete scheme for word-level handwritten cursive script recognition is presented.

- The performance evaluation of both the isolated character recognition and the word-level recognition systems on various national and international databases is carried out. First of all, in order to test the isolated character recognition scheme, National Institute of Standards and Technology (NIST) Special Database 19 is used. This set also contains the data of Special Database 3 and 7, which consist of isolated digits and alphabetic characters. Total of 402 953 digits and 140 393 lowercase letter of this data set are used to test our isolated character recognition scheme. The second database used for this scheme is the locally generated handwritten digits. The digits are extracted from the connected digits collected from 19 persons. It contains 8000 number of digits. The cursive handwriting recognition system is tested on the dataset generated in [152] which contains 300 handwritten words. In addition, locally genereted 2000 connected digits are used for testing this system.

CHAPTER 2

AN OVERVIEW FOR MACHINE SIMULATION OF HUMAN READING

In this chapter, an overview of Automatic Character Recognition (ACR) techniques is presented. The available techniques are classified and compared. The pros and cons of each technique are discussed and directions for future research are suggested. The material will serve as a guide and update for the readers, who are working in the ACR area. Special attention is given to the OCR of handwritten characters, since this area requires more research to fill in the gap to the ultimate goal of machine simulation of human reading.

2.1 Introduction

Machine simulation of human functions has been a very challenging research area since the advent of digital computers. In some areas which require certain amount of intelligence, such as number crunching or chess playing, tremendous improvements are achieved. On the other hand, humans still outperform even the most powerful computers in the relatively routine functions such as vision. Machine simulation of human reading is one of these areas, which was the subject of intensive research for the last three decades, yet it is still too far from the final frontier.

This overview serves as an update for those papers which are published once in a while in relevant journals that cover the recent developments in ACR area. The study gives a bird's eye view of the ACR systems for various applications that can be classified

based upon two major criteria: Firstly, the ACR systems can be classified according to the data acquisition process as, on-line or off-line character recognition. Secondly, the ACR systems can be classified according to the text type, as printed or handwitten character recognition. No matter which class the problem of ACR belongs, in general there are four major stages (with or without feedback) in an ACR problem (Figure 2.1):

1. Pre-processing,
2. Segmentation,
3. Feature extraction,
4. Training and Recognition.

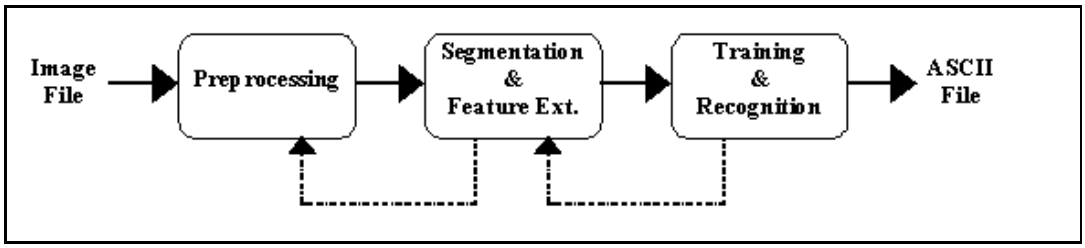


Figure 2.1: Stages of an ACR System

The chapter is arranged to review the ACR methodologies with respect to the stages of the ACR systems, rather than surveying the complete systems for the above mentioned classes. After giving a historical review of the ACR developments in Section 2.2, ACR systems are classified in Section 2.3. Then, the methodologies of ACR for pre-processing, segmentation, feature extraction, training and recognition are reviewed in section 2.4. Finally, the existing techniques are compared and the future research directions are discussed in Section 2.4.5.

2.2 History

Writing which has been the most natural mode of collecting, storing and transmitting information through the centuries, now serves not only for the communication among humans, but also for the communication of humans and machines.

Since the advent of digital computers, machine simulation of human reading has become the subject of intensive research. A large number of references can be found

at the survey studies published on this topic [9], [10], [11] . The reason for such an effort was not only because of its challenge on simulation of human reading, but also, because it provides efficient applications for the automatic processing of bulk amount of papers (i.e. transferring data into machines) such as bank cheques, commercial forms, government records, credit card imprints and postal code reading. Historically, ACR systems have evolved in three ages.

1900-1980 Early ages: The ACR studies were started by the Russian scientist Tyurin in 1900 [9]. The first modern character recognizers appeared in the middle of the 1940s with the development of the digital computers. The early work on the automatic recognition of characters has been concentrated either upon well printed text or upon small set of well distinguished hand-written text or symbols. Successful, but, constrained algorithms had been implemented mostly for Latin characters and numerals. Besides, some studies on Japanese, Chinese, Hebrew, Indian, Cyrillic, Greek and Arabic characters and numerals in both printed and handwritten cases are, also done [13], [14], [15].

The commercial character recognizers were available in 1950s when electronic tablets capturing the x-y coordinate data of pen-tip movement was first introduced. This innovation enabled the researchers to work on the on-line hand-written recognition. A good source of references until 1980 can be found in [12].

1980-1990 Developments: The studies until 1980s suffered from the lack of advanced algorithms, powerful hardware and optical devices. With the explosion on the computer hardware and software technology, the previously developed methodologies found a very fertile environment for rapid growth in many application areas as well as ACR system development [9], [10].

After 1990 Advancements: Nowadays, there is a renewed interest in the character recognition field, which involves the recognition of both printed and handwritten characters. It is not only we now have more compact and powerful computers and more accurate electronic equipment such as scanner, camera, electronic tablet etc., but, we have better recognition algorithms which utilize advanced methodologies such as Neural Networks, Hidden Markov Models or Fuzzy Set Reasoning. But, still there is a long way to go in order to reach the ultimate goal of machine simulation of fluent human reading.

In the following sections, we will describe and classify the stages of ACR systems and

the existing techniques in the literature, emphasizing their problems and superiorities.

2.3 ACR Systems

A literature review of ACR for the last fifteen years, based on journal articles, conference proceedings and patents show that there are hundreds of independent studies conveyed in the field, yielding dozens of commercially available products for various size and applications. All these systems can be examined in two categories :

1. Systems classified according to the data acquisition techniques.
2. Systems classified according to the text type.

Data is captured from its source to the computers by various data acquisition techniques. The ACR methodologies are very much dependent on the type of the equipment used for data acquisition. The text type is another determining factor for the ACR techniques. In the following sub-sections the classification of the ACR techniques will be given based upon the criteria mentioned above.

2.3.1 Systems Classified According to the Data Acquisition

The progress in automatic character recognition systems is evolved in two categories according to the mode of data acquisition:

- On-line character recognition systems,
- Off-line character recognition systems.

Off-line character recognition captures the data from paper through optical scanners or cameras whereas the *on-line* recognition systems utilize the digitizers which directly captures writing with the order of the strokes, speed, pen-up and pen-down information.

2.3.1.1 On-line Character Recognition Systems

The problem of recognizing handwriting recorded with a digitizer as a time sequence of pen coordinates is known as on-line character recognition. While the digitizer captures the data during writing, the ACR system with or without a lag makes the recognition. The digitizers are mostly electromagnetic-electrostatic tablets which send the coordinates of the pen tip to the host computer at regular intervals. Some digitizers use

pressure-sensitive tablets which have layers of conductive and resistive material with a mechanical spacing between the layers. There are also, other technologies including laser beams and optical sensing of a light pen.

While nothing opposes the idea of a computer that would use multiple input modalities including speech, keyword and pen, some applications call for a pen-only computer interface: in social environment, speech does not provide enough privacy, for small hand-held devices and for large alphabet, the keyboard is cumbersome. Applications are numerous: personal organizer, personal communicator, notebook, data acquisition device for order entries, inspections, inventories, surveys, etc.

The on-line handwriting recognition problem has a number of distinguishing features, which must be exploited to get more accurate results than the off-line recognition problem;

- It is adaptive. The immediate feed-back is given by the writer whose corrections can be used to further train the recognizer.

- It is a real time process. It captures the temporal or dynamic information of the writing. This information consists of the number of pen-strokes (i.e. the writing from pen down to pen up), the order of pen-strokes, the direction of the writing for each pen-stroke and the speed of the writing within each pen-stroke.

- Very little preprocessing is required. The operations such as smothing, deslanting, deskewing and feature extraction operations such as the detection of line orientations, corners, loops and cusps are easier and faster with the pen trajectory data than on pixel images.

- Segmentation is easy. Segmentation operations are facilitated by using the pen-lift information, particularly for handprinted characters.

- Ambiguity is minimal. The discrimination between optically ambiguous characters may be facilitated with the pen trajectory information.

- It captures human signatures and sketches.

On the other hand, the disadvantages of the on-line character recognition are as follows:

- The writer requires a special equipment which is not as comfortable and natural to use as pen and paper.

- It can not be applied to documents printed or written on papers.
- Punching is much faster and easier than handwriting for small size alphabet such as English or Arabic.

2.3.1.2 Off-line Character Recognition Systems

Off-line character recognition is also, known as "Optical Character Recognition" (OCR); because the image of writing is converted into a bit pattern by an optically digitizing device such as optical scanner or camera. the recognition is done on this bit pattern data for both printed or hand-written text. The research and development is well progressed for the OCR of the printed documents. Recently, the focus of attention is shifted towards the recognition of hand-written script.

The bit pattern data is shown by a matrix of pixels. This matrix can be very large. In order to meet the complexity and sophistication and to insert much data in recognition, most scanners are designed to have an x-y resolution of typically 100-1600 dots per inch.

The major advantage of the off-line recognizers is to allow the previously written and printed texts to be processed and recognized. Some applications of the off-line recognition are large scale data processing such as postal address reading, cheque sorting, short hand transcription, automatic inspection and identification, reading aid for visually handicapped.

The drawbacks of the off-line recognizers, compared to on-line recognizers are summarized as follows:

- Off-line conversion usually requires costly and imperfect pre-processing techniques prior to feature extraction and recognition stages.
- They do not carry temporal or dynamic information such as the number and order of pen-on and pen-off movements, the direction and speed of writing and in some cases, the pressure applied while writing a character.
- They are not real-time recognizers.

2.3.2 Systems Classified According to the Text Type

There are two main areas of interest in character recognition, namely:

1. Printed Character Recognition.

2. Hand-written Character Recognition.

The problem of printed character recognition is relatively well understood and solved with little constraints. When the documents are typed on a high quality paper with modern printing technologies, the available systems yield as good as 99 % recognition accuracy. On the other hand, hand-written character recognition systems have still limited capabilities even for recognition of the Latin characters.

2.3.2.1 Printed Character Recognition

The printed texts include all the printed materials such as books, newspapers, magazines, and documents which are the outputs of typewriters, printers or plotters. On the basis of the capabilities and complexities printed characters can be further classified as [16]:

- Fixed-font Character Recognition (recognition of a specific font),
- Multi-font Character Recognition (recognition of more than one font),
- Omni-font Character Recognition (recognition of any font).

The above classes of printed character recognition problems are well studied. However, the recognition rates of the commercially available products are very much dependent on the age of the documents, quality of paper and ink which may result in significant data acquisition noise.

2.3.2.2 Hand-written Character Recognition

Handwritten caharacter recognition, based on the form of written communication, can be divided into two categories: *cursive script* and *handprinted characters*. In practice, however, it is difficult to draw a clear distinction between them. A combination of these two forms can be seen frequently. Based on the nature of writing and the difficulty of segmentation process, Tappert [17] has defined five stages for the problem of handwritten word recognition:

- Stage 1: Boxed discrete characters,

- Stage 2: Spaced discrete characters,
- Stage 3: Run-on discretely written characters,
- Stage 4: Pure cursive script writing,
- Stage 5: Mixed cursive, discrete and run-on discrete.

Although, many powerful and efficient techniques are available for the recognition of printed documents, hand-written recognition is still an unsolved problem. It is, also, the most difficult part of the ACR area, because depending on the style of the writer and the speed of writing, some characters may vary in size or shape, in stroke number and order (dynamic variation). Human visual system is insensitive to the position, orientation and size changes of characters. However, representation of the knowledge of these variations affects the recognition rates a great deal.

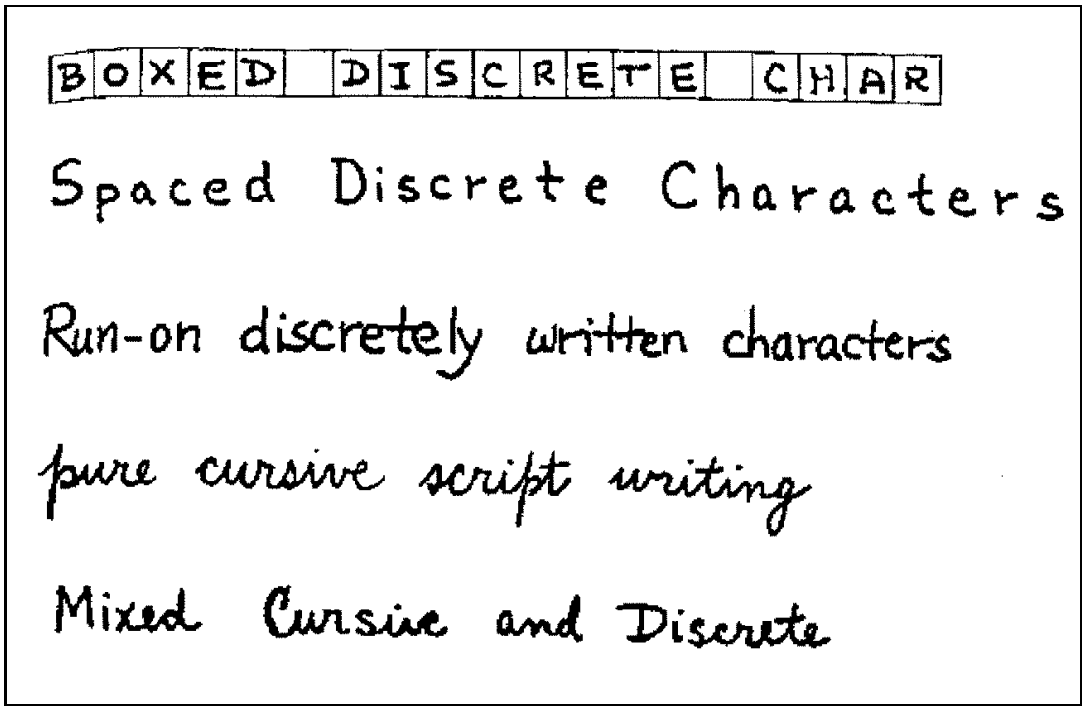


Figure 2.2: Different types of handwritten words

Figure 2.2 gives the example of each stage. The first item on this list requires the writer to place each character within its own box on a form. The boxes themselves can be easily found and dropped out of the image, or can be printed on the form in a special color ink that will not be picked up during scanning, thus eliminating the segmentation problem entirely. Spaced discrete characters can be segmented reliably

by means of horizontal projections, creating a histogram of grey values in the image over all the columns, and picking the valleys of this histogram as the points of segmentation. This has the same level of segmentation difficulty as is usually found with clean machine printed characters. Characters at the third stage are usually discretely written, however they may be touching, therefore making the points of segmentation less obvious. Degraded machine printed characters may also be found at this level of difficulty. There has been a fairly extensive research on the first three stages [18], [19], [20], [9].

However, cursively written and mixed written texts require more sophisticated approaches compared to the previous cases. First of all, advanced segmentation techniques is to be used for character based recognition schemes. In pure cursive handwriting, a word is formed mostly from a single stroke (line written from pen-down until pen-up). This makes segmentation by the traditional projection or connected-component methods ineffective. Secondly, shape discrimination between characters that look alike, such as U-V, I-1, O-0, is, also, difficult and requires the context information. In some languages, such as Arabic, there are characters which can only be differed from the others according to the number or position of dots. A good source of references in hand-written character recognition can be found in [22].

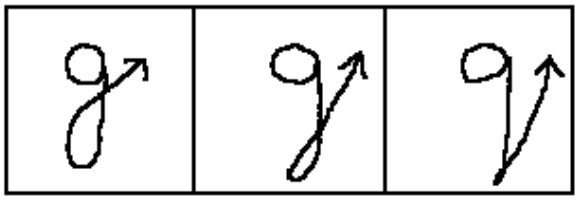
Handwriting is an art of drawing pictures for expressing the human thoughts in a condensed and systematic fashion. As a result there are infinite variations of hand-writing. Before developing an effective algorithm for hand-written cursive script, it is necessary to categorize the variations of hand-writing for individuals. A comprehensive study, where the variability effects in handwritten documents are given in two major categories, is presented in [23]:

1. *Perceptual Variability:* It is a relatively simple concept which looks for an answer to the question of "How humans read?". It deals with the subjectivity in perception of the same shape by different subjects or the same subject at different times.

2. *Generative Variability:* It deals with how characters are written. One important generative variability effect is various styles for connecting the strokes within a character. Figure 2.3 represents an example for generating three different sets of strokes for generating the letter "N".

Figure 2.3: Variants for "N" generated by connecting the strokes.

The straight strokes are always generated with some curvatures, yielding a problem in recognition. While some looped characters are generated with cusps, the cusps may be written as small loops yielding a lot of confusions (Figure: 2.4).



(a) (b) (c)

Figure 2.4: a) Open Loop, b) Partially Collapsed loop, c) Loop is converted into cusp

Different way of writing results the interchange of cusps and small loops in some characters. An example is given in Figure: 2.5.

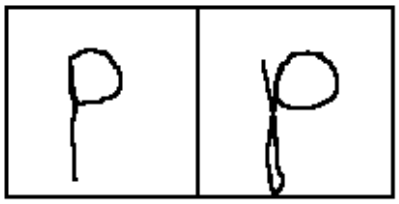


Figure 2.5: Interchange of physical loop and cusps in "p"

The above variability effects in handwriting makes the machine recognition of characters difficult. If a set of reasonable measures for variability can not be formalized, then making tests with any data will not show the performance of the ACR system in actual use.

2.4 Methodologies of ACR Systems

In this section, we focus on the methodologies of off-line character recognition. A hierarchical approach for most of the OCR systems would be from pixel to text, as follows:

$$Pixel \longrightarrow Feature \longrightarrow Character \longrightarrow Subword \longrightarrow Word \longrightarrow Meaningful\ \ text$$

This bottom up approach varies a great deal, depending upon the type of the OCR system and the methodology used. The literature review in the field of OCR indicates that the above hierarchical tasks are grouped in the stages of the OCR for pre-processing, segmentation, feature extraction, training and recognition. In some methods, some of the stages are merged or omitted, in others a feedback mechanism is used to update the output of each stage.

In this section, the available methodologies to develop the stages of the OCR will be presented.

2.4.1 Pre-processing

Pre-processing is done prior to the application of segmentation and feature extraction algorithms. The raw data is subjected to a number of preliminary processing steps to make it usable in the descriptive stages of character analysis. Pre-processing aims to produce clean document images that are easy for the OCR systems to operate accurately. The main objectives of pre-processing are:

- Noise reduction,
- Normalization of the data,
- Compression in the amount of information to be retained.

In order to achieve the above objectives, the following techniques are utilized in the pre-processing stage:

2.4.1.1 Noise Reduction

The noise, which is introduced by the optical scanning devices or the writing instrument, causes disconnected line segments, bumps and gaps in lines, filled loops etc. The

distortion which includes local variations, rounding of corners, dilation and erosion, is also a problem. Prior to the character recognition, it is necessary to eliminate these imperfections. There are many techniques to reduce the noise [24],[25],[26]:

1. *Filtering:* The aim of filtering is to remove noise and diminish spurious points, usually introduced by uneven writing surface and/or poor sampling rate of the image acquisition device. Various spatial and frequency domain filters can be designed for this purpose. The basic idea is to convolve a pre-defined mask with the image to assign a value to a pixel as a function of the gray values of its neighboring pixels. One example is to use linear spatial mask where the intensities $x(i,j)$ of the input image is transformed to the output image by

$$v(i,j) = \sum_k \sum_l a_{kl} \quad x(i-k, j-l) \tag{2.1}$$

where a_{kl} is the weight of the gray levels of pixels of the mask at location (k,l). Filters can be designed for smoothing, sharpening, theresholding and contrast adjustment purposes [27], [28].

2. *Morphological Operations:* A powerful tool for image enhancement is the Minkovsky morphological operations defined between the input image and a structuring element. Minkovsky operations replace convolution by logical operations. Two basic morphological operations are called dilation and erosion and are defined as follows:

$$Erosion : I \bigoplus A = \{x : (A_x) \cap I \neq \Phi\},$$

$$Dilation : I \Theta A = \{x : (A_x) \subseteq I\},$$

where I and A are the input image and the structuring element respectively and A_x is the translation of A located at the origin x. Various morphological operations can be defined and structural elements can be designed to successfully connect the broken strokes, smooth the contours, prune the wild points, thin the characters and extract the boundaries [28], [29].

3. *Noise modeling:* The noise could be removed by some calibration techniques if a model for it were available. However, modeling the noise is not possible in most of the applications. There is very little work on modeling the noise introduced by optical distortion, like speckle, skew, and blur [30], [31].

2.4.1.2 Normalization

Normalization methods aim to remove all types of variations during the writing and obtain standardized data. The followings are the basic methods for normalization [32], [33].

1. *Skew Normalization:* Due to inaccurasies in the scanning process and writing style, the writing may be slightly tilted within the image. This can hurt the effectiveness of later algorithms and therefore should be detected and corrected. Additionally, some characters can be distinguished regarding to the relative position with respect to the baseline (e.g. "9" and "g"). Methods of detecting the amount of skew include using the projection profile of the image [34], using the Hough transform [35], and a form of nearest-neighbor clustering [36]. After skew detection, the character or word is translated to the origin, rotated until the baseline is horizontal and re-translated back into the display screen space.

2. *Slant Normalization:* One of the measurable factors of different handwriting styles is the angle between longest stroke in a word and the vertical direction referred to the word slant. Slant normalization is used to normalize all characters to a standard form with no slant. In [10], slant detection is performed by dividing the images into different vertical windows through removal of horizontal lines with long runs. These windows are then further divided horizontally by removing window portions that contain no writing. The slant is estimated based on the center of gravity of the upper and lower half of each window averaged over all the windows. Another study uses an approach in which projection profiles are computed for a number of angles away from the vertical direction. The angle corresponding to the projection with the greatest positive derivative is used to detect the least amount of overlap between vertical strokes, and therefore the dominant slant angle. Lastly, in [37] a variant of Hough transform is used by scanning left to right across the image and calculating projections in the direction of 21 different slants. The top three projections for any slant are added and the slant with the largest count is taken as the slant value.

3. *Size normalization:* It is used to adjust the character size to a certain standard. Methods of character recognition may apply both horizontal and vertical size

normalizations. In [38], the character is divided into number of zones and each of these zones is seperately scaled. On the other hand, word recognition, due to the desire to preserve large intra-class differences in the length of words so they may assist in recognition, tends to only involve vertical height normalization, or bases the horizontal size normalization on the scale factor calculated for the vertical normalization [37].

4. *Curve Smoothing:* It eliminates the errors due to the erratic hand motion during the writing. It, generally, reduces the number of sample points needed to represent the script, thus improves efficiency in remaining pre-processing steps [149].

2.4.1.3 Compression

In addition to the classical lossless image compression techniques (like JPEG), the characters can be further compressed by thresholding and thinning algorithms.

1. *Thresholding:* In order to reduce storage requirements and to increase processing speed, it is often desirable to represent grey scale or color images as binary images by picking some threhold vlaue for everything above that value is set to 1 and everything below is set to 0. Two categories of thresholding exist: *Global* and *Adaptive*. Global threholding picks one threshold value for the entire document image, often based on an estimation of the backgroung level from the intensity histogram of the image. Adaptive threholding is a method used for images in which different regions of the image may require different threshold values. In [39], a comparison of many common thresholding techniques is given by using an evaluation criteria that is goal-directed in the sense that the accuracies of a character recognition system using different techniques were compared. On those tested, Niblack's method [40] produced the best result.

2. *Thinning:* This process extracts the shape information of characters. However, during the thinning process, a compression in data can be obtained. In ACR problems, most of the information can be extracted from the shape of the strokes. Therefore, the skeleton of a given character or stroke is necessary and in most cases sufficient for recognition. There are two basic approaches for thinning: *Pixelwise thinning* and *Non-pixelwise thinning*. In the pixelwise thinning, the

image is locally and iteratively processed until one pixel wide skeleton is remained. These methods includes, erosion and iterative contour pealing. They are very sensitive to noise and may deform the shape of the character. On the other hand, the non-pixelwise methods use some global information about the character during the thinning. Clustering based thinning methods, use the cluster centers as the skeleton [41]. Some thinning algorithms identify the singular points of the characters, such as end points, cross points and loops. These points are the source of problems. In a non-pixelwise thinning, they are handled with global approaches [42],[43]. In most of the recognition systems, thinning is applied before the segmentation and feature extraction stage [41], [42]. A survey of thinning methods is available in [44].

All of the above techniques are well explored and applied in many areas of image processing besides ACR. Therefore, recent text books in Digital Image Processing such as [24],[25],[26] can serve as good sources of available techniques and references.

Note that, the above techniques affect the data and may introduce unexpected distortions to the document image. As a result, these techniques may cause to the loss of important information about the writing. They should be applied with care.

2.4.2 Segmentation

The preprocessing stage yields a "clean" document in the sense that maximal shape information with maximal compression and minimal noise on normalized image is obtained. The next stage is segmenting the document into its sub components and extracting the relevant features to feed to the training and recognition stages. Segmentation is an important stage, because the extent one can reach in separation of words, lines or characters directly affects the recognition rate of the script. There are two types of segmentation : *External Segmentation*, which is the isolation of various writing units, such as paragraphs, sentences or words, prior to the recognition. *Internal Segmentation*, which is the isolation of letters, especially, in cursively written words.

External segmentation decomposes the page layout into its logical parts. It provides savings of computation for document analysis. Page layout analysis is accomplished in two stages [53]. The first stage, is the *structural analysis*, which is concerned with the segmentation of the image into blocks of document components (paragraph, rows,

words, etc). The second, is the *functional analysis*, which uses location, size and various layout rules to label the functional content of document components (title, abstarct, etc). Structural analysis may be performed in top-down, bottom-up or in some combination of the two approaches.

A number of approaches regards a homogeneous region in a document image as a textured region. Page segmentation is then implemented by finding textured regions in gray-scale images. As a representative approach of Jain et al.'s Gabor filtering and mask convolution [45] is used, Tang et al.'s approach is based on fractal signature [46] and Doermann's method [47] employs wavelet multiscale analysis. Many approaches to page segmentation concentrate on processing background pixels or using the white space in a page to identify homogeneous regions. These techniques include X-Y tree [48], pixel based projection profile [49], and connected component based projection profile [50], white space tracing [51], and white space thinning [52]. They can be regarded as top-down approaches, which segment a page recursively by X-cut and Y-cut from large components, starting with the whole page to small components, eventually reaching individual characters. On the other hand, there are some bottom-up methods which recursively grow the homogeneous regions from small components based on the processing on pixels and connected components. An example of this approach may be docstrum method which uses k-nearest neighbor clustering [53]. Some techniques combine both top-down and bottom-up techniques [54]. A brief survey of the work in page decomposition can be found in [55].

Internal Segmentation is an operation that seeks to decompose an image of a sequence of characters into subimages of individual symbols. Explicit segmentation of cursive script into letters is still an unsolved problem. Methods for treating the problem have developed remarkably in the last decade and a variety of techniques has emerged. In [56], character segmentation strategies are divided into four categories;

- *Segmentation by dissection method* identfies the segments based on character like properties. This process of cutting up the image into meaningful components is given a special name, "dissection". Dissection is an intelligent process that analyzes an image without using specific class of shape information. The criterion for good segmentation is the agreement of general properties of the segments with those expected for valid characters. Available methods based on the dissection

of an image use white Space and Pitch [57], vertical projection analysis [58], connected component analysis [59], landmarks [60]. Moreover, segmentation by dissection can be subjected to evalution using linguistic context [61].

- *Recognition Based Segmentation* searches the image for components that match predefined classes. Segmentation is performed by use of recognition confidence, including syntactic or semantic correctness of the overall result. In this approach, two different methods can be employed; methods that make some search process and methods that segment a feature representation of the image.

  The first one attempts to segment words into letters or other units without use of feature based dissection algorithms. Rather, the image is divided systematically into many overlapping pieces without regard to content. Conceptually, these methods originate from schemes in [62], [63] for the recognition of machine printed words. The basic principle is to use a mobile window of variable width to provide sequences of tentative segmentations, which are confirmed by character recognition. Another technique combines dynamic programming and neural networks [64]. Lastly, the method of selective attention takes neural networks even further in the handling of segmentation problems [66]

  On the other hand, the second method segments the image implicitly by classification of subsets of spatial features collected from the image as a whole. This method can be divided into two categories: Hidden Markov Model Based approaches and Non-Markov based approaches. The object of Hidden Markov Models is to model variations in printing or cursive writing as an underlying probabilistic structure, which is not directly observable. The survey in [67] provides an introduction to this approach in recognition applications. Non-Markov approaches stems from concepts used in machine vision for recognition of occluded objects [68]. This family of recognition based approaches use probabilistic relaxation [69], the concept of regularities and singularities [70] and backward matching [71].

- *Mixed Strategies (Oversegmenting)* combines dissection and search methods in a hybrid way. A dissection algorithm is applied to the image, but the intent is to "oversegment", i.e., to cut the image in sufficiently many places that the

correct segmentation boundaries are included among the cuts made. Once this is assured, the optimal segmentation is sought by evaluation of subsets of the cuts made. Each subset implies a segmentation hypothesis, and classification is brought to bear to evaluate the different hypothesies and choose the most promising segmentation [72], [73].

- *Holistic Strategies* attempts to recognize words as a whole, thus avoiding the need to segment into characters. Recognition is based on the comparison of a collection of simple features extracted from the whole word against a lexicon of codes representing the theoretical shape of the possible words. Words are represented by a list of features such as ascenders, descenders, directional strokes, closed loops, cusps, etc. Recent techniques used in comparison between hypotheses and references are flexible and take into account the dramatic variability of handwriting. These techniques generally rely on dynamic programming to satisfy optimization criteria based either on distance measurements [74] or else on a probabilistic framework using Markov [75] or Hidden Markov Chains[76]. Use of holistic methods is ordinarily constrained to a specific lexicon. However, more general applications require a dynamic generation of holistic description [77].

It is very difficult to compare the effectiveness of the above techniques. They must be considered together with the systems for which they were developed. However, as Casey and Lecolinet [56] pointed out, the wise use of context and classifier confidence has led to improved accuracies.

2.4.3 Feature Extraction

During or after the segmentation procedure the feature set, which is used in the training and recognition stage, is extracted. Feature sets play one of the most important roles in a recognition system. A good feature set should represent characteristic of a class that helps distinguish it from other classes, while remaining invariant to characteristic differences within the class. Hundreds of features which are available in the literature can be categorized as follows:

2.4.3.1 Global Transformation and Series Expansion Features

These features are invariant to global deformations like translation and rotations. A continuous signal generally contains more information than needs to be represented for the purposes of classification. This may be true of discrete approximations of continuous signals as well. One way to represent a signal is by a linear combination of a series of simpler well-defined functions. The coefficients of the linear combination provide a compact encoding known as *series expansion.* Common transform and series expansion features are:

1. *Fourier Transforms:* The general procedure is to choose magnitude spectrum of the measurement vector as the features in an n-dimensional Euclidean space. One of the most attractive properties of the Fourier Transform is the ability to recognize the position-shifted characters, when it observes the magnitude spectrum and ignores the phase. Fourier Transforms has been applied to OCR in many ways [82] [83].

2. *Walsh-Hadamard Transform:* This feature is more suitable in high-speed processing since the arithmetic computation involves only addition and subtraction. The major drawback of Walsh-Hadamard transform is that its performance depends heavily upon the position of the characters.

3. *Rapid transform:* It is same as the Hadamard Transform except for the absolute value operation, which may be credited with the elimination of the position-shifting problem.

4. *Hough Transform:* It is a technique for baseline detection in documents. It is also applied to characterize parameter curves of characters.

5. *Gabor Transform:* The Gabor transform is a variation of the windowed Fourier Transform. In this case, the window used is not a discrete size, but is defined by a Gaussian function [84].

6. *Wavelets:* Wavelet transformation is a series expansion technique that allows us to represent the signal at different levels of resolution. In OCR area, it is our advantage to handle each resolution seperately [85].

7. *Karhunen-Loeve Expansion:* It is an eigen vector analysis which attempts to reduce the dimension of the feature set by creating new features that are linear combinations of the original features [86].

8. *Moments:* Moment normalization strives to make the process of recognizing an object in an image size, translation, and rotation independent [81].

2.4.3.2 Statistical Features

These features are derived from the statistical distribution of points. They provide high speed and low complexity and take care of style variations to some extent [12]. They may, also, be used for reducing the dimension of the feature set. The followings are the major statistical features:

1. *Zoning:* The frame containing the character is divided into several overlapping or non-overlapping zones and the densities of the points and some strokes in different regions are analyzed and form the features. Contour direction features measure the direction of the contours of the characters [87]. This can be done by applying a number of 2 x 2 masks over the image, one set of masks for each of the four countour directions. Another example is bending point features. Bending points are points at which a stroke in the image has a strong curvature. These can be found by tracing the contour of strokes in the image and matching local parts of the contour against prespecified templatesi [88].

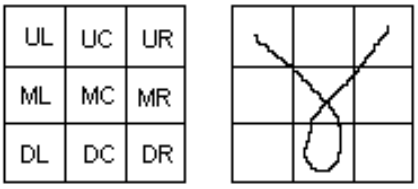


Figure 2.6: Zoning and 3x3 non uniform matrix representation (U: Up, D: Down, M: Middle, C: Center, R: Right, L: Left).

2. *Characteristic Loci:* For every white point in the background of the character, vertical and horizontal vectors are generated. The number of times that the line

segments intersected by these vectors are used as features.

3. *Crossing and Distances:* Features are measured from the number of times line segments are traversed by vectors in specified directions, or the distances of elements or line segments from a given boundary such as the frame which contains the character.

2.4.3.3 Geometrical and Topological Features

These features may represent global and local properties of characters and have high tolerances to distortions and style variations. They also tolerate a certain degree of translation and rotation. These topological features may encode some knowledge about the contour of the object or may require some knowledge as to what sort of components make up that object.

1. *Strokes:* These are the primitive elements, which make up a character. The strokes can be as simple as lines (l) and arcs (c) which are the main strokes of Latin characters (Figure 2.7) and can be as complex as curves and splines making up Arabic characters. In on-line character recognition, a stroke is also defined as a line segment from pen-down to pen-up [11].

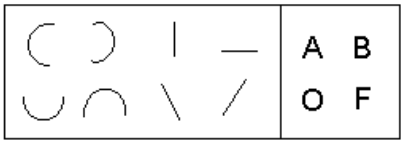


Figure 2.7: Main strokes used for Latin characters

2. *Stroke Directions and Bays:* The sequence of directions of pen motion during the writing of a character is used as features.

3. *Chain-Codes:* This feature is essentially obtained by mapping the strokes of a character into a 2-dimensional parameter space which is made up of codes as shown in Figure 2.8.

4. *End points, intersections of line segments and loops.*

5. *Strokes relations and angular properties.*

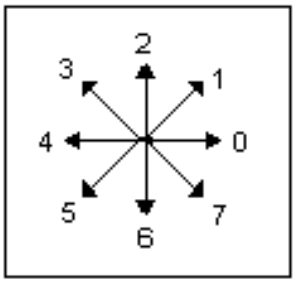


Figure 2.8: Chain Codes

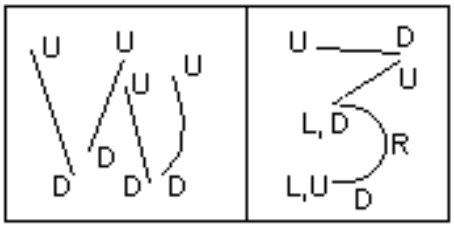


Figure 2.9: Coding of extreme (U: Up, D: Down, L: Left, R: Right)

6. *Extremum points* (Figure: 2.9)

   Some features reported by Brown [89] for the recognition of Latin scripts are:

7. *Number of central threshold crossings.*

8. *Number of upper or lower crossings.*

   The above two features imply that, a horizontal threshold is established above, below and through the center of the normalized script. The number of times that the script crosses a threshold becomes the value of that feature.

   The obvious intent is that upper portions in the script (such as for the letters b, d, f, h, k and l) are to be detected by the upper threshold and similarly for the lower portions (f, g, j, p, q, y) by the lower threshold.

9. *Number of maxima and/or minima.*

10. *Number of cusps above and/or below center threshold.*

11. *Number of openings right, left, up and/or down.*

12. *Number of cross (x) points, branch (T) points, line ends (J) and/or loops (O)* [139].

    Some other features reported by El-Sheikh [14] for the recognition of Arabic characters are the followings:

13. *The ratio between width and height or its inverse.*

14. *The number and sequence of minima and/or maxima in the x and/or y directions of the main stroke.*

15. *The relative distance between the last point and the last y-min.*

16. *The relative horizontal and vertical distances between first and last points.*

    Some features reported by Kundu [139], [91] as geometrical and topological features are as follows:

17. *Zero crossings:* Through the center of gravity (calculated with the moment features) of thinned character, a horizontal line is drawn. The number of intersections of this line with the letter gives the number of zero crossings.

18. *Approximate semi-circles in East-West-North and South directions* (e.g.: "c" has a semi circle in the west whereas, b has q semi circle in the east etc.).

19. *Isolated dots.*

20. *A bend between two points.*

21. *Horizontal curves at top or bottom.*

22. *Distance between two points.*

23. *Direction of a stroke from a special point.*

24. *Inflection between two points.*

25. *Comparative lengths between two strokes.*

26. *Location of special points in frame.*

27. *Profile on the left or right.*

28. *Straight stroke between two points.*

29. *Width of a stroke.*

30. *Symmetry of character.*

31. *Spline approximation of strokes.*

The chief design function is to select the best set of features, which maximizes the recognition rate with the least amount of elements. This problem can be formulated as a dynamic programming problem for selecting the k-best features out of N features, with respect to a cost function such as Fishers Discrimant ratio. Selection of features using a methodology as mentioned here requires expensive computational power and most of the time yields a sub optimal solution. Therefore, the feature selection is mostly done by heuristic or intuition for a specific type of the ACR application.

2.4.4 Training and Recognition Techniques

As in many areas of Image Analysis, ACR systems extensively use the methodologies of pattern recognition, which assigns an unknown sample into a pre-defined class [92]. There are three main ACR techniques:

- Statistical ACR Techniques.
- Syntactic ACR Techniques.
- Neural Networks.

The above techniques use either holistic or analytic strategies for the training and recognition stages: Holistic strategy employs top-down approaches for recognizing the full word, eliminating the segmentation problem. The price for this computational saving is to constrain the problem of ACR to limited vocabulary. Also, due to the complexity introduced by the whole cursive word (compared to the complexity of a single character or stroke), the recognition accuracy is decreased.

On the other hand the analytic strategies employ bottom up approaches starting from stroke or character level and going towards producing a meaningful text. Explicit or implicit segmentation algorithms are required for this strategy, not only adding extra complexity to the problem, but also introducing segmentation error to the system. However, with the cooperation of segmentation stage, the problem is reduced to the

Table 2.1: Strategies:Holistic vs. Analytic

Holistic Strategy	**Analytic Strategy**
Whole word recognition	Subword or letter recognition
Limited vocabulary	Unlimited vocabulary
No segmentation	Requires explicit or implicit segmentation

recognition of simple isolated characters or strokes, which can be handled for unlimited vocabulary with high recognition rates.

2.4.4.1 Statistical Techniques

Statistical decision theory is concerned with statistical decision functions and a set of optimality criteria [140].

$$\max_{1 \leq k \leq n} Prob(O/M_k) \tag{2.2}$$

where O is the unknown observation vector and M_k is the model for class $k \in \{1, 2, ..., c\}$. Statistical techniques are mostly based on three major assumptions:

- Distribution of the feature set is Gaussian or in the worst case uniform.
- There is sufficient statistics available for each class
- Given Ensemble of images $\{I\}$, one is able to extract a set of features $\{f_i\} \in F$, $i \in \{1, ..., n\}$ which represents each distinct class of patterns.

The measurements taken from N-features of each word unit can be thought to represent an N-dimensional vector space and the vector, whose coordinates correspond to the measurements taken, represents the original word unit. The major statistical approaches, which are applied in the character recognition field are the followings:

- *Non-parametric Recognition :* This method is used to seperate different pattern classes along hyper planes defined in a given hyper space. The best known method of non-parametric classification is the Nearest Neighbor (NN) and is extensively used in character recognition [141]. It does not require a priori information about the data. An incoming pattern is classified using the cluster, whose center is the minimum distance from the pattern over all the clusters.

- *Parametric Recognition:* When a priori information is available about the caharacters, it is possible to obtain a parametric model for each character [142]. Therefore, once the parameters of the model, which may be based on some probabilities, are obtained, the characters are classified according to some decision rules such as maximum Likelihood or Bayes' method.

- *Clustering Analysis:* The clusters of character features, which represent distinct classes, are analyzed by way of clustering methods. Clustering can be performed either by an agglomerative or a divisive algorithm. The agglomerative algorithms operate step-by-step merging of small clusters into larger ones by a distance criterian. On the other hand, the divisive methods split the character under certain rules for identifying the underlying character [143].

- *Hidden Markov Modeling (HMM):* Hidden Markov Model is defined as a stochastic process generated by two mechanisms; a Markov Chain having a finite number of states and a set of random functions, each of which is associated with a state [144]. Here, the job is to build a model that explains and characterizes the occurrence of the observed symbols [145], [93]. HMM is widely used in speech recognition area. It models the speech generation so that the system is capable of being in only a finite number of states. Each state is capable of generating a finite number of possible outputs. In generating a word, the system passes from one state to another, each state emitting an output according to some probabilities, which are also the parameters of the model until the entire word is out. Its use in character recognition is now becoming popular. There are two basic approaches to ACR systems using HMM:

    1. Model Discriminant HMM: A model is constructed for each class (word, letter or segmentation unit), in the training phase. States represent cluster centers for the feature space. Recognition is mostly done by VITERBI algorithm [93] based on dynamic programming. In ref [29] Kundu et al. used model discriminant HMM for numeral recognition. In ref [139] and [91] various studies are summarized for limitted vocabulary handwriting recognition, numeral recognition and signiture verification by using model discriminant HMM.

2. Path Discriminant HMM: Given an HMM, the goal is to find the best path for each class. In this approach modelling the contex or language is supported by transitional probabilities. In recognition of a language such as English, the letter sequences of the words can be modeled by Markov chains where the letters are identified as states. The initial and the transition probabilities are then calculated on the basis of observations from a random experiment, generally, by using a dictionary [106].

- *Fuzzy Set Reasoning:* Instead of using a probabilistic approach, this technique employs fuzzy set elements in describing the similarities between the features of the characters. Fuzzy set elements give more realistic results, when there is no a priori knowledge about the data and therefore the probabilities cannot be calculated. The characters can be viewed as a collection of line segments, (i.e. strokes) which can be used in similarity measures. The similarity between two characters can also be measured. Since the strokes under consideration are fuzzy in nature, the concept of fuzzy set is utilized in the similarity measure. In order to recognize a character, an unknown input character is matched with all the reference characters and is assigned to the class of the reference character with the highest score of similarity among all the reference characters [146].

  In [146], Cheng, Hsu and Chen utilize a fuzzy similarity measure to define Fuzzy Entropy for off-line handwritten Chinese characters. In [147], an off-line Handwritten character recognition system is proposed by Abuhaba and Ahmed using fuzzy graph theoretic approach, where each character is described by a fuzzy graph. A fuzzy graph matching algorithm is, then, used for recognition. In [148], Wang and Mendel propose an off-line handwritten recognition algorithm system which generates crisp features first, and then, fuzzify the characters by rotating or deforming them. The algorithm uses average values of membership for final decision.

2.4.4.2 Syntactic / Structural Techniques

The recursive description of a complex pattern in terms of simpler patterns based on the physical "shaping" of the object was the initial idea behind the creation of syntactic pattern recognition. These patterns are used to describe and classify the characters in

ACR systems. The characters are represented as the union of the structural primitives. It is assumed that the character primitives extracted from a writing are quantifiable and one can find the relations among them. The following Syntactic Pattern Recognition methods are applied to the ACR problems:

- *Grammatical Methods:* In mid 1960's, researchers started to consider the rules of linguistics for analyzing the speech and writing. Later, various orthographic, lexicographic and linguistic rules were utilized to build up recognition schemes. As a result the theory of formal grammars was developed.

  The grammatical methods create some production rules in order to form the characters from a set of primitives through formal grammars. These methods utilize any type of structural feature under some syntactic and/or semantic rules, which may yield a combined approach [107], [108], [109], [110]. Formal tools, like language theory, permit us both to describe the admissible constructions and to extract the contextual information about the writing, utilizing various types of grammars, such as string grammars, graph grammars, stochastic grammars and picture description language (PDL) [108], [111].

  In grammatical methods, training is accomplished by describing each character by a grammar $G_i = \{V_t, V_n, P, S\}$ for each class $i = 1, ..., n$ where V_t is a set of terminal symbols, V_n is a set of non-terminal symbols, P is a set of relations among the terminal and nonterminal symbols, and S is a starting symbol. In the recognition phase, the string, tree or graph of any writing unit (characters, words, sentences) is analyzed in order to decide to which pattern language (or grammar) it belongs [112]. Syntax Analysis is done by parsing. If a sentence is given, then a derivation of the sentence is constructed and the corresponding derivation tree is obtained. Various approaches can be [113] top-down or bottom-up parsing [114] and tree automata. Application of the grammatical methods in the ACR area can be character level, word level and context level [115]. In character level, Picture Description Language (PDL) is used to model each character in terms of a set of strokes and their relationship. This approach is used for Indian Character Recognition, where Devanagari characters are presented by a PDL [117]. The system stores the structural description in terms of primitives and the relations. Recognition involves a search for the unknown character, based on the stored

description and context. In word level, a dictionary of words with some linguistic rules is used during or after the recognition stage for verification and improvement purpose. Context information can be introduced to the ACR systems by natural language understanding methodologies. The output of the recognition stage can be further processed through parsing as a post processing stage of ACR systems to increase the recognition rates.

- *Graphical Methods:* Writing units are described by trees, graphs, di-graphs or attributed graphs. The character primitives (e.g. strokes) are selected by a structural approach, irrespective of how the final decision making is made in the recognition [118], [119]. For each class a graph or tree is formed in the training stage to represent strokes, letters or words. Recognition stage assigns the unknown graph to one of the classes by using a graph similarity measure.

  In the recent ACR literature, there are great variety of approaches that use the graphical methods. In [118], Suen et. al. propose a hierarchical attributed graph representation for Chinese characters. ln [120], Pavlidis and Rocha use Polygonal approximation of the character boundaries obtained from a split and merge algorithm, for numeral recognition. The features of their method are concave arcs, strokes, singular points and their relationships extracted from the polygons.

  In [121], an off-line Cursive Script Recognition scheme is proposed by Simon. The features are regularities, which are defined as uninformative parts and singularities, which are defined as informative strokes about the characters. Stroke trees are obtained after skeletonization. The goal is to match the trees of singularities.

  Although it is computationally expensive, relaxation matching is also a popular method in graphical approaches to ACR problem([122] and [123]). The hierarchical representation of strokes and word graphs are formed from the skeleton. All possible segmentation points or boundaries and n-gram statistics can be used in the recognition stage.

  Once a set of structural primitives are quantified in trees of a graph, edit distance can be used as a graph similarity measure [115]. Edit distance is computed by dynamic programming methods.

- *Matching:* ACR techniques vary widely according to the feature set formed by selecting the relevant features among the long list of features, described above. Based on the features, recognition is done by matching the feature vectors in order to decide to which pattern language (or grammar) they belong. Matching techniques can be further classified as follows:

  Template Matching and Correlation Techniques: An input character is directly compared to a standard set of stored prototypes. According to a similarity measure (e.g., Euclidean , Mahalanobis, Jaccard or Yule similarity measures etc.), a prototype matching is done for recognition. The comparison methods can be as simple as one-to-one comparison, or as complex as decision tree analysis in which only selected pixels are tested. A template matcher can combine multiple information sources, including match strength and k-nearest neighbor measurements from different metrics [124], [125]. Although template matching method is intuitive and has a solid mathematical backround, the recognition rate of this method is very sensitive to noise.

  Feature Analysis and Matching Techniques: These techniques are based on matching on feature spaces, which are distributed on an n-dimensional plane. A set of features that represents a characteristic portion of a letter, or a group of letters is compared to the feature vector of the ideal character. The description that matches most "closely" according to a distance measure, provides recognition [12].

  Flexible Matching Techniques: Some flexible matching techniques are proposed as a better alternative to template or feature analysis matching techniques.

  1) Curve and Elastic Matching: A character is recognized by matching a representative curve against those of prototype characters. It is a linear alignment of the points of the curve. However, due to nonlinearities, the best fit is usually an elastic matching. The basic idea of elastic matching is to optimally match the unknown symbol against all possible elastic stretching and compression of each prototype. Once the feature space is formed, the unknown vector is matched using dynamic programming and a warping function. The dynamic time warping technique was applied to speech recognition problems long time ago and has since became widespread in ACR field [126], [21]. Nowadays, it is quite popular

in on-line recognition systems.

2) Relaxation Matching: It is a symbolic level image matching technique which uses feature based description for the document image. First, the matching regions are identified. Then, based on some well- defined ratings of the assignments, the image elements are compared to the model. This procedure requires a search technique in a multi-dimensional space, for finding the global maximum of some functions [127], [122].

The matching methods mentioned above are sometimes used individually or combined in many ways as part of the ACR schemes.

In [128], Huang et. al. proposed a Multi font Chinese character recognition system where sampling points, including cross, branch and end poits on the skeleton are taken as nodes of a graph. Each character class is represented by a constrained graph model which captures the geometrical and topological invariance for the same class. There are two types of constraints: Position constraints are based on the distance between two points. Connection constraints are taken as arcs between nodes if there is a direct connection between them, Recognition is made by a relaxation matching algorithm.

In [122], Xie et. al proposed a handwritten Chinese character system, where small number of critical structural features, such as end points, hooks, T shape, cross, corner are used. Recognition is done by computing the matching probabilities between two features by a relaxation method. A new distance measure based on these matching probabilities are defined.

2.4.4.3 Neural Network (NN)

A Neural Network is defined as a computing architecture that consists of massively parallel interconnection of simple "neural" processors. Because of its parallel nature, it can perform computations at a higher rate compared to the classical techniques. Because of its adaptive nature, it can adapt to changes in the data and learn the characteristics of input signal. It is used in pattern recognition by defining nonlinear regions in the feature space. A neural network contains many nodes. The output from one node is fed to another one in the network and the final decision depends on the complex interaction of all nodes. Neural Network finds intensive applications in ACR.

Multilayer Feedforward NN proposed by R. Rosenblall, in 1962 and elaborated by Minsky, Papert, Nilsson, Widro and Hoff, is applied in character recognition and classification by many authors [129], [130]. One example is the limited vocabulary NN classifier of Y. Pao and G. Park, where segmentation and recognition are done simultaneously. Features which are formed by dividing the image into upper, lower and main body regions are fed to the NN.

Handwritten Digit Recognition of Price, Knerr [115], cooperates two feedforward NN with identical structure and different feature vectors. The first feature space consists of edges whereas the second one consists of line strokes. Complementary characteristic of the two data representations is incorporated in the recognition stage.

Neocognitron of Fukushima et. al. [131] is a hierarchical network consisting of several layers of alternating neuron-like cells. S-Cells are for feature extracting and C-Cells allow for positional errors in the features. Last layer is the recognition layer. Some of the connections are variable and can be modified by learning. Each layer of S and C cells are called cell planes. The method is deformation invariant.

Feature Recognition Network proposed by Hussain and Kabuka [132], has two level detection scheme: The first level is for detection of sub pattern and the second level is for detection of patterns.

2.4.4.4 Combined ACR Techniques

The above review indicates that, there are many training and recognition methods available for the ACR systems. Now, the questions "Which method will be the best?" or "May these methods be combined in a meaningful way ?" appear. The answer to the first question is very hard and depends on the specific ACR application.

As an answer to the second question, various strategies are proposed [133]. Among the others, the voting [134], Bayesian [135] and Dempster-Shafer [133], [136] and Behaviour-Knowledge space approaches [137] are the most representative. Voting is a democracy-behavior approach based on "the opinion of the majority wins" [137]. It treats classifiers equally without considering their differences in performance. Each individual classifier represents one score that is either as a whole assigned to one class label or divided into several labels. The label, which receives more than half of the total scores, is taken as the final result. While voting methods are only based on the

label without considering the error of each classifier, Bayesian and Dempster-Shafer approaches take these errors into consideration.

The Bayesian approach uses the Bayesian formula to integrate classifiers' decisions; usually. it requires an independence assumption in order to tackle the computation of the joint probability [135]. The Dempster-Shafer (D-S) formula, which has been applied to deal with uncertainty management and incomplete reasoning, can aggregate committed, uncommitted and ignorant beliefs. It allows one to attribute belief to subsets, as well as to individual elements of the hypothesis set. Both Bayesian and D-S approaches make use of probabilty to describe the different qualities of classifiers' decision. However, in the Bayesian approach, the sum of $P(C)$ and $P(\neg C)$ is equal to 1; this is not necessarily true for the D-S approach, where $P(C)$ represents the probability that C is true.

Finally, the Behavior-Knowledge Space method has been developed in [137]. The above three approaches require the independence assumption which may not hold in real applications. To avoid this assumption, the information shoul be derived from a knowledge space which can concurrently record the decisions of all classifiers on each learned sample. The knowledge space records the behavior of all classifiers and is called "Behavior-Knowledeg Space". This method derives the final decisions from the Behavior Knowledge space.

Some examples of the combined approaches may be given as follows;

A hybrid approach for handwritten numeral recognition proposed by Wang et.al [150], combines the Neural Network with a relaxation matching method. First, they apply a Neural Network scheme, where they use histograms obtained from different directions (vertical, horizontal, left and right diagonal) on the bit map image, as features. If the recognition is succesfull, then, they stop. If the character is rejected by the Neural Network, then, they apply a relaxation matching scheme which compares the feature vector of unknown pattern to the ones obtained from a dictionary. For this case, Freeman's chain code is used to represent each character. Depending on the matching criteria, credits are assigned to each comparison. Final decision is made according to the credit.

Another example is given by Shridhar and Kimura, where two algorithms are combined for the recognition of unconstrained isolated handwritten numerals [151]. In the first algorithm, a statistical classifier is developed by optimizing a modified quadratic

discriminant function derived from the Bayes rule. The algorithm has two thresholds to control error-rejection rate. The feature set of this algorithm is formed by directional vector features obtained from a local histogram of the chain codes for each 4x4 rectangular zone. The second algorithm implements a structural classifier: A tree structure is used to express each number. Recognition is done in a two pass procedure with different thresholds. The feature set of the second algorithm is extracted from the left and right profiles of the external contours. Character width, ratio, location of extreme and discontinuities are also added to the feature space. The recognition is made either using parallel or sequential decision strategies. In a parallel decision strategy, a character is accepted if both passes accept the same result, Otherwise, the result of both algorithms are analysed for final decision. In a sequential decision strategy, if the first algorithms succesfull, then the character is accepted. Otherwise, the second algorithm is applied. If it is succesfull, then the character recognized by the second algorithm is accepted. Otherwise, the final decision is made based on the multiple membership table derived from the results of both algorithms.

In [133], Xu et.al, studied the methods of combining multiple classifiers and their application to handwritten recognition. They proposed a sequential combination of structural classification and relaxation matching algorithm for the recognition of handwritten zip codes. It is reported that the algorithm has very low error rate and high computational cost.

In [10] Srihari et.al, proposes a recognition scheme for off-line cursive script word recognition where they combine three algorithms; template matching, mixed statistical and structural classifier and structural classifier The results derived from three algorithms are combined in a logical way. Significant increase in the recognition rate is reported.

In another scheme, [10] Srihari et.al, propose a holistic handwritten word recognition method, where they first normalize the image to remove the writer dependent characteristics as much as possible. Secondly, they find regularities (smooth strokes) and singularities (loops, cusps, crosses, end points etc.) as features. Then, they globally analyze the features of the input word and reduce the lexicon, by identifying the similar features. Three recognition method generates independent rankings of the lexicon: binary template matching, Bayesian classification of structural features, nearest neighbor matching of the moment based features. Finally, they generate a single consensus using

the rankings.

In [115] Camillerapp et.al, propose an off-line methods for recognition of bank checks, where they represent each word by a graph. Nodes represent singularities (end points, cross points etc.). A tree of primitives is derived from each graph. Tree comparison is based on dynamic programming. A filter module avoids useless tree comparisons. They maximize a likelihood function which is defined as the difference between the similarities and dissimilarities for final decision.

In [116], Shomaker proposes an on-line cursive script recognition scheme, where he uses Kohonen Network for symbolic classification. He feeds two sets of features to the same network; vector quantized strokes and characters. He reports that character based recognition is better than stroke based one. He combines both results to increase the recognition rate.

2.4.5 Discussion

In this chapter, we have overviewed the main approaches used in the ACR field. Our attempt was to bring out the present status of ACR research. Although each of the methods summarized above have their own superiorities and drawbacks, the presented recognition results of different methods seem very successful. Most of the recognition accuracy rates reported are over 85 %. However, it is very difficult to make a judgment about the success of the results of recognition methods, especially in terms of recognition rates, because of different databases, constraints and sample spaces. In spite of all the intensive research effort, none of the proposed methods solve the ACR problem for free style handwriting. A number of potential weaknesses which exist in the proposed methods can be summarized as follows:

- While holistic methods eliminate complicated and imperfect segmentation, they bring restriction to the vocabulary. Also, trying to recognize a whole word requires very complicated recognition scheme which nevertheles can not reach the rates of recognizing isolated characters.

- In all of the proposed methods, recognition is isolated from training . Therefore, it is not easy to improve the recognition rate using knowledge obtained from the analysis of the recognition errors.

- The reference pattern classes or the feature sets are pre-defined and dependent on the particular training set. If the training set is not large enough, the method will not work well for unknown samples.

- Many assumptions and parameters of the algorithms are set by trial and error at the initial phase. They are good for some specific training and recognition data, but are no longer valid even for small changes.

We should also evaluate the methods according to the information gathered by them:

- The structural information about the interconnections in complex patterns cannot be handled very well by statistical methods. Therefore, one needs to combine statistical and structural information in a meaningful way. On the other hand, the use of formal language-theoretic models to represent characters is the main drawback of the syntactic approach, because the characters are natural entities, which can not strictly obey the mathematical constraint set of the formal language theory. Imposing a strict rule on the pattern structure is not particularly applicable to character recognition, where intra-class variations are almost infinite. Furthermore, the linguistic approaches give little concern on the limitations of the feature extractor.

- Neural Networks are very succesful in combining statistical and structural information for many pattern recognition problems. They are comparatively immune to distortions. But, it has a discrete nature in the matching process, which may cause drastic mismatching. In other words, it is flexible, but, it is not continuous.

- Template matching methods deal with a character as a whole in the sense that an input plane is matched against a template, constrained on X-Y plane. This makes the procedure very simple and the complexity of character shape is irrelevant. However, it suffers from the sensitivity to noise and is not adaptive to variations in writing style. The capabilities of human reasoning are better captured by feature analysis techniques than by template matching. Some features are tolerant to distortion and take care of style variations, rotation and translation to some extent.

For a real life ACR problem, we need techniques to describe a large number of similar structures of the same category while allowing distinct descriptions among categorically different patterns. It seems that a hybrid model is the only solution to practical character recognition problems.

We should also outline some important points which is to be combined in ACR research areas.

- The statistics, relating to the factors such as character shape, slant and stroke variations and the occurrences of characters in a language and interrelations of characters, should be gathered.
- The feature set should represent structural and statistical properties properly so that they are sensitive to intra-class variances and insensitive to inter-class variances.
- Training sets should have considerable size and contain random samples including poorly written ones.
- Training and recognition processes should use the same knowledge-base. This knowledge-base should be built incrementally with several analysis levels by a cooperative effort from the human expert and the computer.

Based upon the above observations concerning the drawbacks and superiorities of the presented methods, a new hybrid approach which tries to maximize the superiority of HMM method and gets rid of some of the weaknesses of existing imperfect segmentation algorithms is proposed and presented in the following chapters.

CHAPTER 3

ISOLATED HANDWRITTEN CHARACTER RECOGNITION

This chapter describes a new handwritten character recognition scheme. The scheme first normalizes the character and then extracts a new set of one-dimensional discrete, constant length features to represent two dimensional structural shape information for HMM based recognition. The proposed feature set embeds the two dimensional information into a sequence of one-dimensional codes, selected from a code book. It provides a consistent normalization among distinct classes of shapes, which is very convenient for HMM based shape recognition schemes. The normalization parameters, which maximize the recognition rate are dynamically estimated in the training stage of HMM. This scheme is also used as a first level recognizer in the complete off-line cursive handwriting recognition system which will be described in Chapter 5.

3.1 Overview

HMM is a very powerful tool for modeling and recognition problems of one-dimensional signals. However, the extension of the HMM into two-dimensional image processing applications are not as successful as the one-dimensional cases. This is, basically, because of the requirement of large number of parameters for large amount of training data. On the other hand, the use of the one-dimensional HMM in the two-dimensional problems requires embedding the two dimensional information in the one-dimensional representation. In these applications, the major limitation is the difficulty of expressing

the neighborhood relation in the observation sequences. Therefore, the power of the Markovian property is not effectively reflected in the neighborhood relations.

HMM based recognizers are one of the most popular character recognition systems. There are many studies, which employ hundreds of features on various types of HMM, for solving the handwritten chracter recognition problem. However, none of the representations are sufficient to express full characteristics of handwriting. A good source of references in recent developments in hand-written character recognition by HMM can be found in [37] and [149].

In this study, the image grid is represented by a one dimensional equal length string of codes. For this purpose, a set of directional sparse skeletons of the binarized character shapes are extracted by scanning the image grid in various directions. The skeletons of the binary image is computed in each direction. Then, the directional skeletons are appended one after the other. Finally, the coordinates of the skeleton pixels are coded by the assigned code of the regions where the sparse skeleton pixels lie. This feature set embeds the two dimensional information into a sequence of one dimensional codes. It provides a consistent normalization among distinct classes of shapes. Then, the features are fed to a left to right discrete HMM. Training stage of HMM involves dynamical adjustments of the parameters of the normalized feature set.

3.2 Normalization

In their original form, the images are not tightly cut to the characters contained within them, leaving a border of whitespace around the characters of varying width (Figure 3.1.a). In order to reduce these inconsistencies, the boundaries of the image were cut to the bounding box of the character. For each edge of the image, an algorithm starts at the edge and examines each row/column moving inward. Once a row/column containing black pixels is found, this is taken as the new boundary of the image (Figure 3.1.b).

The character images are next normalized to a fixed size window. For this purpose, horizontal and vertical size normalizations are applied (Figure 3.1.c). Then, each normalized window is divided into equal regions in pre-defined directions. The size of the window, the number of scanning directions and the number of regions in each scanning direction are the normalization parameters. The parameters are taken as variables of

the training stage of HMM. The parameters which maximizes the HMM probabilities in the training data are estimated. Once the normalized window size, the corresponding scanning directions and number of regions are estimated, they are fixed for the recognition stage. The normalization process is not only for making the system size invariant, but also for making the recognition probabilities of the characters comparable.

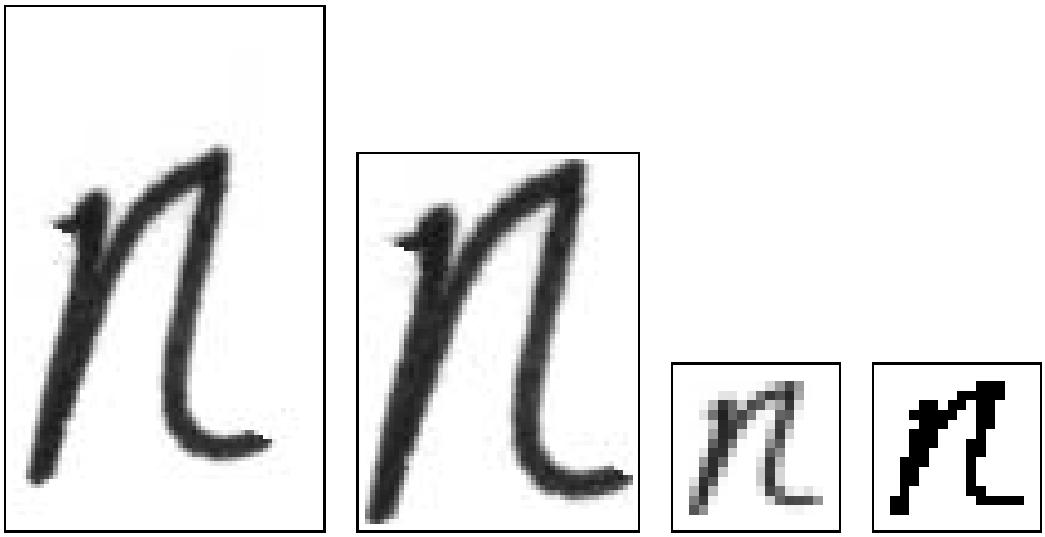


Figure 3.1: (a) Input character image, (b) connected component analysis (c) normalized image, (d) normalized binary image .

The normalized gray scale segment is binarized for feature extraction (Figure 3.1.d). There are various thresholding algorithms for binarization in the literature [39]. In this study, we implement two different thresholding technique; *locally adaptive* and *optimal thresholding*. However, it is observed that optimal thresholding gives better results than the locally adaptive one. Optimal thresholding is one of the global approaches with no parameters. The algorithm can be given as follows;

OPTIMAL THRESHOLDING FOR BINARIZATION

1. Initialize four corners of the image contain background pixels only and the remainder contains character pixels.

2. At step t, compute mean value of background and character pixels.

$$\mu_B^t = \frac{\sum_{(i,j)\in background} f(x,y)}{No.of background pixels},$$

$$\mu_O^t = \frac{\sum_{(i,j)\in character} f(x,y)}{No.of character pixels}$$

   where segmentation into background and objects at step t is defined by the threshold value T^t determined in the previous step;

3. Update $T^{(t+1)} = \frac{\mu_B^t + \mu_O^t}{2}$

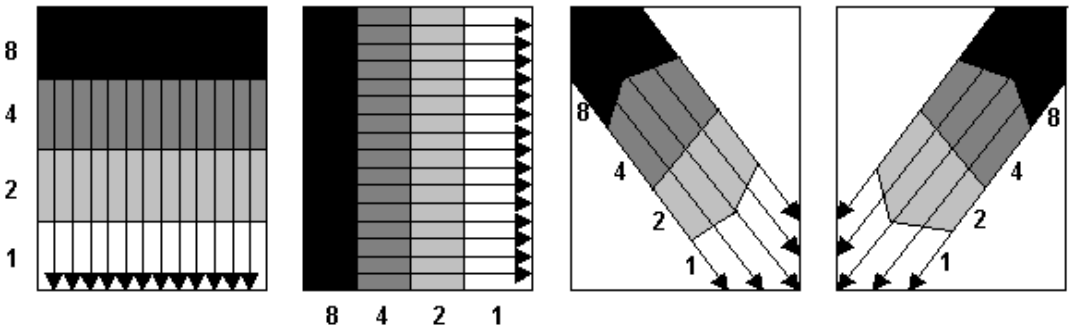
4. IF $T^{(t+1)} = T^{(t)}$,stop; ELSE go to *2*.

After the binarization, smoothing operation is performed in order to fill in the small holes and to delete protrusions.

Finally, the normalized window is scanned in various directions. The number of directions depends on the window size. For relatively small size, only the horizontal and vertical directions are considered. As the size of the window is increased, two or more diagonal directions are included into the scanning procedure. The scan lines in all directions are divided into equal regions (see Figure 3.2). The number of the regions also, depends on the size of the normalized window. Small windows are divided into two regions. As the size of the window grows, more regions are required to represent each character. Each region is coded with the powers of 2.

3.3 Feature Extraction

For feature extraction, the medians of the black runs are identified in each scan line. The code of the region, wherein the median of that run is located, represents that particular run. Each scan line is, then, represented by the sum of the codes of the regions where the medians of the runs are located. The size of the normalized window



(a) Vertical (b) Horizontal (c) Leftdiagonal (d) Rightdiagonal

Figure 3.2: Scanning in various direction each of which is divided into four regions and coded by power of 2.

and the number of the regions of the scan lines are selected in such a way that there exists, at most, one run in each region. This procedure is implemented as follows;

FEATURE EXTRACTION PROCEDURE

1. Construct and reset a bit stream which contains N bits, where N is the number of regions

$$bitstream[i] = 0, 0 \leq i \leq N-1$$

2. Load the bits which corresponds to the medians of the runs.

3. Find the value of this stream as the representative of that scan line as

$$\sum_{i=0}^{N-1} bitstream[i] * 2^i$$

Figure 3.3, indicates a binarized and normalized sample character, obtained as the output of the normalization stage. For this particular training set, the optimal normalized window size is 12x12 pixels, with four-directional scan lines, each of which is divided into four regions. Thus, the summation of the codes generates integers between 0 and 15 (8+4+2+1). This character is represented by a concatenated integer string of length 36 (12 values for columns and 12 values for rows, 6 values for each

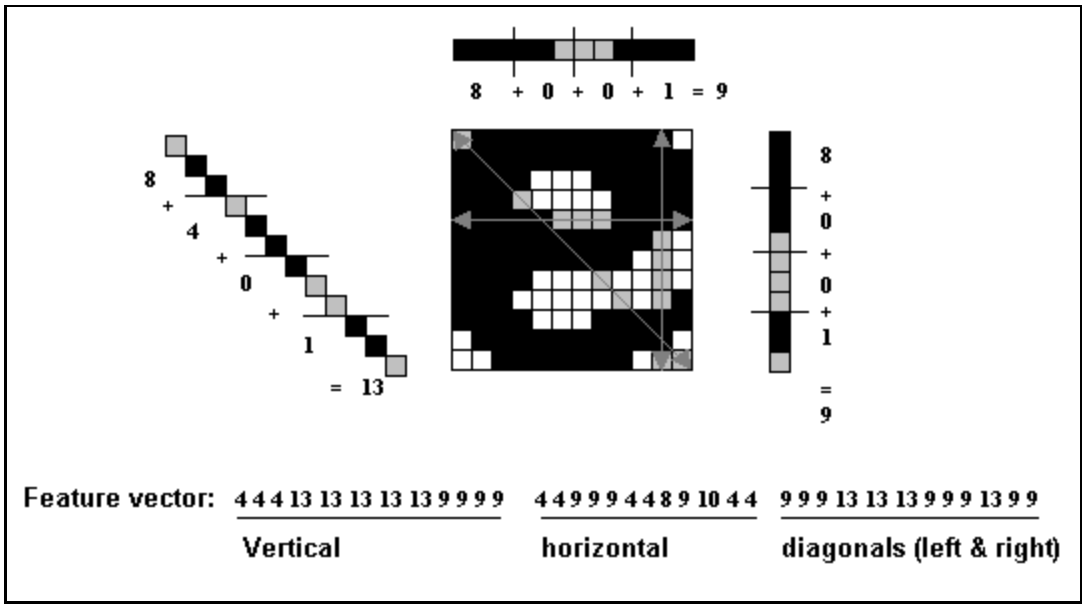


Figure 3.3: Feature extraction process for the character "e"

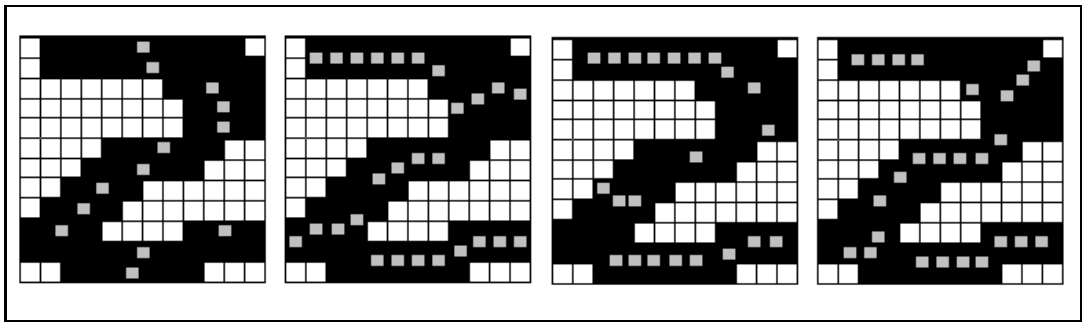
diagonal).

In the above feature extraction method, the median in each scanning direction represents a sparse directional skeleton of the character. It is well known that a skeleton of a character preserves almost all the shape information of a character. The directional skeletons, proposed in this study are much more convenient than the classical skeletons for representing the shape information of the characters in HMM based recognition schemes: First of all, concatenated directional skeletons preserves almost all the skeleton information of the characters. By increasing the number of scanning directions, it is possible to obtain curse to fine representation of the complicated skeletons. More importantly, in the proposed directional skeleton representation, there are no singularities such as cross points and branch points, as in the classical skeletons. These qualitative features form inconvenient bases for HMM recognizers. In the proposed directional skeleton representation there are no such points (see Figure 3.4).

3.4 Estimation of the Normalization Parameters in HMM Training

If we consider an individual, producing a handwriting, as a signal source, then the characters are the signal source outputs. Furthermore, the coded characters may be regarded as the output of a vector quantizer applied to the signal.

In this study, the discrete density left-to-right HMM is used as recognizer. The



(a)Horizontal (b)Vertical (c)Leftdiagonal (d)Rightdiagonal

Figure 3.4: Skeletons in four directions

discrete HMM for each character can be represented by $\lambda_l = \{\Delta, \Theta, S, T\}$, where $\Delta = \{\delta_{ij}\}$ is state transition probability matrix for transition from state i to state j, $\Theta = \{\theta_i(k)\}$ is the observation probability matrix of observing the code k in i^{th} state for $1 \leq i, j \leq S, i \leq j$, S is the number of states in the model, T is the number of distinct code symbols per state, $k \in V$ and $V = \{0, ..., M\}$ is the set of codes obtained in the feature set [6]. For the example in Figure 3.3, $M = 15$, and $T = 36$.

A training set is formed by using the isolated characters. This set is used to estimate the model parameters, as well as, the three normalization parameters mentioned in the previous section. For this purpose, a tree is formed with the nodes in the first level representing possible window sizes. The nodes in the second level represent the possible number of scanning directions for each window size and the third level represents the possible number of regions for each number of scanning directions. Then, the problem is to find the optimum path from the root to the leaf, which maximizes the recognition rate.

In other words, for each path, Δ and Θ parameters of HMM model for every character class is estimated via the Baum-Welch method [93]. Then, the characters in the training set are recognized via forward-backward algorithm [93]. The path, which gives the maximum recognition rate for the training set, is taken as the optimum path. During this process, the number of states of the HMM model is increased proportionally with the size of the normalized window, (from S=10 to 30 for our test data). The normalization parameters are fixed for the next stage of recognition.

The probability of observing the unknown character code sequence by a HMM is obtained as the sum of the probabilities of observing this sequence for every possible

state sequence of the HMM, i.e.:

$$P(O \mid \lambda_l) = \sum_{allQ} P(O, Q \mid \lambda_l),$$

where $O = (O_1, O_2, ..., O_T)$ is the observation sequence of the coded character, Q is the hidden state sequence which generates the given observation sequence O, and λ_l is the HMM model l^{th} character class. Adjusting the Δ and Θ parameters of a HMM model we may obtain high $P(O \mid \lambda)$ probability values for observations from the true class and low probabilities from false ones.

In the recognition stage, the probability of observing the coded character by every HMM is calculated. Then the observed string is labeled with the class which maximizes the probability $P(O \mid \lambda)$. Computation of $P(O \mid \lambda)$ for each λ_l requires an iterative process, called forward-backward algorithm.

The training and recognition algorithms will be discussed in Chapter 4 in more detail with the theoretical background and implementation issues.

3.5 Results

The proposed scheme is tested on three different database;

- NIST Handprinted Forms and and Characters Database,
- A local database collected from 19 persons,
- A free ware database generated by [152], download from Internet

All the performance evaluation of the character recognition scheme will be given in Chapter 6.

CHAPTER 4

THE HMM CLASSIFIER

The statistical methods of Hidden Markov Modeling have been found to be extremely useful for a wide spectrum of applications in many fields of pattern recognition. This technique was applied to speech recognition problems prior to its application to handwriting recognition. More recently, the benefits of their application to different forms of OCR have been shown. In particular, one problem that has proved troublesome using traditional methods of recognition is that of handwriting recognition. On-line handwriting recognition, in which the temporal information of how characters or words are constructed by the writer is available, is a very good example of the use of HMM approach due to its one dimensional nature as in the speech signals. In addition, methods of simulating temporal data from statical one have been applied HMM recognition for off-line handwriting and have shown promising results.

This chapter presents an introduction to the theories and algorithms for Hidden Markov Models, as well as the issues involved in their use in practical applications. The treatment here is greatly indebted to [93], [94] and [37].

4.1 Overview on Hidden Markov Models

The Markov Model is a mathematical tool for modeling a stochastic process as a sequence of observable outputs produced by an unknown source. Modeling a process this way allows us to potentially learn about the source that created the process without having to directly examine the source itself. With such a model we have the ability to recognize the patterns produced by the source or to predict the most probable future

observations of the process given a partial sequence of observations. The Markov Model makes the assumption that the process was produced by a Markov source, a type of source in which symbols currently produced are dependent only on a fixed number of symbols that have been produced preceeding the current output. The order of the model specifies the number of preceeding outputs which are taken into account for the next symbol to be produced. Since the complexity of the model grows exponentially with the order and the added benefit of increasing the order of the model decreases as the order grows higher, first-order or second-order Markov models are assumed to be sufficient for most of the applications.

The output corresponding to a single symbol can be characterized as discrete or continuous. Discrete outputs may be characters from a finite alphabet or quantized vectors from a codebook, while continuous outputs might be speech samples or music, which are represented as samples from a continuous waveform.

In general, a system can be classified in two categories: those whose statistical properties vary with time, and those whose statistical properties do not vary with time. These have been referred to as time-varying (or nonstationary) and time-invariant (or stationary), respectively. A time-invariant system could produce a sine wave, whose parameters (amplitude, freqency, and phase) remain constant over time, while an example of a time-varying system could produce a sound wave, which can be characterized by a combination of sine waves, whose frequencies, amplitudes, and phases change over time. A popular Markov Model approach represents a time-varying system as a combination of short time-invariant systems. These systems are what make up the output sequence (often referred to as the observation sequence) that represents the system.

A model may exploit some known properties of the system, or it may only rely on statistical properties of it. If the known properties of the system are used to benefit the model, the model is known as deterministic, and all that need be done is to estimate the parameters of the model (such as if we know the system produce sine wave). A model that relies only on the system's statistical properties is known as a statistical or stochastic model. In this case, it is assumed that the system can be characterized as a random process. The Markov Model is a type of stochastic model.

The Hidden Markov Model (HMM) is a doubly stochastic variant of the Markov Model, with an underlying stochastic process that is not observable (hidden), but can only be observed through another set of stochastic processes that produce the sequence

of observed symbols [94]. The underlying process of the HMM is a finite automaton (see Figure 4.1). It contains states, state transitions, and transition probabilities. Initial state probabilities exist for each state, which define the chances of the model being found in that particular state at the beginning of an observation sequence. Observation densities, which indicate the probability of a particular observation being produced by the model, are defined for each state, and these make up the observable stochastic process. In a given state, the observation probability density function for that state defines the expectancy of observations in that state, while the sequence of states traveled define a movement from one expectancy of observations to another. It is unknown to an outside observer what state the source (or the model representing the source) is in at any given time. All that can be observed is the sequence of symbols produced that make up the process. Therefore, this type of model is referred to as Hidden.

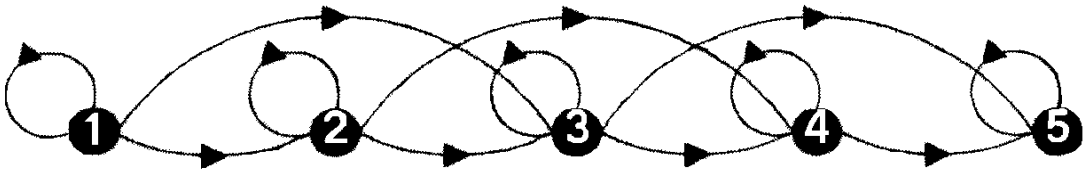


Figure 4.1: A left-right HMM.

A symbol is produced by a Hidden Markov Model by first selecting a starting state. For simplification, this can be thought of as time $t = 1$, although it may be the case that previous symbols produced may have helped determine what state the model is presently in. At this time, some symbol is produced which is the first observation of the output sequence. In deciding which symbol will be produced, the model will randomly select a symbol from a set of all possible symbols that can be produced by this state, with probability determined by the observation density function for that state of the model. At the next time segment, time $t = 2$, the model will move to a next state by selecting a state transition from the set of all possible transitions for the present state with probability determined by the present state's transition probability density function. Having arrived in this new state, the new state will select a symbol to be produced from its possible symbols with probability determined by this new state's observation density function. The model continues in this fashion where each increment

in time t brings another state transition followed by the emission of another symbol. The probability of the state sequence is equal to the product of the probability of the initial state and the probabilities of all transitions that were taken multiplied together, and the probabilities of all symbols that were emitted in each of these states are known. Therefore, we can easily calculate the probability of the state sequence occurring and the symbol sequence being produced by multiplying these values together. This is not the case, however, when trying to determine the probability that a symbol sequence was produced by a model when the state sequence is unknown. Such is the case when using Hidden Markov Models for recognition of patterns produced by a source, where we have no information on how these patterns were produced by the source and therefore it may be the case that many different state sequences could be a valid choice to produce this pattern. This will be the topic of Section 4.1.2.

4.1.1 Model Representation

As was described in the previous Section, each state in the model has a number of parameters associated with it which describe the probability of making a transition from that state to another state in the model, the probability of particular observations being produced while in that state, as well as the probability that a particular state was the start state for the sequence under observation.

We now define a notation, which will assist us in the discussion of these models.

Let,
$O = \{O_1, O_2, ..., O_T\}$, be the observation sequence
T is the length of the observation sequence
$Q = \{q_1, q_2, ..., q_N\}$, be the set of states in the model
N is the number of states in the model.
Additionally, for discrete HMMs:
$V = \{v_1, v_2, ..., v_M\}$, be the set of possible observations
M is the number of observation symbols.

For continuous HMM, the number of different possible observation symbols is infinite. In the discrete case, a Hidden Markov Model can be described by a set of parameters $\lambda = (\mathbf{A}, \mathbf{B}, \Pi)$, where $\mathbf{A}$ and $\mathbf{B}$ are matrices, and Π is a vector. The differences for the continuous case will be discussed in Section 4.1.5.

Matrix **A** is defined as the set of all state transition probability distributions for the model:

$$a_{ij} = P\{q_j \text{ at } t+1 \mid q_i \text{ at } t\}, \tag{4.1}$$

where q_i is the state the model is in at some time t, q_j is a possible next state, and a_{ij} is the probability that the model will make the transition to this next state at the next time increment.

Matrix **B** is defined as the set of all observation probability distribution functions for the model:

$$b_j(k) = P\{v_k \text{ at } t \mid q_j \text{ at } t\}, \tag{4.2}$$

where q_j is the state the model is in at some time t, v_k is an observation that may be seen by the model, and $b_j(k)$ is the probability of this observation occurring in the observation sequence while in state q_j

Vector Π is defined as the initial state distribution for the model:

$$\pi_i = P\{q_i \text{ at } t = 1\}, \tag{4.3}$$

where q_i is a state in the model and π_i is the probability that the model will begin in state q_i at the beginning of an observation sequence.

4.1.2 Model Evaluation

So far, we have viewed a sequence of symbols as being produced by a model, but when an HMM is used for the purpose of classification, we view a sequence of symbols as having been produced by unidentified sources. The goal of classification is then to decide which source this observed sequence was produced by. In this case a number of HMMs exist, one to model each respective source. The sequence of symbols is presented to each model and its probability of being produced by that model is evaluated. The model that has the greatest likelihood of producing this observation sequence then defines the chosen identity of the source which is believed to have produced it. In order to evaluate models, we must first calculate $P(O \mid \lambda)$, the likelihood that a sequence of observations, O, were produced by a given model, λ. For any observation sequence produced by a model and a known sequence of states followed to produce this sequence of observations, the probability of the model producing these observations and following

this state sequence can be calculated as:

$$P(O, s \mid \lambda) = \pi_{s_1} b_{s_1}(O_1) \prod_{i=2}^{T} a_{s_{i-1}s_i} b_{s_i}(O_i) \quad . \tag{4.4}$$

In most practical situations, however, the state sequence is unknown to the observer. Since there may be many state sequence paths that can produce the underlying observation sequence, the probability that O was produced by model λ is the sum of all the probabilities $P(O, s \mid \lambda)$ for all possible state sequence paths, s. Computing the proabilities for each of these paths would be computationally very expensive, requiring on the order of $2TN^T$ calculations. Fortunately; an efficient algorithm using a set of variables called forward probabilities exists which can calculate this sum of probabilities on the order of N^2T calculations. This is known as the Forward-Backward algorithm since it can be similarly defined using backward probabilities.

Define $\alpha_t(i)$, the forward probability, as the probability of the partial observation sequence from time 1 to time t occurring and the model being in state q_i at time t.

$$\alpha_t(i) = P(O_1 \ O_2 \ ... \ O_t, \ q_t = s_i \mid \lambda) \quad . \tag{4.5}$$

This can be solved for inductively as follows:

FORWARD PROBABILITIES

1. *Initialization:*

$$\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N$$

2. *Induction:*

For all $t = 1, 2, ..., T-1$, calculate

$$\alpha_{t+1}(j) = (\sum_{i=1}^{N} \alpha_t(i) a_{ij}) b_j(O_{t+1}), \quad 1 \leq j \leq N \tag{4.6}$$

3. *Termination:*

$$P(O \mid \lambda) \ = \sum_{i=1}^{N} \alpha_T(i) \tag{4.7}$$

In a similar fashion, the backward probabilities, $\beta_t(i)$, can be defined as the probability of the partial observation sequence from time $t+1$ to time T occurring, and the model being in state q_i at time t.

BACKWARD PROBABILITIES

1. *Initialization:*

$$\beta_T(i) = 1, \quad 1 \leq i \leq N$$

2. *Induction:*

For all $t = T-1, T-2, ..., 1$, calculate

$$\beta_t(i) = \sum_{i=1}^{N} a_{ij} b_j(O_{t+1}(j)), \quad 1 \leq j \leq N \tag{4.8}$$

The backward probabilities will be used later. The underlying principle of the forward- backward algorithm is the following: If we know the probabilities, for each state q_i, of the partial sequence of observations up to time t leaving the model in state q_i, the probability of the model being in some state q_j at time $t+1$ can be calculated using only those probabilities, and those for times before t are not needed. Therefore. the forward probabilities act as accumulators for each state as we increase t from 1 to T, and the value at time T is the probability of being in that state at the end of the observation sequence.

The forward and backward probabilities have been found to be useful for other computations involving state path traversals. This will be shown in algorithms used to reestimate model parameters presented in the next Section.

With the help of the forward and backward probabilities, we now have the probability that an observation sequence was produced by a given model. In order to be able to choose one competing model over another we must be able to compare, for each model, the probability that given the observed Sequence, the model produced it. This can be calculated using the Bayes Rule.

4.1.3 Estimating Model Parameters

In creating a model of a source, the details of the source's inner workings are mostly unknown. We need to be able to define the characteristics of our model based on previous observation sequences, or *training examples*. Therefore, we require a method of estimating the model parameters using these examples. The most common method

of estimating a model's parameters is to set them so as to maximize the probability of the training observation sequences being produced by the model [93]. This is a maximum likelihood method of training. Since an analytical solution to estimating the parameters is intractable, the parameters must be estimated iteratively or by some method of gradient descent. A very common iterative re-estimation method is the Baum-Welch algorithm [95]:

First define the probability of the model λ being in state q_i at time t when presented with the sequence of observations O,

$$\gamma_t(i) = P(q_i \text{at } t \mid O, \lambda)$$

and the probability of the model λ being in state q_i at time t and making a transition to state q_j at time $t+1$ when presented with the sequence of observations O,

$$\xi_t(i,j) = P(q_i \text{at } t, q_j \text{at } t+1 \mid O, \lambda) \quad .$$

The above probabilities can be calculated using forward and backward probabilities as follows:

BAUM-WELCH ALGORITHM

1. *Calculate* $\gamma_t(i)$ and $\xi_t(i,j)$

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{P(O \mid \lambda)} \quad 1 \le i \le N, \quad 1 \le t \le T \tag{4.9}$$

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(i)}{P(O \mid \lambda)} \quad 1 \le i,j \le N, \quad 1 \le t \le T \tag{4.10}$$

2. *Update the model parameters;*

$$\bar{\pi}_i = \gamma_1(i) \tag{4.11}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{4.12}$$

$$\bar{b}_j(k) = \frac{\sum_{t=1 O_t=k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \tag{4.13}$$

Each iteration of the Baum-Welch algorithm has been proven to increase $P(O \mid \lambda)$ for that instance of O until a local maximum has been reached [95]. The Baum-Welch algorithm is by far the most well-known parameter re-estimation technique for HMMs. This does not mean, however, that it is the only technique, or even necessarily the optimal technique for re-estimation. Gradient descent techniques have also been applied with good results [96]. Levinson et.al, also discuss the use of Lagrange multipliers for parameter estimation. In addition, it should be pointed out that all these estimations are maximum-likelihood estimators, since they focus on maximizing an individual model's likelihood of having produced observations from the source that it is modeling, but does not take into consideration the likelihood of competing models producing these same observations. This may produce sub-optimal decision boundaries. A better estimator would be one that is based on Maximum Mutual Information (MMI) [97]. This method focuses on optimizing the parameters of the set of models simultaneously, making adjustments so that an observation sequence O_v, that was produced by a source v modeled by model λ_v, has a higher probability of being produced by λ_v, than any model other than v. Unfortunately, the amount of computation required makes MMI training infeasible for most HMM applications.

A similar method, known as Corrective Training [98] focuses on minimizing recognition errors by iterative parameter readjustment and has been shown to give better results than Maximum Likelihood Estimation. At each iteration, every incorrectly classified signal element causes the model which was incorrectly chosen to update its parameters so as to decrease that element's probability of being produced by that model. Although Corrective Training can produce better results than maximum likelihood training, it is still computationally more expensive and is not guaranteed to converge.

Another alternative to the Maximum Likelihood estimator is based on Minimum Discrimination Information (MDI) [99]. This attempts to minimize the discrimination information between the probability densities of the HMM and the true densities of the source being modeled. Unfortunately, obtaining such an minimum is highly non-trivial and therefore training methods based on this approach are scarce.

4.1.4 Picking an Optimal State Sequence

Given a model and a sequence of observations, one problem that needs to be addressed is the selection of an optimal sequence of states traversed to create this observation sequence. It is more than likely the case that many possible state sequences could produce a given observation sequences. One way of selecting a state sequence is to, for each observation of a sequence, choose those states that have the highest individual probability of producing that observation. The problem with this method is that, having not taken into account the actual state connectivity, an impossible state sequence might be selected. A method of selecting the best path over state transitions is required. A well known efficient method is a dynamic programming algorithm known as the *Viterbi Algorithm* [100], [101]. The algorithm obtains a solution recursively as follows:

VITERBI ALGORITHM

1. *Initialization:*

$$\delta_1(i) = \pi_i b_i(O_1), 1 \leq i \leq N \tag{4.14}$$

$$\Psi_1(i) = 0 \tag{4.15}$$

2. *Recursion:*

$$\delta_t(j) = \max_{1 \leq i \leq N} (\delta_{t-1}(j) a_{ij}) b_j(O_t),\ 2 \leq t \leq T,\ 1 \leq j \leq N \tag{4.16}$$

$$\Psi_t(j) = \arg \max_{1 \leq i \leq N} (\delta_{t-1}(j) a_{ij}),\ 2 \leq t \leq T,\ 1 \leq j \leq N \tag{4.17}$$

3. *Termination:*

$$P^* = \max_{1 \leq i \leq N} (\delta_T(i)) \tag{4.18}$$

$$i_T^* = \arg \max_{1 \leq i \leq N} (\delta_T(i)) \tag{4.19}$$

4. Path Backtracking:

$$i_t^* = \Psi_{t+1}(i_{t+1}^*),\ t = T-1, T-2, ..., 1 \tag{4.20}$$

The Viterbi algorithm is very similar to the Forward-Backward algorithm and in

fact, could be used in place of the Forward-Backward algorithm to evaluate a model, given the observation sequence. The Viterbi Algorithm contains an additional variable to the Forward variable $\alpha_t(i)$, denoted as $\delta_t(i)$. While $\alpha_t(i)$ represents the sum of probabilities of all possible paths into a state q_i at time t, $\delta_t(i)$ is the maximum of these different path probabilities. $\delta_T(i)$ is then the probability that the maximum probability path ends in state i, and P^* is the total maximum path probability of all paths ending in any on the N states. Similarly, i_t^* is the final state that the sequence will end in with probability P^*. Therefore, by starting at state i_T^* and backtracking using $\Psi_t(i)$ to reach state i_{t-1}^*, from any state in the path i_t^*, we can completely reconstruct this path.

4.1.5 Observation Densities

Each state of an HMM has an observation density function, $b_j(O)$, that describes the probability that a given observation O was created by the model while in this state. Since usually neither the true densities nor reliable approximation of these densities are known in advance, an approximation must be done using the training data. Three main types of observation probability densities can be found in the literature that are used in HMMs.

- *Discrete densities* assume a finite variety of observation vectors exist, such that the probability of each occurrence can be estimated.
- *Continuous densities*, on the other hand, represent the output distributions by a continuous distribution which is often estimated as a mixture of Gaussian densities.
- *Semi-continous densities* (also known as tied mixture models) are a technique of creating observation densities as a weighted combination of Gaussian prototypes created by a clustering of the data independent of its class.

4.1.6 Discrete Densities and Vector Quantization

Discrete density HMMs represent every observation as one of a number of discrete vector prototypes. By observing the frequency of each of these prototypes, the probability of each can be estimated. In their original form, the observations may take the same form

as these prototype vectors, take on a larger variety of discrete values, or may take on values from some continuous distribution. In either of the last two cases, before training a model we must first create a *codebook* of prototype vectors using *vector quantization* [102] so that any observation can be represented by one of these prototypes. When creating this codebook, a predetermined number of clusters must be created that will partition the input space into linearly separable regions, one for each codebook vector. This partitioning is most often done by selecting an initial set of prototype vectors and iteratively improving on them.

Gray et.al, [102] describes methods of selecting an initial codebook. Methods of random selection include selecting the first set of observations from the training data to act as the initial codebook vectors, and selecting training observations at regular intervals to act as these initial vectors, the latter of which acts as a better starting point if data are highly correlated.

Once an initial set of codebook vectors has been selected, training observations are clustered by classifying each to its closest prototype vector. Using these new clusters each codebook vector can be recalculated to more accurately reflect the real cluster of vectors that are falling within its cluster. It has been shown that the best vector quantized representation of a cluster of vectors is that cluster's centroid. With this new codebook, iterations of classifying the training data and re-adjusting the codebook vectors are continued until the average distortion induced by this representation falls below some threshold level. Often a maximum number of iterations is specified for this algorithm to remedy the case where the average distortion below a certain threshold can not be obtained.

Clusters can be created using data from the entire set of training observations as a single pool of data or by using class labels, provided with the training observations to create clusters from each individual class pool of data. The former technique is kuown as intrinsic or unsupervised clustering, while the latter is known as extrinsic or supervised, clustering [103] . As an example, these labels may be characters from an alphabet, words, phonemes, or some smaller sub-grouping of one of these. Labels are often provided by humans through a very tedious job called *truthing*. Due to the large number of man-hours that this job can require (which grows proportional to the size of the training data pool) it may be the case that higher level labels (such as word labels) will be available, but labels of higher granularity, such as characters, are not

available. In this case a supervised clustering can be performed at the word level, while unsupervised clustering must be performed at the sub-word level.

When vector quantization is used to create a finite number of discrete prototypes to represent continuous data, this representation becomes a compressed version of the original data. Associated with this compression will be a quantization error. In order to construct good vector quantized codebooks, we need a measure of the distortion imposed on the data by this representation. One popular distortion measure used in vector quantization is the Mean Squared Error (MSE). This is basically the sum of squared Enclidean distances between each observation vector and the quantized vector representing it.

4.1.7 Model Topologies

By placing certain restrictions on the topology, the model will tend to behave differently. A model in which all states can be reached from any state is known as an *ergodic model*. For certain types of processes, other types of models have been shown to characterize the properties of the system better [93]. In the case of a time-variant system, a model that inherently represents the passage of time may be advantageous.

The *left-right* model, also known as the Bakis model, is such a model (see Figure 4.1). The underlying state sequence of this type of model is forced to start in the initial state (i.e., the leftmost state) and can only make transitions to higher states (i.e. to the right) or back to the present state as time passes. In addition, the model must end in the final state (the rightmost state). This allows the sequence of states to represent the passage of time and has become the most popular form of Markov model used in speech recognition and handwriting recognition. It is often the case that such models are simplified to only allow jumps of no more than two states, and this was the typical method of restricting jumps in our experiments.

Another form of state transition that has been found useful is known as the Null transition [104]. The idea of the null transition is that a state transition should be allowed to occur without requiring an observation to be absorbed. What this amounts to is that null transitions are a special kind of state transition, with transition probabilities, that can occur without requiring the time counter, t, to be updated. The present observation in the sequence, O_t, will remain the present observation in the next state.

A null transition from the end state of the model back to the start state will create a model capable of observing an arbitrary length string of elements which the model represents.

Due to self state transitions, a_{ii}, the Hidden Markov model can remain in the same state throughout many transitions. For a given state, the number of transitions for which the model remains in the same state is known as the state's duration. This can be easily formulated as [93]:

$$p_i(d) = a_{ii}^{d-1}(1 - a_{ii}), \tag{4.21}$$

where $p_i(d)$ is the probability of the model being in state q_i, for d consecuitive observations. This is an exponential distribution and is more than likely not the most appropriate model of the true duration density. One possible solution, in the left-right model, is to create a number of consecutive duplicate states to model the state duration.

As an alternative and a more accurate model of the duration, the self transitions can be eliminated and a duration density can be estimated for each state (Rabiner and Juarig [1986]). Tracing an observation sequence through the model now requires not only tracing each possible path, but also must take into account all possible durations in each state. This same problem is encountered during model parameter estimation. A restriction must be placed on the model to limit the maximum duration to a predefined value, D, otherwise the number of duratious that need to be checked would only be limited by the length of the observation sequence. Even with the restriction, the added complexity of modeling these state durations using this non-parametric approach is great, requiring about D times the storage and $D^2/2$ times the computation. Also, this method requires an additional $D - 1$ parameters per state (minus one because of loss of self transitions), and less data to estimate state transitions (since every state now claims multiple observations for $d > 1$). Parametric state duration densities, such as a Gaussian distribution or a gamma distribution of the duration, have been proposed as a simpler alternative duration estimators. A great deal of effort was spent on finding appropriate topologies for these models.

4.2 Implementation Issues for HMMs

The discussion in the previous section has primarily dealt with the theory of HMM and several variations on the form of the model. In this section, we deal with several practical issues including scaling, multiple observation sequences, initial parameter estimates, missing data, and choice of model size and type.

4.2.1 Scaling

In order to understand why scaling is required for implementing the re-estimation procedure of HMMs, consider the definition of $\alpha_t(i)$ in equation 4.5. It can be seen that $\alpha_t(i)$ consists of the sum of a large number of terms, each of the form

$$\left(\prod_{s=1}^{t-1} a_{q_s q_{s+1}} \prod_{s=1}^{t} b_{q_s}(O_s)\right)$$

with $q_t = S_i$. Since each a and b term is significantly less than 1, it can be seen that as t starts to get big, each term of $\alpha_t(i)$ starts to head exponentially to zero. For sufficiently large t (e.g., 100 or more) the dynamic range of the $\alpha_t(i)$ computation will exceed the precision range of essentially any machine. Hence, the only reasonable way of performing the computation is by incooporating a scaling procedure.

The basic scaling procedure is used to multiply $\alpha_t(i)$ by a scaling coefficient that is independent of i (i.e., it depends only on t), with the goal of keeping the scaled $\alpha_t(i)$ within the dynamic range of the computer for $1 \leq t \leq T$. A similar scaling is done to the $\beta_t(i)$ coefficients and then, at the end of the computation, the scaling coefficients are canceled out exactly.

To understand the scaling procedure better, consider the re-estimation formula for the state transition coefficients a_{ij}. If we write the reestimation formula 4.12 directly in terms of the forward backwrad variables we get

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(i)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(i)} \quad . \tag{4.22}$$

Consider the computation of $\alpha_t(i)$. For each t, we first compute $\alpha_t(i)$ according to the induction formula 4.6, and then we multiply it by a scaling coefficient c_t, where

$$c_t = \frac{1}{\sum_{i=1}^{N} \alpha_t(i)} \quad . \tag{4.23}$$

Thus for a fixed t, we first compute

$$\alpha_t(j) = \left(\sum_{i=1}^{N} \bar{\alpha}_{t-1}(i) a_{ij} \right) b_j(O_t) \quad . \tag{4.24}$$

Then, the scaled coefficient set $\bar{\alpha}_t(i)$ is computed as

$$\hat{\alpha}_t(j) = \frac{\left(\sum_{i=1}^{N} \hat{\alpha}_{t-1}(i) a_{ij} \right) b_j(O_t)}{\sum_{j=1}^{N} \left(\sum_{i=1}^{N} \hat{\alpha}_{t-1}(i) a_{ij} \right) b_j(O_t)} \quad , \tag{4.25}$$

By induction we can write $\bar{\alpha}_{t-1}(i)$ as

$$\hat{\alpha}_{t-1}(i) = \left(\prod_{T=1}^{t-1} c_T \right) \alpha_{t-1}(i) \quad . \tag{4.26}$$

Thus we can write $\hat{\alpha}_t(i)$

$$\hat{\alpha}_t(i) = \frac{\alpha_t(i)}{\sum_{i=1}^{N} \alpha_t(i)} \quad , \tag{4.27}$$

i.e., each $\alpha_t(i)$ is effectively scaled by the sum over all states of $\alpha_t(i)$.

Next, we compute the $\beta_t(i)$ terms from the backward recursion. The only difference here is that we use the same scale factors for each time t for the β's as was used for the α's. Hence the scaled β's are of the form

$$\hat{\beta}_t(i) = c_t \beta_t(i) \quad . \tag{4.28}$$

Since each scaled factor effectively restores the magnitude of the α terms to 1 and since the magnitudes of the α and β terms are comparable, using the same scaling factors on the β's as was used on the α's is an effective way of keeping the computation within reasonable bounds. Furthermore, in terms of the scaled variables we see that the reestimation equation 4.22 becomes

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \hat{\alpha}_t(i) a_{ij} b_j(O_{t+1}) \hat{\beta}_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \hat{\alpha}_t(i) a_{ij} b_j(O_{t+1}) \hat{\beta}_{t+1}(j)} \quad , \tag{4.29}$$

but each $\hat{\alpha}_t(i)$ can be written as

$$\hat{\alpha}_t(i) = \left(\prod_{s=1}^{t} c_s \right) \alpha_t(i) \;=\; C_t \alpha_t(i) \tag{4.30}$$

and each $\hat{\beta}_{t+1}(j)$ can be written as

$$\hat{\beta}_{t+1}(j) = \left(\prod_{s=t+1}^{T} c_s \right) \beta_{t+1}(j) \;=\; D_{t+1} \beta t + 1(j) \quad . \tag{4.31}$$

Thus equation 4.29 can be written as

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} C_t \alpha_t(i) a_{ij} b_j(O_{t+1}) D_{t+1} \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} C_t \alpha_t(i) a_{ij} b_j(O_{t+1}) D_{t+1} \beta_{t+1}(j)} \quad . \tag{4.32}$$

Finally, the term $C_t D_{t+1}$ can be seen to be of the form

$$C_t D_{t+1} = \prod_{s=1}^{t} c_s \prod_{s=t+1}^{T} c_s \;=\; C_T \quad , \tag{4.33}$$

independent of t. Hence the terms $C_t D_{t+1}$ cancel out both in the numerator and denominator of 4.32 and the exact reestimation equation is realized.

It should be obvious that the above scaling procedure applies equally well to reestimation of the π or B coefficients. It should also be obvious that the scaling procedure need not be applied at every time instant t, but can be performed whenever necessary. If scaling is not performed at some instant t, the scaling coefficiens c_t are set to 1 at that time all the conditions discussed above are met.

The only real change in the HMM procedure, because of scaling, is the procedure for computing $P(O \mid \lambda)$. We can not merely sum up the $\hat{\alpha}_T(i)$ terms, since these are scaled already.

However we can use the property that

$$\prod_{t=1}^{T} c_t \sum_{i=1}^{N} \alpha_T(i) \;=\; C_T \sum_{i=1}^{N} \alpha_T(i) \;=\; 1 \quad , \tag{4.34}$$

$$\prod_{t=1}^{T} c_t \; P(O \mid \lambda) \;=\; 1 \quad , \tag{4.35}$$

$$P(O \mid \lambda) \;=\; \frac{1}{\prod_{t=1}^{T} c_t} \quad and \tag{4.36}$$

$$\log\big(P(O \mid \lambda)\big) \;=\; -\sum_{t=1}^{T} \log c_t \quad . \tag{4.37}$$

Thus the log of P can be computed, but not P. It would be out of the dynamic range of the machine anyway.

Finally we note that, when using the Viterbi algorithm to give the maximum likelihood state sequence, no scaling is required if we use logarithms in the following way. We define

$$\phi_t(i) \;=\; \max_{q_1, q_2, \ldots, q_t} \log P\big(q_1 q_2 \ldots q_t, O_1, O_2, \ldots, O_t \mid \lambda\big) \tag{4.38}$$

and initially set

$$\phi_t(i) \;=\; \log(\pi_i) + \log\big(b_i(O_1)\big) \quad , \tag{4.39}$$

with the recursion step,

$$\phi_t(j) \;=\; \max_{1\le i\le N}\big(\phi_{t-1}(i) + \log a_{ij}\big) + \log b_j(O_t)\big) \tag{4.40}$$

and termination step

$$\log P^* \;=\; \max_{1\le i\le N}\big(\phi_T(i)\big) \quad . \tag{4.41}$$

Again, we arrive at $\log P^*$ rather than P^*, but with significantly less computation and with no numerical problems.

4.2.2 Multiple Observation Sequences

The major problem with left-right models is that one can not use a single observation sequence to train the model. This is because the transient nature of the states within the model only allow a small number of observations for any state (until a transient is made to a successor state). Hence, in order to have sufficient data to make reliable estimates of all model parameters, one has to use multiple observation sequences.

The modification of the re-estimation procedure is straightforward and goes as follows. We denote the set of K observation sequences as

$$O \;=\; (O^{(1)}, O^{(2)}, ..., O^{(k)}) \quad , \tag{4.42}$$

where $O^{(k)} = (O_1^{(k)}, O_2^{(k)}, ..., O_{T_k}^{(k)})$ is the kth observation sequence. We assume each observation sequence is independent of every other observation sequence, and our goal is to adjust the parameters of the model λ to maximize

$$P(O \mid \lambda) \;=\; \prod_{k=1}^{K} P(O^{(k)} \mid \lambda) \;=\; \prod_{k=1}^{K} P_k \quad . \tag{4.43}$$

Since the re-estimation formulas are based on frequencies of occurrence of various events, the re-estimation formulas for multiple observation sequences are modified by adding together the individual frequencies of occurence for each sequence. Thus, the modified reestimation formulas for $\bar{a_{ij}}$ and $\bar{b_j(l)}$ are

$$\bar{a}_{ij} = \frac{\sum_{k=1}^{K}\frac{1}{P_k}\sum_{t=1}^{T_k-1}\alpha_t^k(i)a_{ij}b_j(O_{t+1}^k)\beta_{t+1}^k(j)}{\sum_{k=1}^{K}\frac{1}{P_k}\sum_{t=1}^{T_k-1}\alpha_t^k(i)\beta_t^k(i)} \tag{4.44}$$

and

$$\bar{b}_i(l) = \frac{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1_{O_t=v_t}}^{T_k-1} \alpha_t^k(i)\beta_t^k(i)}{\sum_{k=1}^{K} \frac{1}{P_k} \sum_{t=1}^{T_k-1} \alpha_t^k(i)\beta_t^k(i)} \tag{4.45}$$

and π_i is not reestimated since $\pi_1 = 1, \pi_i = 0, i \neq 1$.

The proper scaling of (4.44) and (4.45) is now straightforward since each observation sequence has its own scaling factor. The key idea is to remove the scaling factor from each term before summing. This can be accomplished by writing the reestimation equations in terms of the scaled variables. In this manner, for each sequence $O_{(k)}$, the same scale factors will appear in each term of the sum over t as appears in the P_k term, and hence will cancel out exactly. Thus, using the scaled values of the alphas and betas results in an unscaled $\bar{a_{ij}}$ and $b_j\bar{(l)}$ terms.

4.2.3 Lack of Training Data

Another problem that is encountered during model parameter re-estimation, due to a finite size training set, which occurs when a model parameter is set to zero. For instance, after training the model, the probability of seeing an observation, v_k, in state q_j is set to zero (i.e. $b_j(k) = 0$). The true distribution of the source from which the training observations were produced may be such that it is possible for an outcome to be produced in which v_k is observed in state q_j. However, since this case did not ever occur in the training data, this model will never recognize such outcome since $b_j(k) = 0$ states that this is an impossible observation. This problem can easily be overcome by restricting all parameters from falling below some minimum value. Modifications of the Baum-Welch algorithm have been formulated to take into account this new constraint [96] . In our experiments a minimum value of 0 for all varianes on the diagonal of the covariance matrix was enforced.

Lack of training data can often result in parameters receiving poor values and certain characteristics of the source being poorly modeled or even left out all together. This may possibly be remedied by reducing the size of the model, thus reducing the number of parameters that need to be estimated. Unfortunately, if the structure of the model was chosen because it is believed to accommodate some inherent characteristics of the source, changing this is undesirable. Jelinek et.al, [105] have devised a method of interpolating the parameters of a model with the parameters of a reduced model for

which there is adequate training data. In our experiments, categories for which there was not enough data to train an individual model were combined into meta-categories. This does not cause a problem if the goal of the system does not involve discriminating between the categories within these meta-categories.

4.2.4 Initial Parameter Values

During the training of a model, the model parameters are estimated using training examplcs as described previously. Both iterative methods of estimating these parameters, such as the Baum-Welch algorithm, and gradient descent methods are based on creating a new set of parameters using results obtained from the model with a presently defined set of parameters. Therefore, some initial set of parameters must be picked in order for either of these methods of training to be possible. Since neither method of training guarantees an optimal solution due to the possibility for getting stuck at a local minima, the choice of a good starting point can have an impact on how well the model will train.

Many of the initial parameter decisions must be made by some intuitive guesses and/or trial and error. Parameters such as the number of states and the number of mixtures used to estimate the observation distribtitions, will have a great impact on how each training example effects are distributed across the model. For models with a relatively small number of states, the observation density estimates for each state will be well trained due to the relatively large number of samples that will go into their training. However, much of the contextual information that could be trained by spreading this data over more states may be lost.

The topology of the model is a factor that is greatly influenced by the initial values of the state transition matrix. If an infinite number of training examples were available, we could set all state transitions more or less equal, and let the data decide which transitions are most important. Since, in the real world, lack of data is always a problem, we can greatly help the model to focus on the important parameters by choosing which of these transitions should not be allowed, and which ones will probably be chosen over others. By setting the disallowed transitions to zero and more probable transitions to higher values than the less probable transitions, the training process can concentrate on the important parameters. In our experiments, it was discovered that

the initial values of parameters such as those that define the model topology, the number of Gaussian mixtures, and the dimensionality of the feature vectors have an impact on the efectiveness of the model. However, the exact initial value of the parameters, such as the initial values of the Gaussian means and covariances, and small changes to the initial values of the transition matrix, tend to have little effect on the model after training.

4.2.5 Choice of Model

Unfortunately, there is no simple, theorotically correct way of making the choice of the type of model, the choice of model size and the choice of observation symbols. These choices must be made based on trial and error depending on the process being modeled.

CHAPTER 5

CURSIVE HANDWRITING RECOGNITION

In this chapter, a new scheme is proposed for off-line handwritten connected character recognition problem, which uses a sequence of segmentation and recognition algorithms. The proposed system assumes no constraint in writing style, size or variations.

First, a segmentation method finds the character segmentation paths by combining the gray scale and binary information. Then, as we presented in chapter 3 a sequence of codes is extracted from the segments as a feature set. For each training data, the parameters of the feature set are estimated by maximizing the HMM training stage. The features are fed to a high order Hidden Markov Model (HMM), as a preliminary recognition step. Finally, in order to confirm the segmentation paths and recognition results, a recognition based segmentation method is presented.

5.1 System Overview

The most difficult problem of the field of ACR is the recognition of off-line unconstrained cursive handwriting. The tools for modeling almost infinitely many variations in human handwritting, the overlaps and interconnection of the neighboring characters are very primitive yet. Furthermore, when observed in isolation, characters are often ambiguous and require context to minimize the classification error. Thus, current research aims at developing constrained systems for limited domain applications [152], [153], [154]. A well- defined lexicon plus a well-constraint syntax help provide a feasible solution to the problem.

Most of the recognition methods employ binary images in the segmentation and

recognition stages [56], [149], [155]. However, through the binarization of the gray scale image, useful information is lost. In order to avoid the limitation of binary image, some recent methods [156] use gray level image. However, this time the significant details suppress the important shape information.

The major contribution of this study is the dynamic utilization of the gray scale and binary information in a mixed way to extract the maximum amount of information for both segmentation and recognition stages. A new segmentation algorithm extracts the characters in free style handwriting. In the proposed methodology, the character segmentation regions are determined by using the boundary information of the image. Then a nonlinear character segmentation path in each character segmentation region is found by using multi-stage graph search algorithm. Recognition of each segment is accomplished in two stages as we described in chapter 3: In the first stage, the features extracted from the each segment of binary image, is fed to a left-right Hidden Markov Model (HMM) for training and recognition. Finally, a recognition based segmentation algorithm resolves handwriting strings by maximizing a cumulative information measure, computed from the HMM probabilities.

The chapter is organized as follows. The classical preprocessing operations and some statistical information estimations are described in Section 5.2, The proposed segmentation method is explained in Section 5.3. The feature set and HMM based recognition are summarized in Section 5.4. The imperfect results of segmentation and HMM recognition are improved in a recognition based segmentation method in Section 5.5.

5.2 Preprocessing and Normalization

The input image is preprocessed for noise reduction and binarization using a 2 dimensional Gaussian filter and Optimal thresholding method, respectively [39]. Then, the connected characters are decomposed into rectangular units (RU). A RU is the minimum bounding rectangle of a set of connected characters. It is the smallest unit of the image, which may contain a portion, a whole or more than one character up to a whole word. The eight-neighbor boundary extraction algorithm [1] finds the closed contours in the binary RU. The character contours are smoothed by an averaging mask. Finally, we have the gray scale and binary RUs in order to be used in the subsequent stages of

the system.

After preprocessing, average stroke width/height estimation, reference line finding and slant angle detection are performed in binary image. These parameters are used in the segmentation algorithm.

5.2.1 Stroke Width/Height Estimation

A simple program is developed to estimate the average stroke width, which is then used to decide whether the inital character segments and the local maxima/minima on the contours are spurious one or not. This is a two scan procedure. The first scan is used to estimate the upper bound *maxwidth* and the second one is used to estimate the stroke width with the threshold *maxwidth*. For the binary image, where the character is represented by black pixels and the background by white pixels, the algorithm can be summarized as follows;

STROKE WIDTH ESTIMATION

1. For each column, count the number of black pixels and the transitions between black and white pixels i.e., the number of runs.
2. Estimate *maxwidth* by
$$maxwidth = \frac{no.of black pixels}{total no.of runs}$$
3. Repeat step 1 after discarding those pixels, whose run-length is greater than *maxwidth*.
4. The estimated width is the count of black pixels divided by the count of runs.

In order to estimate stroke height, similar algorithm is used. However, scanning procedure is applied in vertical direction, *minheight* is estimated instead of *maxwidth* and in the third step, those pixels whose run-length is smaller than *minheight* is discarded. Estimated stroke height is, then, used in slant angle detection.

5.2.2 Reference Line Finding

First, the baseline for each RU is determined by scanning the binary RU from top to bottom horizontally and counting the number of contour pixels, which overlap each scan line. The *baseline* is assumed to be the horizontal line (with respect to the RU), which contains the highest number of contour pixels (Figure 5.1).

The *lower baseline* is found by using the contours below the baseline. First, the outer contours are followed and the local minima whose increasing and decreasing are longer than the estimated stroke width are determined. The local minima corresponding to the descending characters are eliminated by using a simple heuristic. Then, the minimum of the remainder local minima is taken as the lower baseline. The algorithm used for lower baseline estimation consists of the following steps:

LOWER BASELINE ESTIMATION

1. Reject the part of the image above baseline.
2. Follow the outer contours below baseline and find the local minima whose increasing and decreasing are longer than the estimated stroke width.
3. Calculate the average distance of the minima from the baseline.
4. Eliminate the minima whose distance from the baseline, are longer than the two average distance.
5. Find the minimum of the remainder local minima as lower baseline.

Figure 5.1: Baseline and Lower baseline extraction

5.2.3 Slant Angle Estimation

The slant is the deviation of the strokes from the vertical direction, depending on the writing styles. The system makes use of the slant angle in the segmentation stage. The slant of a word is estimated by finding the average angle of near-vertical strokes. This is calculated by extracting a chain of connected pixels representing the edges of strokes whose lengths are longer than the estimated stroke height. The mode orientation of those edges close to the vertical is used as an overall slant estimate (Figure 5.2). The algorithm can be written as follows;

SLANT ANGLE DETECTION

1. Remove the horizontal lines in the writing.
2. Eliminate the lines whose lengths are less than estimated stroke height.
3. Find the lenght and slope angle of the remainder vertical lines.
4. Take the weighted average slope angle of all the lines.

(a) Writing with removed horizontal lines (b) Slant is indicated by straight lines

Figure 5.2: Slant angle detection

5.3 Segmentation

The segmentation method proposed in this study is motivated from the work of Lee et al. [156], where the character segmentation problem is defined as the problem of finding the shortest path in a graph defined over a segmentation region, minimizing the

accumulated intensity. In [8], the segmentation regions are identified from the peaks of the horizontal projection profile in the gray level image assuming machine printed characters, scanned without skew. On the other hand, our aim is to develop a system for the handwritten characters. For this reason, the proposed method performs the segmentation by combining the best use of characteristics of gray scale and binary images. First, initial segmentation regions are determined in the contour- extracted RU. Then, an improved search process is applied to the initial segmentation regions on the gray scale RU for determining the imperfect segmentation boundaries between characters.

5.3.1 Determination of Initial Character Segmentation Regions

Each RU is decomposed into three horizontal regions:

1. *Upper Portion :* The region between the local maxima above the baseline and the top of the RU.

2. *Middle Portion :* The region between the local maxima and the lower baseline.

3. *Lower Portion :* The region between the lower baseline and the bottom of the RU.

For each RU, the local maxima of the outer contour above the baseline are identified. In order to avoid the spurious maxima, the height of the local maxima and distance between the adjacent local maxima are restricted to be greater than the average stroke width. The same process is applied to the outer contours below baseline for identifing the local minima. Local minima are used to define lower baseline (as we discussed in section 5.2.2) and to decide whether there exists descender character or not.

Two rules are applied on the binary RU for identifying the initial segmentation regions:

> *Rule 1:* A single maximum is an indication of a single character or a portion of a character in a RU.

> *Rule 2:* When a RU has more than one local maxima, the rectangular region between the adjacent local maxima carries a potential segmentation point.

Although, the above rules are valid for most of the cases, they may yield wrong segmentation points in the search algorithm mentioned below. The error is corrected in the recognition based segmentation module.

Determination of character segmentation regions in each RU is accomplished in three stages:

- Segment the upper portion starting from the local maxima and proceeding to the top of the RU,
- Segment the middle portion starting from the local maxima and proceeding to lower baseline of the RU,
- Segment the lower portion starting from the lower baseline proceeding to the bottom of the RU.

In the first stage, a straight line in the slant angle direction is tried to be drawn from local maxima to the top. However, there may be some parts of the ascender character in this path. In order to avoid a cut in the ascender character, this path is forced to pass from the end point in any direction of the ascender character. While going upward, if any contour pixel is met, this contour is followed until the direction is changed to the opposite. In order to decide in which direction the contour following is done, the order of maxima pixel and contour pixel is compared. During the identification of the local maxima, if that local maximum is reached before the contour, then the contour is followed in the left direction, else the contour following goes in the right direction. After finding the end point, it is drawn a line from maxima to the end, and path continues to go upward until the top of the image (Figure 5.3).

The second stage draws a straight path in the slant direction from maxima until lower baseline (Figure 5.4).

Lastly, the same process as in the first stage is performed in order to determine the path from lower baseline to the bottom of the RU. In this case, our aim is to find the path which does not cut any part of the descender character (Figure 5.5).

Combining the three stages, the character segmentation regions from the top to the bottom of a RU are obtained (Figure 5.6). Then, the segmentation points which represent the character boundaries in each region are searched by using the non-linear

Figure 5.3: Character segmentation regions in the upper portion

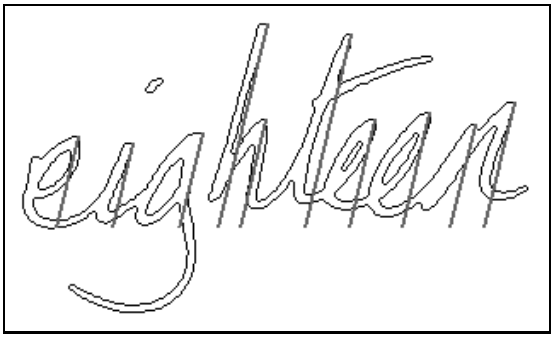


Figure 5.4: Character segmentation regions in the middle portion

character segmentation path search algorithm, which will be presented in the following section.

5.3.2 Search for Nonlinear Character Segmentation Paths

Given the segmentation regions in each RU, a search algorithm finds the shortest path from the top to the bottom of a RU, which separates the connected characters in the segmentation regions. The algorithm performs the segmentation on the gray scale RU by using both the gray scale and binary information.

For a given segmentation region, let, $G(V, A)$ be a multistage di-graph, where the vertices are partitioned into $H(Height)$ disjoint sets , $V_i, 1 \leq i \leq H$ representing the pixel coordinates of the image matrix and the arcs indicates the cost of a path between the vertices u and v $u \in V_i$ and $v \in V_{i+1}$ for some i, $1 \leq i < H$. There are N_i vertices $V_i(j)$, $1 \leq j \leq N_i$ associated with stage i. N_i differs from stage to stage according to the distance between two succesive region borders. The edges in G are of the form

Figure 5.5: Character segmentation regions in the lower portion

Figure 5.6: The character segmentation regions

$< V_i(j),\ V_{i+1}(l) >$ for all $j-2 \leq l \leq j+2$ and $1 \leq i < H$. Then, the segmentation region can be represented by a multistage di-graph as shown in Figure 5.7.

The cost of an edge $< V_i(j),\ V_{i+1}(l) >$ is determined by using the information extracted from grey level and binary image. The grey level image is used as it is. However, the boundary extracted image is processed in order to find potential segmentation points. For this purpose, the outer contours below the baseline is followed and local maxima are found. Then, a contour matrix C is constructed by assigning the black pixel value to all the pixels except these maxima.

Thus, the cost of an arc is defined as follows

$$a_{(ij)(i+1,l)} = I_{i+1,l} + \mid I_{i,j} - I_{i+1,l} \mid + i + C_{i+1,l}, \quad 1 \leq i < H, \quad and \quad j-2 \leq l \leq j+2$$

where I and C are the grey level and contour matrices respectively and i is the y coordinate of the pixel.

The goal is to find the minimum cost from the first stage to the last stage. Therefore, the problem of segmentation can be represented as finding the shortest path which min-

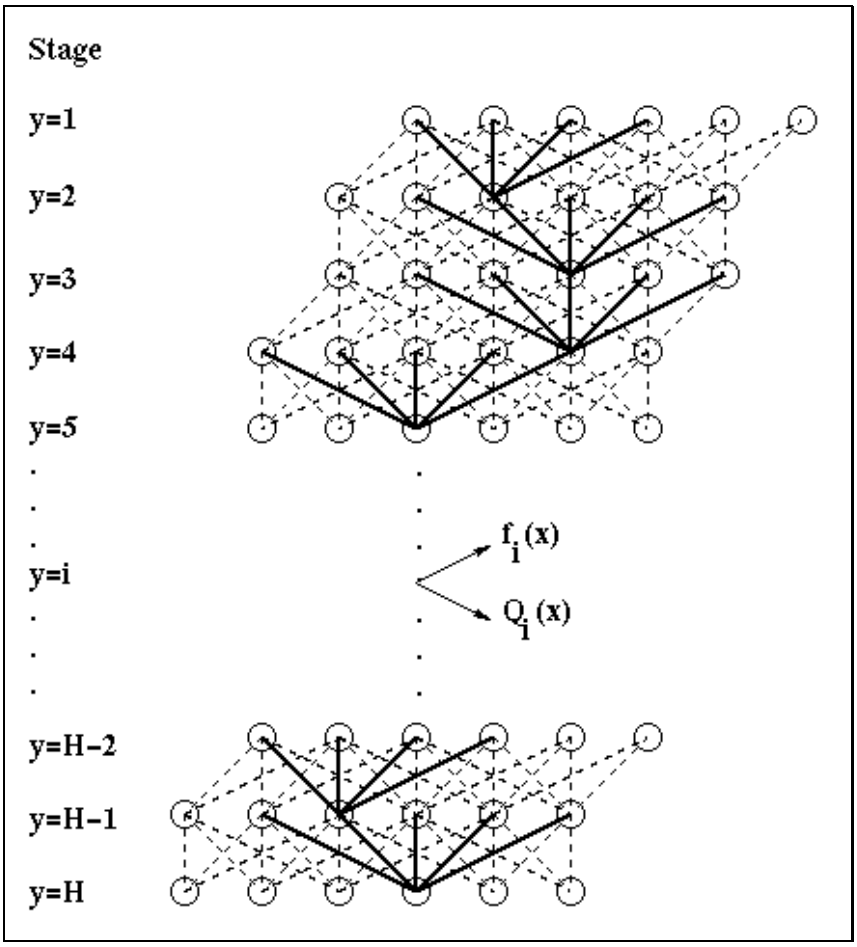


Figure 5.7: Graph representation of segmentation region

imizes the cumulative cost function, $COST = \sum_{1 \leq i < H} a_{ij,i+1l}$. Intuitively, this path passes through the white pixels and the potential segmentation points. Additionally, by adding the y coordinate to the cost function, the path is forced to cut a stroke closest to the bottom. Note that we constrain the possible vertices linked into (v_{ij}) to $(i-2, j+1)$, $(i-2, j+1)$, $(i, j+1)$, $(i+1, j+1)$ *and* $(i+2, j+1)$. This is important, because we don't want to cut any stroke horizontally. A dynamic programming algorithm searches for the shortest path (segmentation path) from top to bottom stages.

Let $f_y(x)$ be the minimum accumulated distance in stage y, and $Q_y(x)$ be a shortest path from vertices of $y-1$ stage to a vertex of y stage. We define the path with the minimum accumulated cost as the nonlinear character segmentation path. The can be searched by the following algorithm.

SEGMENTATION PATH SEARCH ALGORITHM

- *Initialization:* for $1 \leq x \leq N_1$

$$f_1(x) = I(x, 1),$$

$$Q_1(x) = x.$$

- *Recursion:* for $2 \leq y \leq H$

$$f_y(x) = \min_{x-2 \leq j \leq x+2} \{a_{(y-1,j)(y,x)} + f_{y-1}(j)\}, \quad 1 \leq x \leq N_y$$

$$Q_y(x) = \arg \min_{x-2 \leq j \leq x+2} \{a_{(y-1,j)(y,x)} + f_{y-1}(j)\}, \quad 1 \leq x \leq N_y$$

- *Termination:*

$$f^* = \min_{1 \leq x \leq N_H} \{f_h(x)\},$$

$$m_H^* = \arg \min_{1 \leq x \leq N_H} \{f_h(x)\}.$$

- *Backtracking:* for $y = H-1, H-2, ..., 2$

$$m_y^* = Q_{y+1}(m_{y+1}^*).$$

In the first step, $f_1(x)$ and $Q_1(x)$ are initialized with the intensity of pixel (1,x) and the column in the first raw, respectively. Then, the accumulated distance $f_y(x)$ can be recursively evaluated at each stage. In the final stage $y = H$, we have N_H accumulated distances, $f_H(x), x = 1, 2, ..., N_H$. The minimum accumulated distance f^* of these distances is the candidate for the shortest path. The final task, now, is to backtrack from m_H^* to the initial vertex following Q_y. It is not difficult to see that the complexity of this algorithm is $O(N + H)$, where N is the number of vertices $\sum_i^H N_i$, and H is the number of stages in the graph.

5.4 Feature Set

The characters, extracted from the segmentation algorithm of the previous section are sent to the HMM based recognizer, which was described in Chapter 3. In the recognition stage, the probability of observing the coded segment by every HMM is

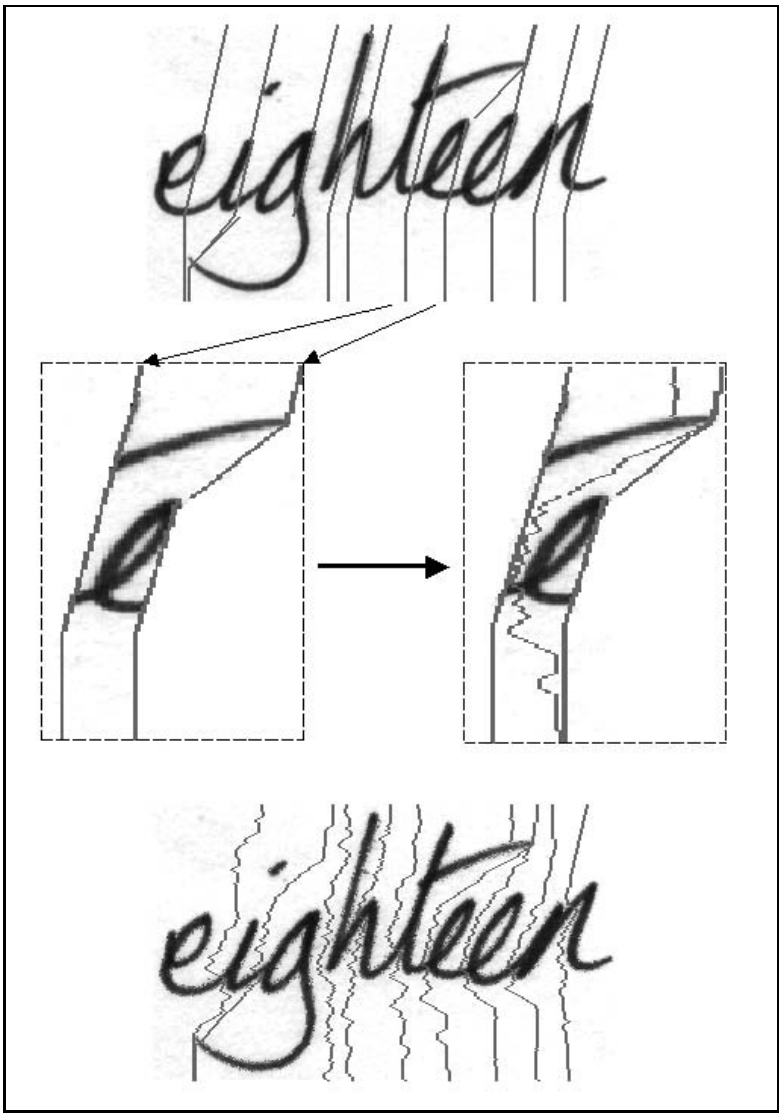


Figure 5.8: Segmentation process

calculated. Then, the observed string is labeled with the class, which maximizes the probability $P(O \mid \lambda)$.

If the preliminary segmentation algorithm yields a false partition, then the HMM classifier is forced to make an incorrect assignment for that particular segment. However, this assignment, mostly, has a low observation probability. This is due to the fact that the training set does not contain a similar sample.

5.5 Recognition Based Segmentation

Candidate character segmentation paths, which are found in the segmentation module, are not always the correct character boundaries. In order to confirm the nonlinear character segmentation paths and recognition results, a recognition based segmentation scheme is developed. The candidate character segmentation paths and the probabilities of the observation sequences $P(O \mid \lambda)$ of the HMM are used to resolve the recognition results. For this purpose, $\log P(O \mid \lambda)$ is computed for each character, recognized in the HMM classifier. Then, the HMM recognition results, and $\log P(O \mid \lambda)$ measure are represented as arcs and segmentation points are represented as nodes, in the graph (Figure 5.9). The optimal nonlinear character segmentation paths can be confirmed by maximizing the cumulative $\log P(O \mid \lambda)$ measure, searching the longest path in the graph. In this study, we assume that each segment contains at most one character and each character can be segmented into at most three segment.

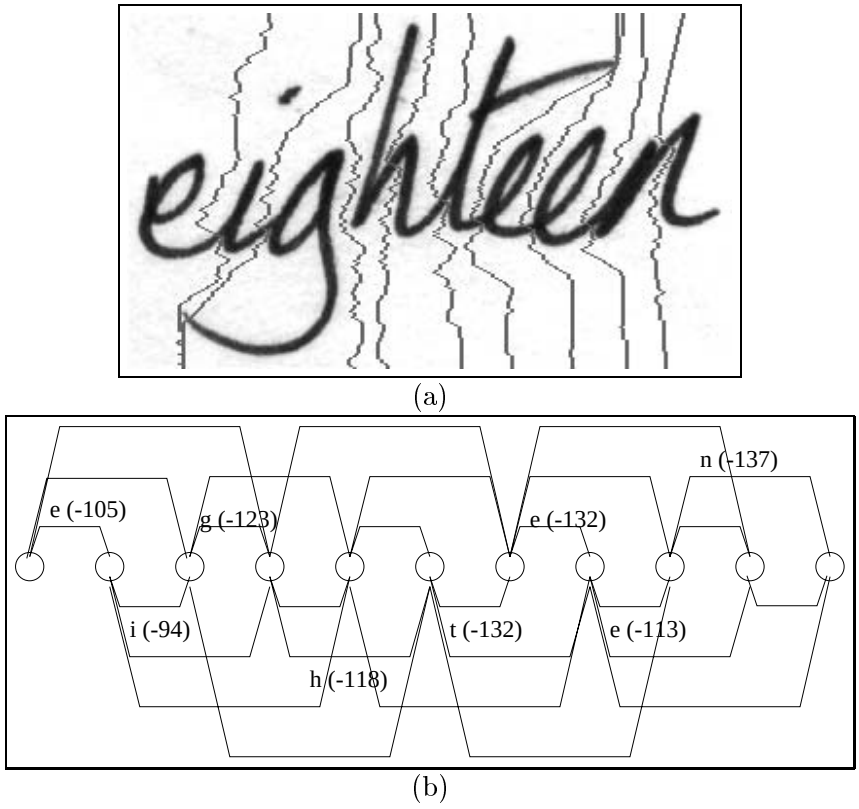


Figure 5.9: Recognition Based Segmentation a) Segmentation Paths b)Segmentation Graph

The set of possible paths in the graph can be represented as follows:

$$P = \{p_k \mid k = 1, ..., M\},$$

where p_k is a path and M is the number of possible paths in the segmentation graph. Let d_{ij} be the distance from node i to node j, $i < j \leq i+3$ and $E(p_k)$ be a set of edges on the path p_k. Then the total distance of a path p_k can be determined as follows:

$$D(p_k) = \sum_{<i,j> \in E(p_k)} d_{ij} \quad ,$$

where $< i, j >$ is an edge in $E(p_k)$.

The optimal nonlinear character segmentation paths and recognition results can be confirmed by searching the shortest path, p_s which maximizes $D(p_k)$ in the segmentation graph is

$$p_s = \arg\max_{p_k} D(p_k)$$

Each node on the shortest path represents a character boundary. The recognition results on the edge of the shortest path become the final recognition results.

CHAPTER 6

PERFORMANCE EVALUATION

This chapter gives the performance evaluation of both the isolated character recognition and cursive script recognition systems proposed in chapters 3 and 5. The systems are tested on various data sets. In isolated character recognition, National Institute of Standarts and Technology (NIST) Special Database 19 (Handprinted Forms and Characters Database) is used. In cursive handwriting recognition, a local database generated in our laboratory and a public domain database received through Internet, generated by [152] are used.

6.1 Test Results On Isolated Character Recogntion System

The isolated handwritten character recognition scheme can be examined into two categories;

1. handwritten digit recognition,

2. handwritten character recognition.

For each category, NIST Special Database 19 (Handprinted Forms and Characters Database) is used. The CD ROM of Special Database 19 (SD19) contains 3699 scanned full page binary images of Handwriting Sample Forms. There are total of 814 255 segmented handprinted digits and letters extracted from those forms. The segmented characters occupy a 128x128 pixel raster and are labelled by one of 62 classes corresponding to "0"-"9", "A"-"Z" and "a"-"z". In this database, the characters have been manually checked and it is observed that there is about 0.1% misclassified characters.

Table 6.1: NIST Database Description

Database	No. of writers	digits	lowers	writer origin
SD3	2100	223594	44593	Census Field
SD7	500	58646	12000	High School
SD19	1000	121273	23783	Census Bureau

The Special Database 19 represents NIST's most comprehensive and probably final release of class referenced images for handprinted optical character recognition. It contains all the data of Special Database 3 (SD3) and 7 (SD7) which it supersedes. The writers of the SD3 are Census Bureau field personnel stationed throughout the United States. SD7 is obtained from 500 forms completed by high school students in Maryland, USA. SD19 is collected from Census Bureau employees in Maryland, USA.

In our experiments, we used the digits and lower case letters of the NIST special databases described above. Table 6.1 shows the detailed information for all the test data of NIST. Experiments are also performed on the extracted digits and characters of our local database and Senior's database [152]. Correctly segmented digits and characters of these databases is also implicitly used to test our isolated character recognition scheme. Because the isolated caharacter recognition scheme is embeded in the cursive handwritten recognition system.

As we mentioned in chapter 3, the normalization parameters of the feature space are estimated in the training stage of HMM. These parameters are;

- the window size,
- the number of scanning directions,
- the number of regions in each scanning direction.

The training samples for each data set, namely SD3, SD7 and SD19, are randomly selected for estimating the above parameters. This is the worst case situation. An intelligent selection of the data set could definitely increase the recognition rates reported in this study. The parameters are gradually increased and the corresponding recognition rates are calculated in the training data. The parameters, which correspond to the maximum recognition rate are fixed in the next stage of training and recognition.

- *The Window Size* : The smallest window size is taken as 10x10 and increased until 30x60. The increaments are also applied to the length to width ratios of the

windows. At each width size, length-to-width ratio is started from 1 and gradually increased to 2. During the experiments, it is observed that the recognition rate depends on the window size as well as the length-to-width ratio. The optimal window size (the total number of pixels) depends on the size of the writing. The bigger the characters of the individual data set, the larger the optimal window size. On the other hand, the optimal length-to-width ratio is different for the digits and lower case character sets. This parameter depends on the deviation of the character's ascending and descending parts from the main body.

- *The Number of Scanning Directions*: In our experiments, we take vertical and horizontal scanning directions first. Then, the left and right diagonals are added to the vertical and horizontal directions. In our local database, the recognition rates were above 90%, by using only two scanning directions. However, the recognition rates on NIST database were around 80%. This result indicates that for a clearly written character, two directions are sufficient. But, more directions are required as the number of writers is increased and the number of samples gets larger. On the other hand, for four scanning directions the local database and NIST database results are above 95%. These results will be represented in the next section.

  Increasing the number of scanning directions to 6, 8 or 10 may yield slight improvements in the recognition rates. However, considering the extra complexity added to the system and the satisfactory results obtained by 4 scanning directions, we did not repeat the experiments for higher number of scanning directions.

- *The Number of Regions at each Scanning Directions:* The experiments are performed for $n = 4, 5, 6, 7$ regions for each scanning directions. It is observed that in our local database $n = 4$ regions was sufficient to obtain a recognition rate above 95%. However, the recognition rates for NIST database, increase for $n = 4, 5, 6$. For $n \geq 7$ the improvements in the recognition rates are insignificant.

The smallest number of states for HMM, is taken as 15 for the smallest window width 10 pixels. The ratio of width and state size is kept constant as 1.5 during the experiments. Therefore, the maximum number of HMM states is 45 for the maximum window width of 30.

Once the normalization parameters are estimated, the HMM's are trained by using more training samples. This is necessary in order to estimate the HMM parameters, which yield the higher recognition rates.

The results of the experiments on isolated digit and character recognition is given in two seperate section below.

6.1.1 Handwritten Digit Recognition

Handwritten digit recognition is tested on the NIST and our local databases. First the experiments are performed on NIST SD 3, 7 and 19. In each special database, 100 samples from each digit class is used for the estimation of normalization parameters.

Figure 6.1 indicates the HMM recognition rates for 4, 5, 6 regions in SD3. Note that, the recognition rate reaches to a maximum value for each window width for an optimal window length. The increasing trend of the recognition continues as we increase the window width. However, the improvement of the recognition becomes insignificant as the window size gets larger. For example, in Figure 6.1.a, the recognition rate increases 0.1% for window size greater than 24x36. Similar trends are observed in the experiments of Figure 6.1.b and c for 5 and 6 regions in each scanning directions.

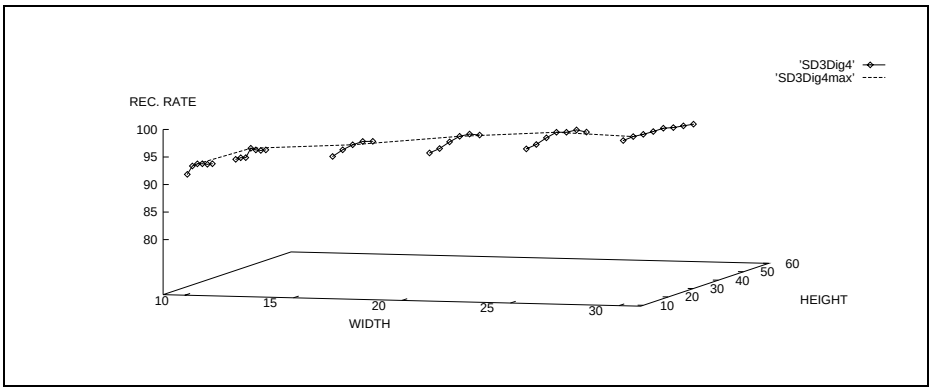
NIST Special Database 7 has more variations in the writing (collected from the students). Therefore, it is a more difficult data set. Note that the normalization parameters are close to the parameters of SD3. Similarly, the recognition rate is accomplished in 24x36 window size and 6 regions in each scanning direction (see Figure 6.2).

NIST Special Database 19 is similar to SD3 based on the writer origin. Thus, we used the same normalization parameters as in the case of SD3.

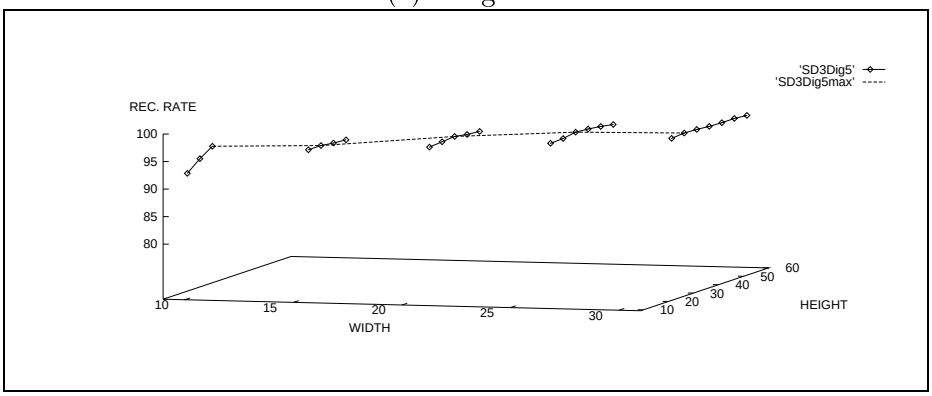
Local database tests are performed only for vertical and horizontal directions similarly with $n = 4, 5, 6$ regions. Note that increasing the number of directions has minimal contribution for increasing the recognition rates. This fact is observed in Figure 6.3 where the slope of the exponential envelope is near flat.

6.1.2 Handwritten Character Recognition

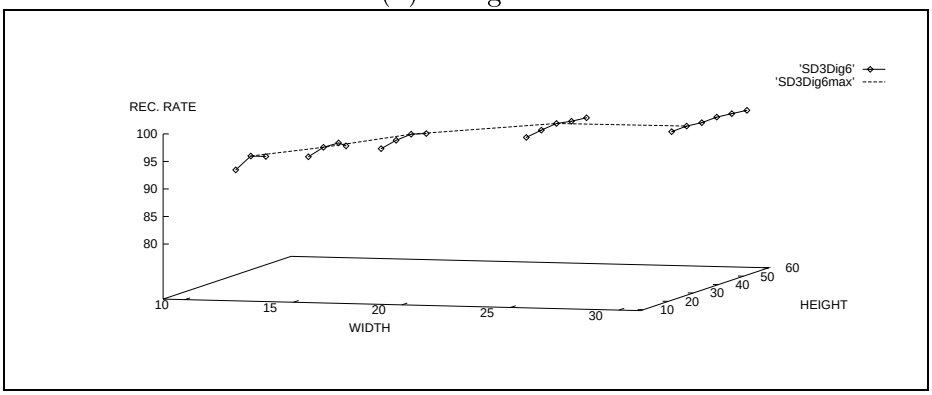
Although the nature of the problem of character recognition is the same as that of the number digit recognition, the test on both cases require data specific care. The



(a) 4 regions



(b) 5 Regions



(c) 6 regions

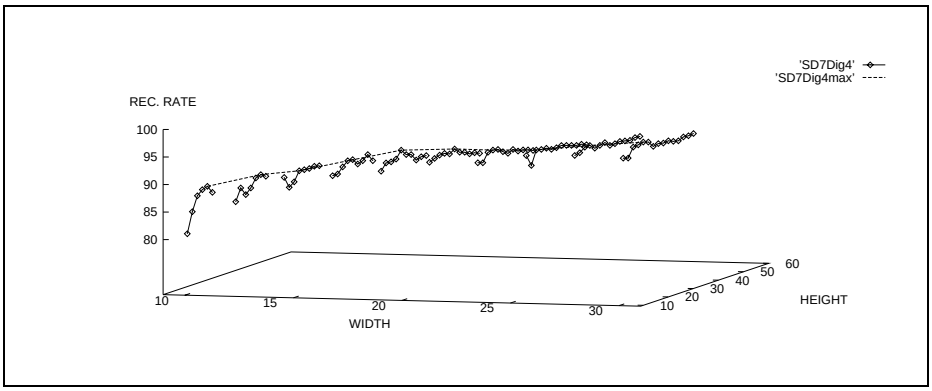
Figure 6.1: SD 3 Digit Training Results

Table 6.2: Maximum Recognition Rates in SD 3 Digit Training

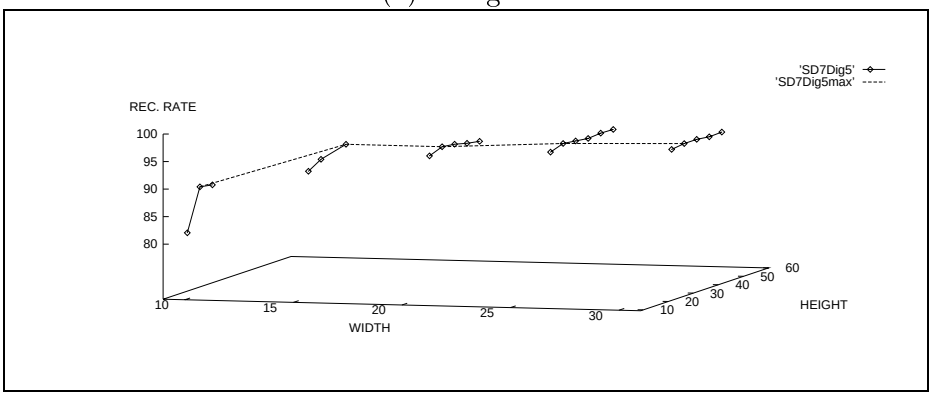
Number of Region	Window Size	Length to Width Ratio	Rec. Rate
4	10x14	1.4	93.1
4	12x18	1.5	95.5
4	16x24	1.5	95.6
4	24x36	1.5	96.8
4	28x32	1.15	97.0
5	10x20	2	96.2
5	15x20	1.33	96.8
5	20x30	1.5	97.4
5	25x35	1.4	97.9
5	30x35	1.17	98.2
6	12x18	1.5	94.9
6	15x21	1.4	96.3
6	18x30	1.6	97.6
6	24x36	1.5	99.2
6	30x36	1.2	99.3

Table 6.3: Maximum Recognition Rates in SD 7 Digit Training

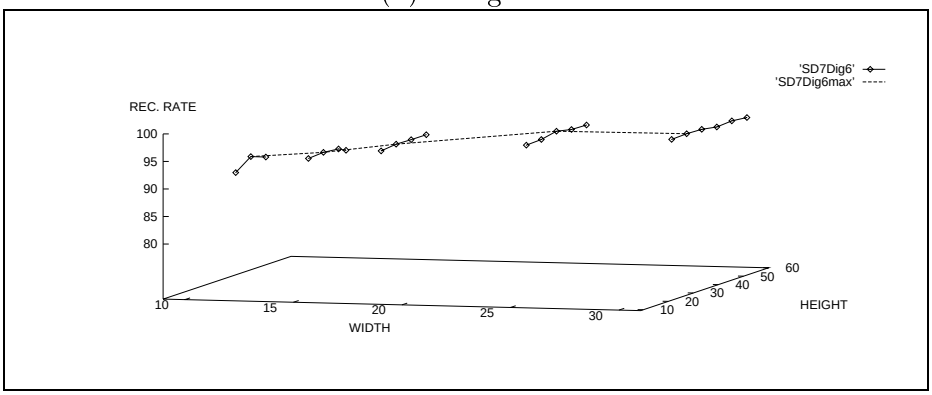
Number of Region	Window Size	Length to Width Ratio	Rec. Rate
4	10x18	1.8	88.4
4	12x22	1.83	90.1
4	14x20	1.43	91.3
4	16x22	1.38	93.0
4	18x26	1.44	94.5
4	20x30	1.5	94.5
4	22x28	1.3	94.6
4	24x28	1.18	94.8
4	26x32	1.23	95.2
4	28x36	1.28	95.4
5	10x15	1.5	89.6
5	15x30	2	95.5
5	20x25	1.25	96.3
5	25x30	1.2	96.6
5	30x35	1.17	96.3
6	12x18	1.5	94.8
6	15x21	1.4	95.4
6	18x24	1.33	96.7
6	24x36	1.5	97.8
6	30x36	1.2	97.9



(a) 4 Regions

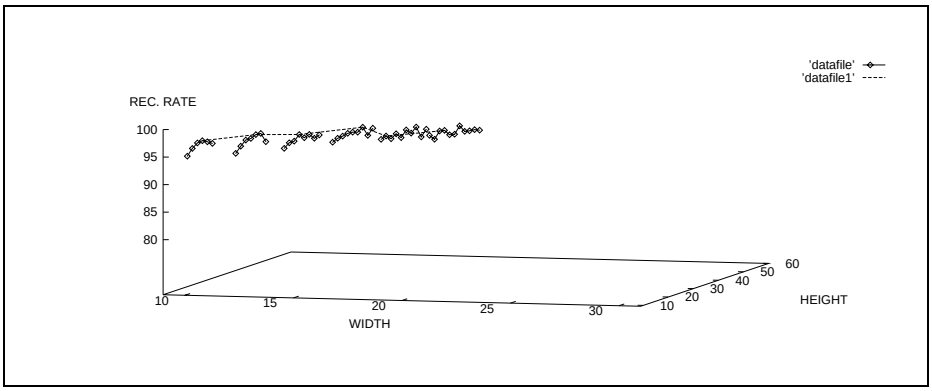


(b) 5 Regions

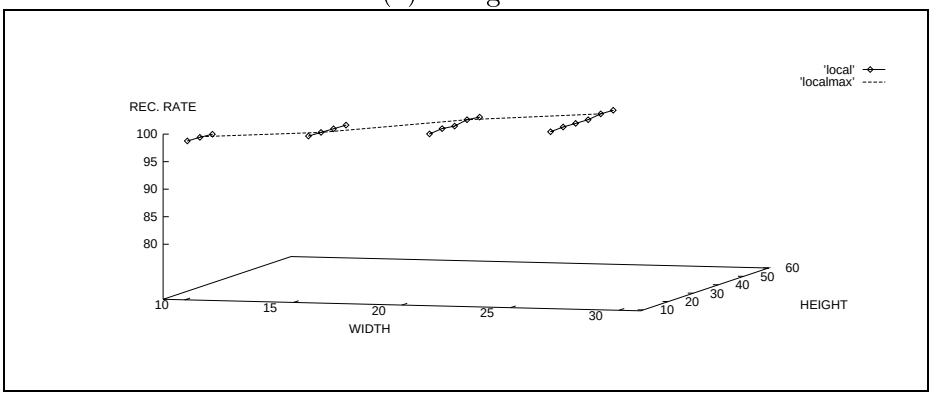


(c) 6 Regions

Figure 6.2: SD 7 Digit Training Results



(a) 4 Regions



(b) 5 Regions

Figure 6.3: Local Database Training Results

Table 6.4: Maximum Recognition Rates in Local Database Digit Training

Number of Region	**Window Size**	**Length to Width Ratio**	**Rec. Rate**
4	10x16	1.6	97.0
4	12x20	1.68	97.7
4	14x20	1.47	97.9
4	16x28	1.75	98.2
4	18x20	1.12	98.0
4	20x24	1.2	98.5
5	10x15	1.5	98.7
5	15x20	1.33	95.5
5	20x35	1.75	99.7
5	25x45	1.8	99.7

Table 6.5: Handwritten Digit Recognition Test Results

Database	No. of Samples	Win. Size	No. of Reg.	Rec. Result
SD3	200	24x36	6	95.6
SD7	500	24x36	6	95.0
SD19	200	24x36	6	95.4
Local	200	20x35	5	98.8

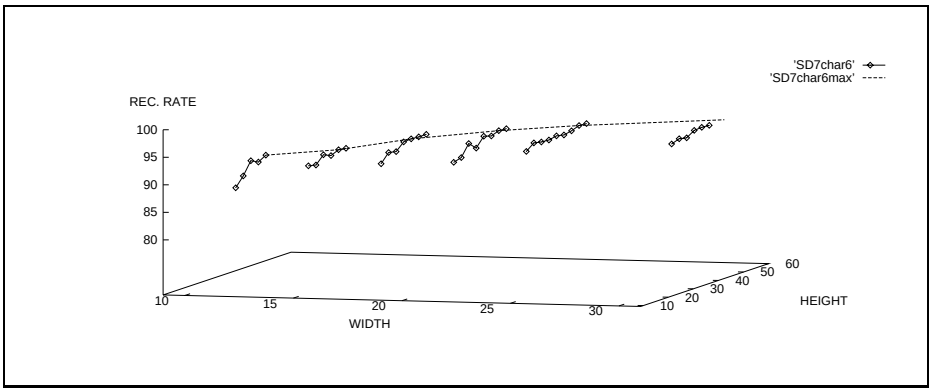


Figure 6.4: NIST SD7 Character with 6 regions

major distinction between the number digit recognition and character recognition is the ascending and descending portions of the characters. These portions do not exist in the numerals.

The existance of the ascending and descending portions affects, firstly, the length-to-width ratio of the normalization window. In fact, the optimal length-to-width ratio for character recognition is obtained as 2.0 for SD7. A close look at Figures 6.4 indicates a similar trend for optimal normalization parameters. In all of them length-to-width ratio is relatively larger than the ones for number digits. Secondly, the ascending and descending portions of the characters forms a relatively more uneven pixel distribution in the normalized window compared to the distribution of number digits. This implies a need for higher number of regions in each scanning direction.

For Senior's dataset [152], the cursive words are segmented by the algorithm explained in chapter 5. As the output of the segmentation procedure, 20 samples are taken for each character. Since the sample data set is limited, the recognition rates for different normalization parameters are close to 100%. However, we used the same normalization parameters in order to estimate the HMM parameters, as in the NIST

Table 6.6: Maximum Recognition Rates in SD 7 Character Training

Number of Region	Window Size	Length to Width Ratio	Rec. Rate
6	12x24	2	93.4
6	15x27	1.8	94.2
6	18x30	1.7	96.0
6	21x39	1.87	96.4
6	24x45	1.88	96.7
6	30x51	1.7	97.4

Table 6.7: Test Results for Character Recognition

Database	No. of Samples	Win. Size	No. of Reg.	Rec. Result
SD3	100	28x56	7	87.3
SD7	100	28x56	7	86.2
SD19	100	28x56	7	87.1
Senior's data	30	24x45	6	100

SD7 character set.

6.1.3 Comparison With Other Studies

It is not easy to compare the results obtained from the proposed method to the reported results in the literature, because of different test platforms and data sizes. A quick scan in the literature indicates a diverse variation in the recognition rates. For handwritten digit recognition, the rates vary between 85% and 99%. For handwritten character recognition, the reports gives as low as 75% as a succcesful rate. All of the methods have their own superiorities and weaknesses depending on the quality of the writing and the size of the training and recognition data.

Although, they use different training and recognition sizes of data set, we choose two very recent studies for comparing our results on NIST database. As a first comparison platform, the best 10 results reported on the NIST database is used. These results are obtained on an OCR competition in NIST Conference [157]. The tests are performed on NIST-SD7 for isolated digit recognition with unlimited computation time and training data size. The best 10 methods give recognition rates between 95.9% and 96.8%. The results are obtained by using huge training set (more than 50 000 samples) gethered from inside of NIST database as well as from the outside sources. Due to the limitations on gathering the outside data in our laboratory environment, we were restricted to use

at most 500 NIST samples for training. The achieved recognition rate for NIST-SD7 is 95.0%. The result is very satisfactory considering the robustness of our approach and 100 times less data for training set.

Note also that, there is no effort spent to construct an appropriate training data in our experiments. This arbitrary selection of the training data corresponds to the worst case situation.

Secondly, the results obtained from the proposed isolated character recognition scheme is compared to the results of the study presented by [37], which is accomplished in Almaden Research Center of IBM. In [37], sets of 1000 and 2000 images were used during training. On the other hand, only 100 samples were selected from the NIST database as test data. Additionally, the authers do not report which special database is used. The method, which uses single state HMM obtains 88.2% recognition rate whereas the method which uses multi state HMM obtains 79.1% recognition rate both for a training set of 1000 samples. On the other hand, our result gives 86-87% recognition rate on various NIST database by using only 100 samples.

The above discussion indicates that the proposed OCR scheme of this study is very promising for the use of professional OCR systems with very high recognition rates as long as it is trained on professional data sets.

6.2 Test On Handwritten Cursive Script Recognition System

Segmentation is the most crucial part of the cursive handwritten recognition problem with analytic approach. It has a great impact on the recognition rates. Small modification on the segmentation algorithms results in serious improvements in the recognition rates.

As we mentioned in chapter 5, the scheme for cursive handwritten recognition embeds the HMM recognition of isolated characters into a preliminary segmentation and recognition based segmentation algorithms. Since we present the HMM recognition test results in the previous section, we focus on the performance of the segmentation algorithms here in this section and only consider the problems of HMM due to imperfect segmentation.

The experiments are performed in two different databases;

- The local database which contains 2000 handwritten connected four digit strings,

Table 6.8: Cursive Handwriting Test Data Description

Database	No. of writers	digit string	cursive word	writer origin
Local	19	2000	-	School
Senior's dataset	1	-	240	England

- Senior's database which contains 240 handwritten words (names of numerals).

In the experiments, first the performance of the algorithm is tested. It is observed that correct segmentation depends on two factors;

- Determination of the segmentation regions,
- Search for the nonlinear segmentation paths.

Determination of segmentation regions requires slant estimation and finding the local maxima over baseline. It is observed that slant estimation contributes a great deal in finding the correct segmentation regions. In order to get rid of the spurious local maxima, a smoothing mask is used on the character boundaries. It is also observed that the smoothing has significant impact on finding the correct local maxima.

Given the correct segmentation regions, finding the false nonlinear segmentation paths is very rare. Definition of *correct sgementation* is based on the recognition. The segmentation algorithm yields correct segmentation if the final result of the recognition based segmentation algorithm is correct, provided that the HMM recognition scheme does not make any mistake. This means that the correct initial segmentation may divide a character (like u,w) into at most 3 segments.

The results on the data sets are presented in the following sections.

6.2.1 Connected Digit String Recognition

Connected digit recognition is tested on our local data set collected from 19 different persons in our department. There are total of 2000 4-digit connected string which yields $2000x4 = 8000$ numerals. The experiments indicate that there are approximately 20% over segmentation due to the digits that have more than one local maxima. However, in some cases, two digits are not seperated by a correct segmentation path. This error is carried until the final recognition based segmentation stage. In our local database, there are total of 28 such cases. Some of these errors come from the error in the

Table 6.9: Connected Digit String Test Result

Database	Err. Rate in Seg.	Err. Rate in HMM Rec.	Err. Rate in Rec.Bas.Seg.	Overall Rec.Rate
Local	1.4	1.2	6.2	91.2

Table 6.10: Cursive Script Recognition Test Result

Database	Err. Rate in Seg.	Err. Rate in HMM Rec.	Err. Rate in Rec.Bas.Seg.	Overall Rec.Rate
Senior's data	5	2	28	65

determination of segmentation regions. The others are due to the error in nonlinear segmentation path. As a result, 91.2% recognition rate is achieved in string level during the HMM recognition.

6.2.2 Cursive Word Recognition

Recognition of cursive handwriting based on the letters is the most difficult problem of the OCR field. Unless the context is not incorporated, the reported results are as low as 30-50%, depending on the writers style. In order to evaluate the performance of our cursive word recognition system, we used the data set generated in [152]. This data contains totally connected 240 cursive words for the names of numerals that apear on the bank checks. It is clear that the data set is quite restrictive and there are not enough samples for training. However, it provides us a comparison platform for our system to the system proposed by [152].

During the experiments, the training data is manually extracted from the database by using the segmentation algorithm. 20 samples for each class for total of 19 distinct characters are extracted and used for HMM training. Then, the cursive words are segmented by the proposed initial segmentation algorithm. The segmentation is correct 95%. Then, the potential characters are recognized by HMM algorithm with an accuracy of 98%. This result is much more superior than the result reported by Senior, et.al., which extensively use lexicon information and achieve 87% recognition rate.

CHAPTER 7

SUMMARY AND CONCLUSION

In this chapter, we first summarize the study in the thesis. Then, the concluding remarks are given and the future research directions are discussed.

7.1 Summary

Free style handwriting recognition is a difficult problem, not only because of the great amount of variations in human handwriting, but also, because of the overlapping and interconnection of the neighboring characters. Furthermore, when observed in isolation, characters are often ambiguous and require context to minimize the classification error. Thus, current research aims at developing systems for limited domain applications. A well-defined lexicon plus a well-constrained syntax help provide a feasible solution to the problem.

Recognition strategies heavily depends on the nature of the data to be recognized. In the cursive case, the problem is made complex by the fact that the writing is fundamentally ambiguous as the letters in the word are generally linked together, poorly written and may even be missing. On the contrary, hand printed word recognition is more related to printed word recognition, the individual letters composing the word being usually much easier to isolate and to identify. As a consequence of this, methods working on a letter basis are well suited to hand printed word recognition while cursive scripts require more specific and/or sophisticated techniques. Inherent ambiguity must then be compensated by the use of contextual information.

In this thesis, we developed two complete schemes for isolated character recognition

and cursive script recognition, respectively.

In the first scheme, we introduced a new set of one-dimensional discrete, constant length features to represent two dimensional shape information for HMM based handwritten character recognition. For this purpose, a set of directional sparse skeletons of the binarized character shapes are extracted by scanning the image grid in various directions. The skeletons of the binary image is computed in each direction. Then, the directional skeletons are appended one after the other. Finally, the coordinates of the skeleton pixels are coded by the assigned code of the regions where the sparse skeleton pixels lie. This feature set embeds the two dimensional information into a sequence of one-dimensional codes. It provides a consistent normalization among distinct classes of shapes, which is very convenient for HMM based shape recognition schemes. The normalization process consists of some parameters which are dynamically estimated in the training stage of HMM.

Although the theory of HMM is well established, the implementation involves a large number of assumptions and restrictive conditions. Therefore, the recognition results are very much dependent on the implementation issues. A number of properties are investigated about HMM based handwriting recognition, including the effects of different methods of initial parameter estimation, the relationship between the number of trainable parameters and the size of the training set, the effects of varying the model size and topology. These implementation issues were also explained.

In the cursive script case, we developed a complete scheme which uses a sequence of segmentation and recognition algorithms. One of the major contributions of this study is the dynamic utilization of the gray scale and binary information in a mixed way to extract the maximum amount of information for both segmentation and recognition stages. First, a new segmentation method performs the segmentation of the whole word into strokes which corresponds mostly to a character or a portion of a character. In the proposed method, the initial segmentation regions are determined in the boundary-extracted image. In each segmentation region, a multistage di-graph is constructed by using gray scale and binary image. A search algorithm finds the shortest path from top to bottom, which corresponds the imperfect character boundaries. Then, each segment is recognized by HMM with an output of classification probability. Finally, in order to confirm the preliminary segmentation and HMM recognition results, a recognition based segmentation method is presented.

Both the isolated character recognition and the cursive script recognition systems are tested on various national and international databases. First of all, in order to test the isolated character recognition scheme, National Institute of Standards and Technology (NIST) is used. The second database used for this scheme is the locally generated handwritten digits. The digits are extracted from the connected digits collected from 19 persons. For each data set, the normalization parameters are estimated by using limited number of samples. The parameters, which correspond to the maximum recognition rate are fixed in the next stage of training and recognition. Once the normalization parameters are estimated, the HMM's are trained by using more training samples in order to estimate the HMM parameters. The proposed method yields more superior results than many reported ones in the literature.

The cursive handwriting recognition system is tested on the data set generated in [152] which contains 300 handwritten words. In addition, locally genereted 2000 connected digits are used for testing this system. In the experiments, it is observed that correct segmentation depends on the determination of the segmentation regions and the nonlinear segmentation paths. Segmentation error in both of the data sets is as low as 5%. However, in recognition based segmentation algorithm, the error rate increases to 6.2 % in connected digits and 28 % in cursive script. This is due to the connected two characters which look like one character. This result can be improved by using a lexicon. Over all recognition rate is 91.2 % in connected digits and 65 % in cursive scripts.

7.2 Discussion and Future Research Direction

This study proposes a succesful handwritten recognition system, which attacts the free style handwritten recognition problem. The motivation of the study was to get rid of the weaknesses of the existing OCR systems. For this purpose, first a thourough analysis on the current literature is made. Then, a complete solution which reduces many drawbacks of the available methods is proposed.

It is well known that the success of the OCR systems heavily depend on the training data. Design of the training set, which represents all the classes with low within-class variances and high between-class variances is a very difficult task. It requires a lot of heuristics with very little control. As a result, the recognition rate of an OCR system

may be high in one data set and may get very low in another.

The OCR system proposed in this study is superior to the reported systems in the literature in the sense that there is no need to design a training set. The training data is selected arbitrarily and the optimum parameters of the OCR scheme are estimated based on this training set. In other words, the OCR system is customized with respect to the training data. Therefore, the recognition rates gets quite high even for very small sizes of the training set.

Another superiority of the proposed system is the utilization of a very robust segmentation method in the analytic recognition approach. When combined with the HMM probabilities, the proposed segmentation algorithm gives very accurate results. However, the recognition based segmentation through a graph optimization problem requires some improvements. The future studies are directed towards representing the graph optimization problem in a better way.

From the experience gained throughout this study, we conclude that the proposed system gets very close to the optimum recognition rates without the use of a lexicon. Therefore, in order to gain few percent more recognition rate in the future, all it is to be done is to incorporate lexicon information.

REFERENCES

[1] N. Arica, F. Yarman-Vural, A New Scheme for Off-line Handwritten Connected Digit Recognition, *International Conference on Pattern Recognition (ICPR)*, accepted, 1998.

[2] N. Arica, F. Yarman-Vural, One Dimensional Representation Of Two Dimensional Information For HMM Based Handwritten Recognition, *International Conference on Image Processing (ICIP)*, accepted, 1998.

[3] N. Arica, F. Yarman-Vural, Off-Line Handwritten Connected Character Recognition, *International Conference on Intelligence Processing Systems (ICIPS)*, accepted, 1998.

[4] N. Arica, F. Yarman-Vural, HMM Based Handwritten Recognition, *Proc. of ISCIS XII*, pp.260-266, 1997.

[5] M. A. Ozdil, F.Yarman-Vural, N. Arica, Optical Character Recognition Without Segmentation, *Lecture Notes on Computer Science* , vol.1311, ed. A. Del Bimbo, Springer, pp.609-615, 1997.

[6] N. Arica, F. Yarman-Vural, Sakli Markov Model ile El Yazisi Tanimada Iki Boyutlu Bilginin Tek Boyutlu Sunumu, *SIU'98*, s. 48-54, 1998.

[7] N. Arica, F. Yarman-Vural, Insan Optik Sistemine Benzetilerek Gelistirilen Bir El Yazisi Tanima Sistemi, *SIU'97*, s.810-816, 1997.

[8] Automated forms-processing software and services, IBM *Journal of Research and Development* Vol. 40, Num. 2, 1996

[9] V. K. Govindan and A. P. Shivaprasad, Character recognition- A review, *Pattern recognition J.* vol.23, no.7 pp. 671-683, 1990.

[10] R. M. Bozinovic and S. N. Srihari, Off-line Cursive Script Word Recognition,IEEE Trans. Pattern recognition and Machine Intelligence, vol.11, 1989.

[11] C. Y. Suen, C. C. Tappert and T. Wakahara, "The State of the Art in On-Line Handwriting Recognition in IEEE Trans. Pattern Anal. Machine Intel., Vol: 12, No: 8, pp: 787 - 808, 1990.

[12] C. Y. Suen, M. Berthod and S. Mori, Automatic Recognition of Handprinted Characters - The State of the Art in Proceedings of the IEEE, Vol: 68, No: 4, pp: 469 - 487, 1980.

[13] Adisleven Lev and Miriam Furst, Recognition of Handwritten Hebrew One-Stroke Letters by Learning Syntactic Representations of Symbols, *IEEE Trans. on Sys. Man and Cybernetics,* vol:19, no:5, pp.1306-1313, 1991.

[14] El-Sheikh and R. M. Guindi, Computer Recognition of Arabic Cursive Scripts" *Pattern Recognition*, vol.21, no.4, pp.293-302, 1988.

[15] Shumji Mori, Kazuhiko Yamamoto and Michio Yasuda, Research on Machine Recognition of Handprinted Characters, *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* vol.6, no.4, pp.386-404, 1984.

[16] Q. Tian, P. Zhang, T. Alexander Y. Kim. Survey: Omnifont Printed Character Recognition, Visual Communication and Image Processing 91: Image Processing, pp 260 - 268, 1991.

[17] C. C. Tappert, Adaptive on-line handwriting recognition *in Proc. 7th Int. Conf. Pattern Recognition,* Montreal, Canada, pp. 1004-1007, 1984

[18] S. N. Srihari, Ed., Computer Text Recognition and Error Correction. Silver Spring, MD: IEEE Computer Society Press, 1984

[19] J. R. Ullmann, Advances character recognition *in Applications of Pattern Recognition.* K. S. Fu, Ed., CRC Press, Inc., ch. 9, 1986.

[20] J. Mantas, An overview of character recognition methodologies, *Pattern Recognition. J.*, vol.19. no.6, pp. 425-430, 1986.

[21] Tappert, Cursive Script Recognition by Elastic Matching IBM J. RES. DEVELOP. Vol: 26, No: 6, pp: 765 - 77 1 1986.

[22] Special Issue on Hand-written Recognition, Pattern Recognition, Vol: 26, no:3, 1993.

[23] Jean R. Ward and Thedore Kuklinski, A Model for Variability Effects in Handwriting Character Recognition Systems in IEEE Trans. Sys. Man. Cybernetics., Vol: 18, No: 3, pp: 438 - 451, 1988.

[24] R.C.Gonzales, P.Wintz, Digital Image Processing Addison-Wesley Publishing Co, 1987.

[25] R. Schalkoff, Digital Image Processing and Computer Vision, Mac Graw Hill, 1990.

[26] W.K. Pratt, Digital Image Processing, Wiley InterScience,1991.

[27] J. S. Lee, Digital Image Smoothing and Sigma Filter, Computer Vision, Graphics and Image Processing, 24 pp.255-269, 1983.

[28] R. M. Haralick and L. G. Shapiro, Computer and Robot Vision, 1, Addison-Wesley Publishing, 1992.

[29] M. Y. Chen, A. Kundu and J. Zhou, Off-line Handwritten Word Recognition Using a Hidden Markov Model Type Stochastic Network, IEEE Trans. Pattern recognition and Machine Intelligence, vol.16, pp.481-496, 1994.

[30] H. Baird. Document Image Defect Models and Their Use, Proc. of Int. Conf.on Document Analysis and Recognition, pp 62-67 Japan, 1993.

[31] T. Kunango,R. Haralick, I.Phillips. Global and Local document degredation models. Proc. of Int. Conf.on Document Analysis and Recognition, Japan, 1993.

[32] W. Guerfaii and R. Plamondon, "Normalizing and Restoring On-line Handwriting", Pattern Recognition, Vol: 26, No:3, pp 418-431. 1993.

[33] A. C. Downtown, C.G.Leedham, "Preprocessing and Presorting of Envelope Images for Automatic Sorting Using OCR" Pattern Recognition, Vol:23, No:3-4, pp: 347-362, 1990.

[34] W. Postl, Detection of Linear Oblique Structures and Skew Scan in Digitized Documents, 8th International Conf. on Pattern recognition (ICPR), 1986.

[35] B. Yu and A. K. Jain, A Robust and Fast Skew Detection Algorithm for Generic Documents, Pattern Recognition, 1996.

[36] A. Hashizume, P. S. Yeh and A. Rosenfeld, A Method of Detecting the Orientation of Aligned Components, Pattern Recognition Letters, vol.4, pp.125-132, 1986.

[37] S. Connell, A Comparison of Hidden Markov Model Features for the Recognition of Cursive Handwriting , Master Thesis, Michigan State University, 1996.

[38] B. A. Yanikoglu and P. Sandon, Recognizing Off-line Cursive Handwriting CVPR'94, Seattle, 1994.

[39] O. D. Trier and A. K. Jain, Goal Directed Evaluation of Binarization Methods, IEEE Trans. Pattern recognition and Machine Intelligence, vol.17, pp.1191-1201, 1995.

[40] W. Niblack, An Introduction to Digital Image Processing, Prentice Hall, Englewood Cliffs, NJ, 1986.

[41] S.AMahmoud, I.Abuhabia, R.J.Green, Skeletonization of Arabic Characters Using Clustering Based Skeletonization Algorithm in Pattern Recognition, Vol:24, No:5, pp:453-464, 1991

[42] Plummer and E.R. Davies, "Thinning Algorithms A Critique and A New Methodology" Pattern Recognition, Vol:14, No: 1, pp:53-63, 1981.

[43] F. Yarman- Vural, A. Atici, A Segmentation and Feature Extraction Algorithm for Ottoman Cursive Script, The Third Turkish Symposium on Artificial Intelligence and Neural Networks, June 1994.

[44] L. Lam, S. W. Lee and C. Y. Suen, Thining Methodologies- A Comprehensive Survey, IEEE Trans. Pattern recognition and Machine Intelligence, vol.14, pp.869-885, 1992.

[45] A. Jain and K. Karu, Page Segmentation Using Texture Analysis, Pattern Recognition, vol. 29, pp. 743-770, 1996.

[46] Y. Tang, H. Ma, X. Mao, D. Liu and C. Suen, A New Approach to Document Analysis Based on Modified Fractal Signature, *3rd International Conference on Document Analysis and Recognition (ICDAR),* pp.567-570, Canada, 1995

[47] D. Doermann, Page Decomposition and Related Research, Proc. Symp. Document Image Understanding Technology, pp. 39-55, Bowie, 1995.

[48] D. Sylwester and S. Seth, A trainable, Single Pass Algorithm for Column Segmentation, *3rd International Conference on Document Analysis and Recognition (ICDAR),* pp. 615-618, Canada, 1995.

[49] T. Pavlidis, and J. Zhou, Page Sementation by White Streams, Proc.*1st International Conference on Document Analysis and Recognition (ICDAR),* pp.945-953, Saint-Malo, France, 1991.

[50] J. Ha, R. Haralick and I. Philips, Document Page Decomposition by Bounding-Box Projection Technique, Proc. *3rd International Conference on Document Analysis and Recognition (ICDAR),* pp. 119-122, Canada, 1995.

[51] O. Akindele and A. Belaid, Page Segmentation by Segment Tracing, Proc. *2nd International Conference on Document Analysis and Recognition (ICDAR),* pp.314-344, Japan, 1993.

[52] K. Kise, O. Yanagida and S. Takamatsu, Page Segmentation based on Thinning of Background, Proc.*13th International Conference on Pattern Recognition,* pp. 788-792, Vienna, 1996.

[53] L. O'Gorman, The Document Spectrum for Page Layout Analysis, *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)* vol.15, pp.162-173, 1993.

[54] J. Liu, Y. Tang, Q. He and C. Suen, Adaptive Document Segmentation and Geometric Relation labeling: Algorithms and Experimental Results, Proc.*13th International Conference on Pattern Recognition,* pp.763-767, Vienna, 1996.

[55] A. K. Jain, Bin Yu, Document Representation and Its Application to Page decomposition, *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)* vol.20, pp.294-308, 1998.

[56] R. G. Casey and E. Lecolinet, "Strategies in Character Segmentation: A Survey", *3rd International Conference on Document Analysis and Recognition (ICDAR),* pp.1028-1033, 1995.

[57] R. B. Hennis, The IBM 1975 Optical Page Reader, system design, *IBM Journal of Research nad Development* pp.346-353, 1968.

[58] S. Tsujimoto and H. Asada, Major Components of a complete text reading system, Proc. IEEE, vol.80, no.7 pp.1133, 1992.

[59] J. Wang and J. Jean, Segmentation of merged characters by neural networks and shortes path, Pattern Recognition, voool.27, no.5, pp.649, 1994.

[60] L. D. Harmon, Automatic Recognition of Print and Scripts, Proc of IEEE vol.60, no.10, pp.1165-1177, 1972.

[61] E. Lecolinet and J-P. Crettez, A Grapheme Based Segmentation Technique for Cursive Script Recognition, Proc.*1st International Conference on Document Analysis and Recognition (ICDAR),* pp.740, Saint-Malo, France, 1991.

[62] V. A. Kovalevski, "Character Readers and Pattern Recognition", Spartan Books, Washington DC, 1968.

[63] R. G. Casey and G. Nagy, " Recursive Segmentation and Classification of Composite Patterns, Proc. 6th ICPR, pp.1023, 1982.

[64] C. J. C. Burges, J. I. Be and C. R. Nohl, Recognition of Handwritten Cursive Postal Words using Neural Networks" Proc. USPS 5th Adv. Technology Conf., pp.117, 1992.

[65] M. Mohamed and P. Gader, "Hanwritten Word Recognition Using Segmentation-Free Hidden Markov Modeling and Segmentation Based Dynamic Programming Techniques" *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)* vol.18, no.5, pp.548-554, 1996.

[66] K.Fukushima and T. Imagawa, "Recognition and Segmentation of connected characters with selective attention" Neural Netowrks, vol.6, no.1, pp.33-41, 1993.

[67] M. Gilloux, "Hidden Markov Models in Handwriting Recognition" Fundamentals in Handwriting Recognition, S. Impedovo (Ed.) NATO ASI Series F, vol.124, Springer Verlag, 1994.

[68] C. Chen, J. DeCurtins, "Word recognition in a segmentation-free approach to OCR" *2nd International Conference on Document Analysis and Recognition (ICDAR),* pp.573, 1993.

[69] K. C. Hayes, "Rading Handwritten Words Using Hierarchical Relaxation", Computer Graphics Image Processing, vol.14, pp.344-364, 1980.

[70] J. C. Simon, "Off-line Cursive Word Recognition", Proc. of IEEE, pp.1150, 1992.

[71] E. Lecolinet, "A New Model for Context Driven Word Recognition" Proc. SDAIR, pp.135, 1993.

[72] H. Fujisawa, Y. Nakano and K. Kurino, "Segmentation Methods for character recognition: from segmentation to document structure analysis",Proc. IEEE, vol.80, no.7, pp.1079-1092, 1992.

[73] R. M. K. Sinha et.al, "Hybrid contextual text recognition with string matching" *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)* vol.15, no.9, pp.915-925, 1993.

[74] B. Plessis, A. Sicsu, L. Heute, E. Lecolinet, O. Debon and J. V. Moreau, "A Multi-Classifier Strategy fro the Recognition of Handwritten Cursive Words", Proc. *2nd International Conference on Document Analysis and Recognition (ICDAR),* pp.642-645, 1993.

[75] M. Gilloux, J. M. Bertille and M. Leroux, "Recognition of Handwritten Words in a limited Dynamic Vocabulary" IWFHR III, Buffalo, pp. 417, 1993.

[76] R. Nag, K. H. Wong and F. Fallside, "Script Recognition Using Hidden Markov Models" IEEE ICASSP, pp.2071, 1986.

[77] S. Madhvanath and V. Govindaraju, "Holistic Lexicon Reduction", Proc. IWFHR III, Buffalo, pp.71, 1993.

[78] S. Impedovo, Ed., Fundamentals in Handwriting Recognition. *NATO- Advanced Study Institute Series* Springer-Verlag, October 1994.

[79] *Proceedings of the Third International Workshop on Frontiers in Handwriting Recognition*, Buffalo, Newyork, May 1993.

[80] R. Plamandon, Handwriting processing and recognition, *Pattern Recognition* 26(3), March 1993.

[81] W. Y. Kim and P. Yuan, A practical Pattern Recognition System for Translation, Scale and Rotation Invariance, *CVPR'94,* Seattle, Washington, June 1994.

[82] A, J, Elms, A connected character recognizer using Level Building of HMMs, *IEEE 12th IAPR International Conference on Pattern Recognition* II pp.439-441, 1994.

[83] S. S. Wang, P. C. Chen and W. G. Lin, Invariant Pattern Recognition by Moment Fourier Descriptor, *Pattern Recognition* , 27 pp. 1735-1742 1994.

[84] Y. Hamamoto, S. Uchimura, K. Masamizu and S. Tomita, Recognition of Handprinted Chinese Characters Using Gabor Features, *3rd International Conference on Document Analysis and Recognition (ICDAR),* Canada, 1995.

[85] S. W. Lee and Y. J. Kim, Multiresolutional Recognition of Handwritten Numerals with Wavelet Transform and Multilayer Cluster Neural Network, *3rd International Conference on Document Analysis and Recognition (ICDAR),* Canada, 1995.

[86] K. Fukunaga, *Introduction to Statistical Pattern Recognition,* Academic Press, Inc., 1990.

[87] K. M. Mohiuddin and J. Mao, A comparative study of different classifiers for handprinted character recognition, *Pattern Recognition* pp. 437-448, 1994.

[88] H. Takahashi, A neural net ocr using geometrical and zonal-pattern features, *1st International Conference on Document Analysis and Recognition (ICDAR),* 1991.

[89] M. K. Brown, S. Ganapathy, "Preprocessing Techniques for Cursive Script Word Recognition" in Pattern Recognition, Vol: 16, No: 5 pp: 447 - 458, 1983.

[90] Amlan Kundu, Yang He, Paramvir Bahl, "Recognition Of Handwritten Word: First and Second Order HMM Based Approach" in Pattern Recognition, Vol:22, No:3, pp: 283 - 297, 1989.

[91] Amlan Kundu, Yang He, On optimal Order in Modeling Sequence Of Letters in Words Of Common Language As a Markov Chain in Pattern Recognition, Vol:24, No:7, pp: 603 - 608, 1991.

[92] J.Mantas, "Methodologies in Pattern Recognition and Image Analysis- A Brief Survey" pattern Recognition, Vol: 20, No:1, pp:1 - 6, 1987.

[93] L.R. Rabiner. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition" *Proceedings of the IEEE*, 77, pp.257-286, 1989.

[94] L.R. Rabiner and B. H. Juang, An Introduction To Hidden Markov Models *IEEE ASSP Magazine*, pp: 4 - 16, 1986.

[95] L. E. Baum, An Inequality and Associated Maximization Technique in Statistical Estimation of Probabilistic Functions of Markov Processes, *Inequalities*, pp.1-8, 1972.

[96] S. E. Levinson, L. R. Rabiner and M. M. Sondhi, An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process to Automatic Speech Recognition, *Bell System Technical Journal* no.62, pp.1035-1074, 1983.

[97] L. R. Bahl, P. F. Brown, P. V. Desouza and R. L. Mercer, Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition, *ICASSP'86*, 1986.

[98] L. R. Bahl, P. F. Brown, P. V. Desouza and R. L. Mercer, A new Algorithm for Estimation of Hidden Markov Model Parameters, *IEEE International Conference on ASSP*, 1988.

[99] Y. Ephraim, A. Dembo and L. R. Rabiner, A minimum discrimination information approach for Hidden Markov Modelling, *ICASSP'87*, 1987.

[100] A. J. Viterbi, Error Bounds for Convolution Codes and An Asymptotically Optimal Decoding Algorithm, *IEEE Trans. on Information Theory* pp.260-269, 1967.

[101] G. D. Forney, The Viterbi Algorithm, *Proc. of IEEE*, pp.268-278, 1973.

[102] R. M. Gray, Vector Quantization, *Readings in Speech Recognition* Morgan Kaufmann Publishers, Inc., pp.75-100, 1990.

[103] A. K. Jain and R. C. Dubes, Algorithms for Clustering Data, *Prentice Hall*, Englewood Cliffs, 1988.

[104] L. R. Bahl, F. Jelinek and L. R. Mercer, A Maximum Likelihood Approach to Continuous Speech Recognition, *IEEE Trans, Pattern Analysis and Machine Intelligence* pp.179-190, 1983.

[105] F. Jelinek and R. L. Mercer, Interpolated Estimation of Markov Source Parameters From Sparse Data, *Pattern Recognition in Practice*, pp. 381-397, 1980.

[106] Amlan Kundu, Yang He, 2-D Sphere Classification Using Hidden Markov Model *IEEE Trans. on PAMI*, Vol:13, No:11, pp: 1172 - 1184, 1991.

[107] Tayli, Murat and A I. Ai-Salamah, "Building Bilingual Microcomputer System" *Comms. of the ACM* Vol.33, No:5, pp:495-504, 1990.

[108] W.H.Tsai, K.S.Fu, Attributed Grammar- A tool for Combining Syntactic and Statistical Approaches to Pattern Recognition, *IEEE Trans on System Man and Cybernetics*, Vol: 10, No:12, pp: 873-885, 1980.

[109] M. Shridhar, A. Badreldin, High Accuracy Syntactic Recognition Algorithm for Handwritten Numerals, IEEE Tran on Systems Man and Cybernetics, Vol 15, No 1, pp 152 - 158, 1985.

[110] T. Pavlidis, Recognition of Printed Text under Realistic Conditions, Pattern Recognition Letters, pp 326, 1993.

[111] Si Wei Lu, Ying Ren and Ching Y. Suen, Hierarchical Attributed Graph Representation and Recognition of Handwritten Chinese Characters in Pattern Recognition., Vol. 24, No. 7, pp: 617 - 672, 1991.

[112] A Belaid and J.P. Haton, "A Syntactic Approach for Handwritten Mathematical Formula Recognition" in IEEE. Trans. on PAMI-6, pp: 105-111, 1984.

[113] H.Bunke, ASanfeliu, "Introduction To Special Issue on Syntactic Pattern Recognition" in Pattern Recognition, Vol: 1 9, No:14, pp:249-254, 1986.

[114] E.Tanaka,M.lkeda and K.Ezure,"Direct Parsing" Pattern Recognition, Vol.119, pp 315-323. 1982.

[115] S. Impedevo, Fundamentals in Handwtiting Recognition, Proc. on NATO-ASI series, Springer Verlag. 1993.

[116] L. Schomaker, "Using Stroke or Character -based Self-organizing Maps in the Recognition of On-line, Connected Cursive Script", *Pattern Recognition*, vol.26, no.3, pp. 443-450, 1993.

[117] El Emami and Mike Usher, On-Line Recognition of Handwritten Arabic Characters in IEEE Trans on PAMI, Vol: 12, No: 7, 1990.

[118] Ching Y. Suen and Tuan A. Mai, " A Generalized Knowledge-Based System for the Recognition of Unconstrained Handwritten Numerals" in IEEE Trans. on Sys., Man and Cyber., Vol: 20, No: 4, pp: 835 - 848, 1990.

[119] T.Watanabe, L Qin, N. Sugie. Structure Recognition Methods for Various Types of Documents Machine Vision and Applications vol, no 6, pp 163 - 176 1993.

[120] J. Rocha, T. Pavlidis, A Shape Analysis Model, IEEE Tran on Pattern and Machine Intelligence, Vol. 16, No. 4, pp 394 - 404, 1994.

[121] S. Mori, C. Y. Suen, K. Yamamoto, Historical Review of OCR Research and Development, IEEE-Proceeding, Vol.80, No.7, pp 1029-1057,1992.

[122] S.L Xie, M. Suk, On Machine Recognition of Hand-printed chinese Characters by Feature Relaxation, Pattern Recognition Vol: 21, No:1, pp:1-7, 1988.

[123] F. Cheng, W. Hsu, M. Kuo, Recognition of Handprinted Chinese Characters via Stroke Relaxation, Pattern Recognition, Vol. 26, No. 4, pp 579 - 593, 1993.

[124] D. Tubbs, "A Note on Binary Template Matching" in Pattern Recognition, Vol: 22, No: 4, pp: 359 - 365, 1989.

[125] Paul Gader et al., "Recognition of Handwritten Digits Using Template and Model Matching"in Pattern Recognition, Vol.24., No:5, pp:421-431,1991.

[126] Jerome M. Kurtzberg, "Feature Analysis for Symbol Recognition by

[127] Keith E. Price, "Relaxation Matching Techniques Comparison" IEEE Trans. on Pattern Anal. Mach. Intel., Vol: PAMI-7, No: 5, pp: 617 - 623, 1985.

[128] X. Huang, J. Gu, A constrained Approach to Multifont Chinese Character Recognition, IEEE Tran on PAMII Vol 15, No 8, pp 838-843, 1993.

[129] Y.Bengio ,R. De Mori ,G.Flammia ,R.Kompe, "Global Optimization of a Neural Network - Hidden Markov Model Hybrid" IEEE Trans. on Neural Networks, Vol:3., No:2, pp: 252-259, 1992.

[130] C.G.Lau, " Neural Networks, I:Theory and Modeling " in Proceedings of the IEEE Special Issue On Neural Networks, 1990

[131] K. Fukushima, N. Wake, Handwritten Alphanumeric Character Recognition by neocognition, IEEE Neural Network,Vol: 2 No :3, pp :355-365, 1991.

[132] B. Hussain, M. Kabuka, A Novel Feature Recognition Neural Network and its Application to Character Recognition, IEEE Tran on Pattern and Machine Intelligence, Vol. 16, No. 1, 1994.

[133] L Xu, A. Krzyzak, C.Y. Suen. Methods of combining Multiple classifiers and their Application to Handwritten Recognition, IEEE Tran on Systems Man and Cybernetics, vol 22, no 3, pp418-435 1992.

[134] C. Y. Suen, R. Legault,, C. Nadal, M. Cheriet and L. Lam, "Building a New Generation of Handwriting Recognition Systems" Pattern Recognition Letters, vol.14, no.4, pp.303-315, 1993.

[135] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Publishers, Inc., California, 1988.

[136] M. D. McLeish, P. Yao and T. Stirtzinger, " A Study on the Use of Belief Functions for Medical Expert Systems", Journal of Applied Statistics, vol.18, no.1, pp.155-174, 1991.

[137] Y. S. Huangand C. Y. Suen, "A Method of Combinig Multiple Experts for the Recognition of Unconstrained Handwritten Numerals", IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI), vol.17, no.1, pp.90-94, 1995.

[138] O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis,* Wiley-Interscience, 1973.

[139] A. Kundu, Y. He and P. Bahl, Recognition of Handwritten Word: First and Second Order Hidden Markov Model Based Approach, *Pattern Recognition,* 22, pp. 283-297, 1989.

[140] P.A. Devijer and J. Kittler, *Pattern Recognition: A Statistical Approach* , Prentice Hall Edition, 1982.

[141] S. Smith, M. Borgoin, K. Sims and H. Voorhees, Handwritten Character Classification Using Nearest Neighbor in Large Databases, *IEEE Pattern Recognition Nad Machine Intelligence (PAMI),* vol.16, no.9, pp. 915-919, 1994.

[142] S. O. Belkasim, M. Shridhar and M. Ahmadi, Pattern Recognition with Moment Invariants: A comparative Survey, *Pattern Recognition*, vol:24, no.12, pp.1117-1138, 1991.

[143] F. T. Yarman-Vural, E. Ataman, Noise Histogram and Cluster Validity for Gaussian Mixtured data, *Pattern recognition* vol.20, no.4, pp.385-401, 1987.

[144] H. Derin, P. A. Kelly, Discrete Index Markov Type Random Processes, *IEEE Proceedings*, vol.77, no.10, pp.1486-1510, 1989.

[145] B. H. Juang, L. R. Rabiner, The Segmental K-means Algorithm for Estimating Parameters of Hidden Markov Models, *IEEE Transaction on A.S.S.P.,* vol.38, no.9, 1990.

[146] F. H. Cheng, W. H. Hsu and C. A. Chen, Fuzzy Approach to Solve the Recognition Problem of Handwritten Chinese Characters, *Pattern Recognition*, vol.22, no.2, pp.133-141, 1989.

[147] I. Abulhaiba, P. Ahmed, A Fuzzy Graph Theoretic Approach to Recognize the totally unconstrained handwritten numerals, *Pattern Recognition,* vol.26, no.9, pp.1335-1350, 1993.

[148] L. Wang, J. Mendel, A Fuzzy Approach to Hnadwritten Rotation invariant Character Recogntion, *Computer Magazine,* vol.3, pp.145-148, 1992.

[149] A. Atici, Fatos Yarman-Vural, A Heuristic Method for Arabic Character Recognition, *Journal of Signal Processing*, vol.62, pp.87-99, 1997.

[150] Paul Wang, Robert Cishiau, "Machine Recognition of Printed Chinese Characters Via Transformation Algorithms" in Pattern Recognition, Vol: 5, pp: 303 - 321, 1973.

[151] M.Shridhar and F.Kimura, "Handwritten Numerical Recognition Based On Multiple Algorithms" in Pattern Recognition, Vol:24, No: 1 0, pp:969-983 1991.

[152] A. W. Senior and A. J. Robinson, "An Off-Line Cursive Handwriting Recognition" *IEEE Pattern Recognition Nad Machine Intelligence (PAMI),* vol.20, No.3, pp. 309-322, 1998.

[153] G. Kim and V. Govindaraju, A Lexicon Driven Approach to Handwritten Word Recognition for Real-Time Applications, *IEEE Trans. on PAMI*, vol.19, no.4, pp.366-379, 1997.

[154] M. Mohamed, P. Gader, Handwritten Word Recognition Using Segmentation Free Markov Modeling and Segmentation Based Dynamic Programming Techniques, *IEEE Trans. on PAMI*, vol.18, no.5, pp.548-554, 1996.

[155] N. W. Strathy, C. Y. Suen, A New System for Reading Handwritten Zip Codes, *Proc. Int. Conf. on Document Analysis and Recognition* Montreal, pp.74-77, 1995.

[156] W. Lee, D.J. Lee, H. S. Park, A New Methodology for Gray Scale Character Segmentation and Recognition, *IEEE Trans. on PAMI*, vol.18, no.10, pp.1045-1050, 1996.

[157] R. A. Wilkinson, J. Geist, S. Janet, P. J. Grother, C. J. C. Burges, R. Creecy, B. Hommond, J. J. Hull, N. W. Larsen, T. P. Vogl and C. L. Wilson, The First Census Optical Character Recognition Systems Conference. *The U.S. Bureau of Census and the National Instute of Standards and Technology, Technical Report*, no. NISTIR 4912, 1992.