A KNOWLEDGE-BASED APPROACH TO RECOGNIZE UNCONSTRAINED HANDWRITTEN NUMERALS

Tuan Anh Mai

A Thesis

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements for
the Degree of Master of Computer Science at
Concordia University
Montréal, Québec, Canada

April 1989

ISBN 0-315-49094-2

ABSTRACT

A KNOWLEDGE-BASED APPROACH TO RECOGNIZE UNCONSTRAINED HANDWRITTEN NUMERALS

Tuan Anh Mai

A method to recognize unconstrained handwritten numerals using knowledge base was proposed. In the training stage, data on features observed in a training set were stored in 2 knowledge bases of features and pattern classes. In the recognition stage, an inference engine compared the features of an unknown sample with the knowledge base of features and inferred the most probable hypothesis on the identity of the sample. An alternative method used a structural classifier which matched the results of feature extraction against the knowledge base of pattern classes. Recognition decision in both methods was based on Bayes' Rule.

The system was trained and tested on real-life handwritten ZIP codes obtained from the U.S Postal Services, using a training set of 8500, and a test set of 8485 samples. It was also trained and tested with the same data used by the Concordia OCR Project Team containing 4000 training and 2000 test samples. The inference method achieved a recognition rate of 94.9%, with a reliability of 97.0%. Threshold values could be applied to the structural method to give a continuous range of recognition and reliability rates. With a very stringent threshold of confidence level, a high reliability rate of up to 99.8% can be achieved. These results compare favourably with recognition results reported by other researchers on totally unconstrained handwritten numerals, and results obtained by other members of the Concordia OCR Project Team.

The method aimed to minimize human subjectivity by not excluding any poor quality samples found in the training set. It can be retrained with different training sets of numerals and modify its knowledge base accordingly. It was also found that incremental training on more samples would improve the recognition rate.

ACKNOWLEDGEMENTS

I am indebted to my thesis supervisor Professor Ching Y. Suen for his guidance, suggestions and encouragement in the course of this work. His concern in the progress of project and his valuable help has been strong motivation for my work.

I would like to thank my supervisors at work for their understanding and support during the period I studied part-time for the degree of Master of Computer Science, especially Mr. Terry Too, Mr. Peter Paquet and Dr. Richard Dimond.

I would like to thank the U.S Postal Services for providing the data and many previous and current researchers in the OCR project at Concordia who have segmented, selected and binarized the raw data to their present form. Most importantly, they have established a wealth of knowledge and experimental results on the subject which serves as a firm foundation for new research directions.

I would also like to thank Dr. R. Shinghal for the use of his thinning algorithm and the Operations staff of the Computer Centre of Concordia University for their cooperation.

Finally, my special thanks go to my family for their support and understanding.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

Optical Character Recognition (OCR) of handwritten characters is a very active field in Pattern Recognition. There are always a large amount of handwritten data that need to be processed in business and government organizations such as mail, bank cheques, income tax returns..,which must be read and sometimes entered into machine-readable format by human. An efficient, fast and economical OCR method to recognize handwritten characters will eliminate most of the repetitive work by human, increase the speed and cut down the cost of data handling.

In character recognition, human perception is capable of emphasizing an important feature while disregarding others, creating a full mental and symbolic image from an incomplete or poorly formed physical image of written characters. Machines are not capable of doing so, and this is the reason why machine recognition of handwritten characters is such a formidable task. The aim of pattern recognition is to use the machine's strength, its high computational speed to compensate for its weakness in perception.

Extensive study has been made in the field of handwritten character recognition. Although over a hundred methods on computer recognition of handwritten characters have been proposed and presented in the literature, computer recognition is not widely implemented in practice because the recognition rate on normal unconstrained handwritings is still low, and the cost in equipment and the computing power required are too high to be practical.

The problem of handwritten character recognition is a combination of theories in pattern recognition on one side, and the pragmatism of using human perception and human intelligence on the other side, to achieve a task which at first seems simple, yet has not been solved successfully by researchers. This study attempts to recognize totally unconstrained handwritten Zip code numbers using a knowledge-based system.

This thesis is divided into the following chapters:

Chapter 1: Brief Survey on the State of the Art.

Chapter 2: Description of Approach used in this Work.

Chapter 3: Preprocessing.

Chapter 4: Features and Feature Extraction.

Chapter 5: Training and Recognition.

Analyses and discussions on experimental results are treated in Chapters 6-10:

Chapter 6: Experimental Results - Overview.

Chapter 7: Analyses of Recognition Errors.

Chapter 8: Dependence on the Size of Training set.

Chapter 9: Performance Curve and Fine Tuning.

Chapter 10: Additional Experiments on Data used by Concordia OCR Project Team.

Chapter 11: Comparison with Other Studies.

Chapter 12: Conclusions.

1. BRIEF SURVEY ON THE STATE OF THE ART

1.1 PATTERN RECOGNITION METHODOLOGY

Pattern recognition is primarily concerned with the description and classification of measurements taken from physical or mental processes ([MANT87]). 3 major steps involved in pattern recognition are:

1/ Image processing, better known as **preprocessing**, consists of operations that transform the original image to other images which are an improved version of the input. Various techniques are used: edge smoothing, hole filling, normalization, thinning (skeletonization), rotation (correcting slanted characters so that they become vertical), centering (making the center of gravity of the character the center of the frame). Most preprocessing techniques have to transform the image pixel by pixel, and require considerable computing effort. In [GUDE76] rotation is found to be most costly (relative cost = 7) while edge smoothing (cost = 3.7) and thinning (cost = 3.7) are also heavy users of computing resource. The benefit of each preprocessing technique is strongly dependent on the feature extraction method and the recognition method that will be used.

2/ Image analysis (or pattern description), better known as **feature extraction**, is the process of getting measurements from the image so that it is possible to classify (recognize) the image. Feature extraction techniques fall into 2 types : **global analysis** and **structural analysis** ([SUEN82]).

One type of **global analysis** methods considers the character matrix and obtains different measurements such as: position and coordinates of points (mainly used in template matching methods), density of points, distance of elements from a reference point (moments), crossings of characters basing on different directions (characteristic loci).

Another type of global analysis methods uses mathematical transformation to convert the image matrix into some mathematical representation: Karhunen-Loeve series, Fourier transform, Walsh transform, Harr power spectra, Hadamard transform ([SUEN82]).

Structural analysis methods describe the image in terms of simpler geometrical patterns such as: skeleton, contour, components of straight strokes and curves, polygonal approximation, end points, fork points, loops, jumps, inflection points, concavities and convexities ...

3/ Classification, or **recognition** is the step where a decision is made on the identity of an unknown sample. There are 2 major categories of recognition methods: statistical and syntactical.

There are many types of **statistical methods.** In the **non-parametric methods,** the measurement taken from N features can be used to represent an N-dimensional vector space called feature vector. Accepted pattern classes are defined as hyperplanes in this vector space. For each unknown pattern, the distance of its feature vector to each hyperplane is measured. The recognition result is the pattern class having its hyperplane closest to the feature vector of the unknown pattern. The **parametric methods** use Bayes' Rule to make a recognition decision from the likelihood ratio of parameters (features) and the probability of occurrences of the pattern classes. Also belonging to the statistical family are **clustering analysis** (agglomerative or divisive), and **fuzzy** set methods ([MANT87]).

Syntactical methods analyze the features of a pattern as an input string and parse it to decide which class it belongs to. This family of methods depends mostly on the description of the pattern by its structure. So the term **structural method** (introduced by Pavlidis) comes into use to describe the approach that decomposes the pattern into structural primitives, and in many instances is analogous with syntactical method.

Statistical and syntactical methods have different advantages and disadvantages. It is only natural to expect that researchers would combine 2 methods into a **hybrid method** ([BAIR86]). Depending on the quality of the characters, some simple methods are adequate for well-formed characters, while more sophisticated methods are required for characters of poorer quality, thus there is the **multistage** approach ([LAM86], [DHTW80], [KS88]). Or, more than one method can be applied and a decision is made by selecting one candidate from the results given by all the methods, this is called the **majority voting** approach ([HULL88]).

Of particular interest is the **neural network** approach which was introduced recently ([KC88],[PLHS88]). The image matrix is fed to the input nodes in the neural net. There are the output nodes which give the result of recognition. Between the input and output nodes are intermediary nodes, all the nodes are interconnected to become a network. Each node generates an output depending on the input, while output from one node is the input to other nodes. The network is self-learning; in the learning step, each node modifies the function that generates its output using some reward or punishment factors so that the final classification output will be correct.

Table 1.1: Summary of recognition results from selected methods

Notes:

- In the column "type of character", "hand-printed" means that the characters were written on isolated boxes, the writers were aware that the characters would be used in machine recognition, and exercised some care when writing it.

- Column "Number of writers": public sampling means the samples were obtained from the public. For Zip code numbers, the number of writers is estimated to be 1 for each Zip code of 5 digits. Hence a sample size of 8500 is estimated to be written by 1700 writers.

Ref	Method used	Type of	Train	Sample	Recognition Rates			No of	CPU	Machine	Resolution	Guide
		character	/Test	Size	Good	Error	Reject	writers	cost(ms)	used		
AHM£86	shape-based table	unconst.	train	5000	93.68	3 96	2.36	public			166 dpi	
(AS87)		Zip codes	test	3540	89.55	3 45	7.01	sampling				
AS82	Multi-directional loci	unconst.	train	390	99 23	0 77	0 0	public			166 dpi	
	and other features	Zip codes	test	333	85.59	10 5	3.9	sampling			(53x53)	
BADR85		unconst	train	400				20			128x128	
(SB86)	Fourier descriptors	numerals	test	400	99							
	Topological features		test	450	99							
	Notes : total data base has 10,000 continuous samples of 3 digits (30,000) and 800 isolated digits											
BEUN73	Topological features	unconst.	train	15000				public				box
		numerals	test	10000	91.37	2.67	5.96	sampling				
BFW88	Topological features	unconst.	train	7000	86.6	0.3	13 1	map & chart				
		numerals	test					documents				
BK88	Points and stroke	numerals	train	400	98.25	0.75	1.0	20	1000	IBM PC-XT		
	analysis		test	400								
DHTW80	Edges and strokes	unconst.	train	5000				200			32x40	
		numerals	test	5000	99.50	0.50						

Ref	Method used	Type of character	Train /Test	Sample Size	Recognition Rates Good	Error	Reject	No of writers	CPU cost(ms)	Machine used	Resolution	Guide
FH83	Stroke analysis	hand-	train	30600				1297			24x16	
		printed	test	3400	92.65	0 65	6.7					
		alphanum.										
		num.only	test	1000	96.00	0.0	4.0					
	Note: characters with broken strokes were not included											
GUDE76	Loeve-Karhunen expansion	hand-	train	10080				200	1.1 ms		9x14	box
		printed alphanum.	test	10080	99.71	0 29	-					
	Note: Total data base has 108,000 samples (3000 x 36 classes) The characters were written following templates of 'master characters'											
HC86	Heuristic	unconst.	train	200					1000 ms	PDP-11	128x128	
	topological feature	numerals	test	200	97.0	-	-					
HOSK72	contours	OCR fonts	train (OCR font)	9431	99.56	0.05	0.38			Myriad	24x24	
		hand-prted	test	4051	97.75	1.11	1.14					
HULL88	template-matching	unconst.	train	1754				public			300 dpi	
		Zip codes	test	8129	61-86	2-11	12-28	sampling				
	boundary approximation		train	1754								
			test	8129	73	4	13					
	structural		train	1754	88							
			test	8129	86 3	3 2	9.6					
	voting from 3 methods		test	>8000 (unclear)	91.0	1 7	7.3					
HUNT72	skeleton and visual features	hand-	train	1100	98.00	0 50	1.50		20 ms	ICL 4120	20x15	box
		printed	test	8000	97.00	0.50	2.50					
KC88	neural network	printed	train	1200				public		VAX 8800	300 dpi	
		& HW Zip codes	test	800	77-88	3-12	0-20	sampling			compressed to 12x12	
KPS79		hand-	train	6000				30			32x64	box
	n-tuples	printed	test	6000	97.68	2.32						
	moments		test	6000	93.16	6.84						
	characteristic loci		test	6000	98.96	1.04						

Ref	Method used	Type of character	Train /Test	Sample Size	Recognition Rates Good	Error	Reject	No of writers	CPU cost(ms)	Machine used	Resolution	Guide
KS88	structural method, stroke from run-length analysis, contour profiles	unconst. Zip codes	train test	 7100	 93.3	 2.5	 4.2	public sampling			300 dpi	
LAM86 (LS88)	structural and relaxation matching	unconst. Zip codes	train test	5000 2000	97.38 93.15 96.00	2.15 2.25 3.15	0.47 4.6 0.85	public sampling	24 ms 662 ms	Cyber 835	166 dpi	
	Computing cost : 24 ms for structural method 662 ms for relaxation matching											
MAI88 (this study)	topological, knowledge-based	unconst. Zip codes	train test	8500 8485	96.59 94.47	1.05 2.49	2.36 3.04	public sampling	303 ms	Cyber 835	166 dpi	
MCLA68	correlation	hand-printed	train test	2725 3100	 99.42	 0.58		4			25x32	box
MG88	structural method, feature matching	unconst. Zip codes	train test	5000 1000	94.0 81.0	0.1 2.0	5.9 17.0	public sampling		Cyto-HSS Symbolic 3650	5 passes in thinning (10 pixels thick)	
MONG82	syntactic geometrical primitives	hand-printed	train test	3000 3000	99.70 98.60	0.03 0.07	0.27 1.33	30				
PLHS88	neural network	unconst. Zip codes	train test	391 1173 1173 1173 1173 1173	 33-74 54-60 54-68 15-35 23-40	 2-26 34-40 22-32 17-65 24-60	 0-64 0-12 0-24 0-68 0-53	public sampling	2-4 sec up to 60 sec	Symbolic 3640 Sun-3 Sun-4	300 dpi compressed to 16 x 16	
PWY87	floating mask feature extraction	Hand-printed	train test	 200000	 99 93	 -	 -	90	14 ms	8 x Z80 processors		
WATA88	Karhunen-Loeve expansion	Kanji	train test	 16000 mails	 40-56				3 mails/s	6 x 68000 processors	200 dpi	

1.2 EVALUATION OF EXISTING METHODS

The methods and results reported in studies on optical character recognition are bewildering. A summary of selected methods are given in Table 1.1. Although this thesis concentrates only on the recognition of totally unconstrained numerals, some methods which recognize carefully written characters, numerals and upper case alphabets, and also Kanji characters are included, because there are similarities in methodology for different types of characters.

Recognition rates as high as 99.9% have been reported. It can be noted that the recognition rate is very much dependent on the quality of the characters as well as the method used. The recognition rate of 99% reported by Badreldin and Shridhar ([SB86]) was obtained by 2 methods, one using Fourier descriptors and statistical recognition, and the other using geometrical measurements and syntactical recognition. The rate of 99.7% reported by Gudesen ([GUDE76]) was from a method that uses normalization in preprocessing, Karhunen-Loeve expansion for feature extraction, and statistical recognition (Bayes' method). Yu Mong ([MONG82]) used geometrical primitives and syntactical recognition to obtain a recognition rate of 98.6%.

Although more than one hundred methods have been proposed in the literature, it is not possible to compare them and to decide which method would be the best approach, for the following reasons:

1/ Quality of the characters: In most studies, some judgments were made by the author(s) to exclude a number of poorly written characters from the data base. These judgments are subjective because different authors will accept and reject different samples. The quality and the authenticity of the data are different from one study to another. For standardization and accurate evaluation of recognition methods, a common data base should be established so that all researchers do not have to compare oranges with apples. So far, this common data base does not exist.

2/ Size of the data base and the number of writers: the small size of the character data base and the small number of writers do not guarantee that the samples are representative of real-life situation. It can be expected that unconstrained handwriting will create more pattern classes than careful writing, and the number of classes increases with the number of writers and number of samples.

3/ Some methods are of theoretical interest, but in the foreseeable future, are not cost-effective or reliable enough to be implemented in practice: either the recognition rate is too low (neural network), or the computation cost is too high (Fourier descriptors, relaxation matching, neural network).

In Table 1.1, it is observed that for all methods that have high recognition rates (98-99%), their data bases either consist of characters written in a controlled manner (such as following a set of given models) or they are small, or they are sizeable data bases but involving only a small number of writers. One exception is the work by Duerr et al [DHTW80], which is discussed in Chapter 11.

For data bases which are known to have totally unconstrained handwritten characters collected from a large number of writers from the general public, the best recognition rates reported are listed below:

Table 1.2 : Summary of recognition results of totally unconstrained characters

REF.	RECOG. (%)	SUBST. ERROR	REJECTION (%)	RELIABILITY (%)	SIZE OF DATA TRAIN	TEST	RESOLUTION OF DATA (DPI)
BEUN73	91.37	2.67	5.96	97.16	15,000	10,000	-
HULL88	91.0	1.7	7.3	98.17	1.754	>8,000 (1)	300
KS88	93.3	2.5	4.2	97.39	-	7,100	300
AHME86	89.55	3.45	7.01	96.29	5,000	3,450	166
LAM86	93.15	2.25	4.6	97.64	5,000	2,000	166
LAM86	96.00	3.15	0.85	96.82	5,000	2,000	166
MG88	81.0	2.0	17.0	97.59	1,500 (2)	1,000	(3)
This study	94.47	2.49	3.04	97.44	8,500	8,485	166

Note: (1) Not accurately specified
(2) This training set, which was used together with the test set of 1000 samples in this table is not the same as the training set in Table 1.1 which had 5000 samples
(3) Good resolution: thinning required 5 passes

Further analyses and comparison of results are treated in Chapter 11.

2. APPROACH USED IN THIS WORK

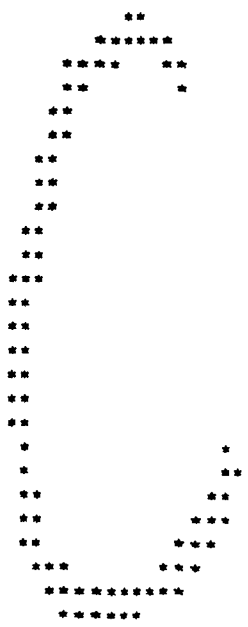
2.1 PROBLEMS RELATED TO UNCONSTRAINED HANDWRITTEN NUMERALS

In unconstrained handwriting, normally the writer does not take special care to make sure that the characters are properly formed when writing, the writing medium such as pen and paper have not been selected to give good written characters. As illustrated with some samples taken from the training set, the following problems may occur:

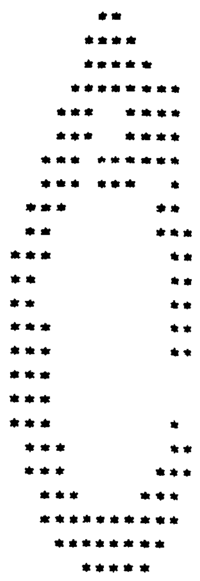
- **Incomplete stroke** formation: the writer did not write the full stroke; or when he(she) did, the scanning and preprocessing steps failed to record the full stroke (see numerals 56, 3185 in Fig. 2.1).
- The writer produced **extra strokes** in writing, he(she) might overwrite or retrace to correct the character. The character might be written in a fancy way with some ornamental strokes (numerals 523, 4179, 4444 in Fig. 2.1).
- **Careless writing**: the writer did not write the character in a proper way (numerals 3650, 7197 in Fig. 2.1).
- Type of **pen** is also important. When writing small characters, a pen with a thick tip will produce a poor image resolution. Also, if the ink of the pen does not flow smoothly, white holes may be created inside the stroke (numerals 683, 3393, 6080 in Fig. 2.1).
- The quality of writing **paper**: A coarse paper surface and thick pen stroke will produce characters with uneven border and inside holes (numeral 683 in Fig. 2.1). The paper may have lines on them which produce background noise (numerals 683, 1022 in Fig. 2.1).

Fig. 2.1 : Examples of problems found in unconstrained handwriting

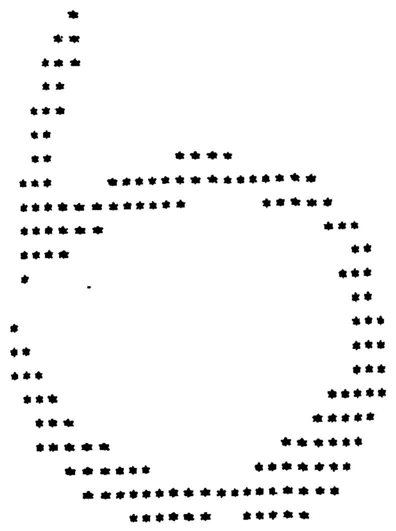
NUMERAL NO. 56 : 0
Incomplete stroke



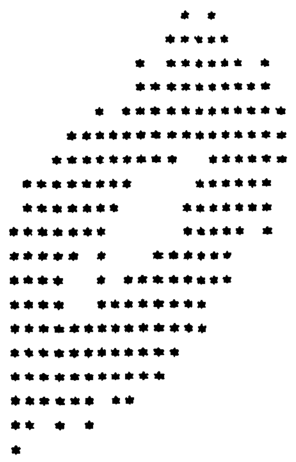
NUMERAL NO. 146 : 0
Coarse paper or uneven ink flow



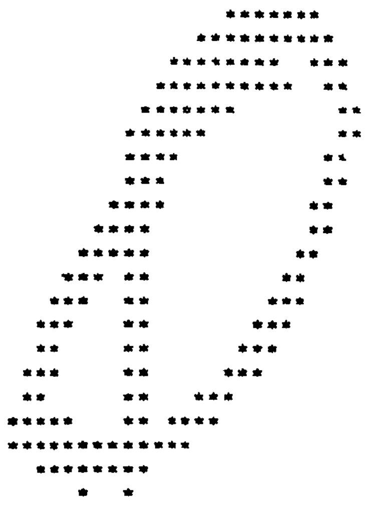
NUMERAL NO. 523 : 0
Extra stroke



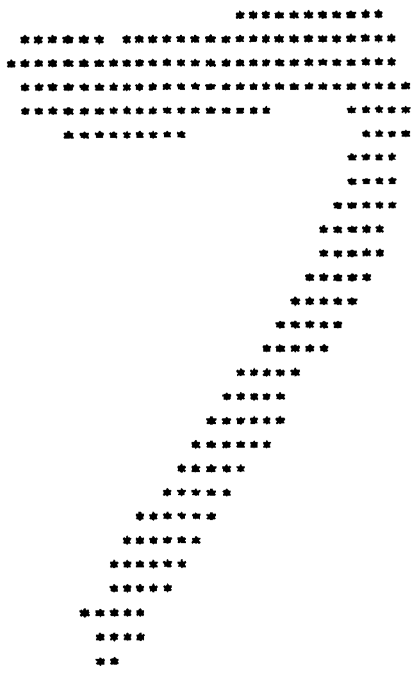
NUMERAL NO. 683 : 0
Coarse paper and thick pen tip



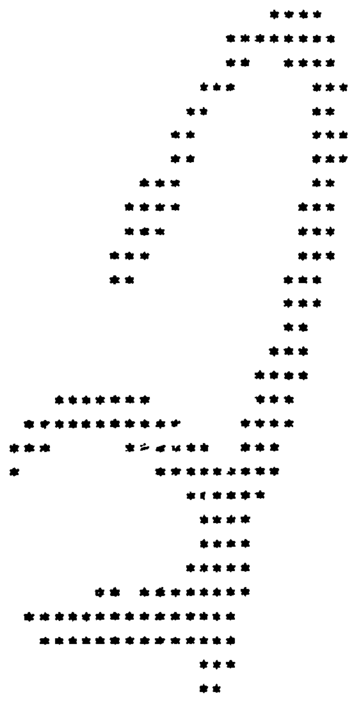
NUMERAL NO.1022 : 0
Background noise



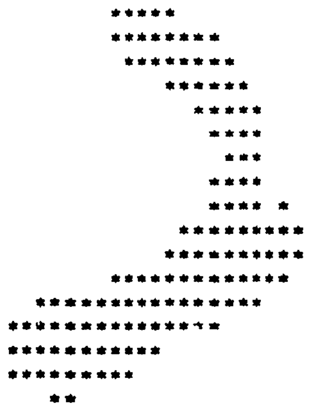
NUMERAL NO.3178 : 2
Missing stroke or careless writing



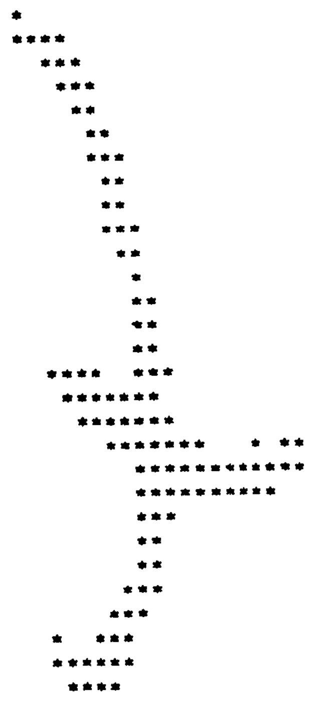
NUMERAL NO.3185 : 2
Incomplete stroke



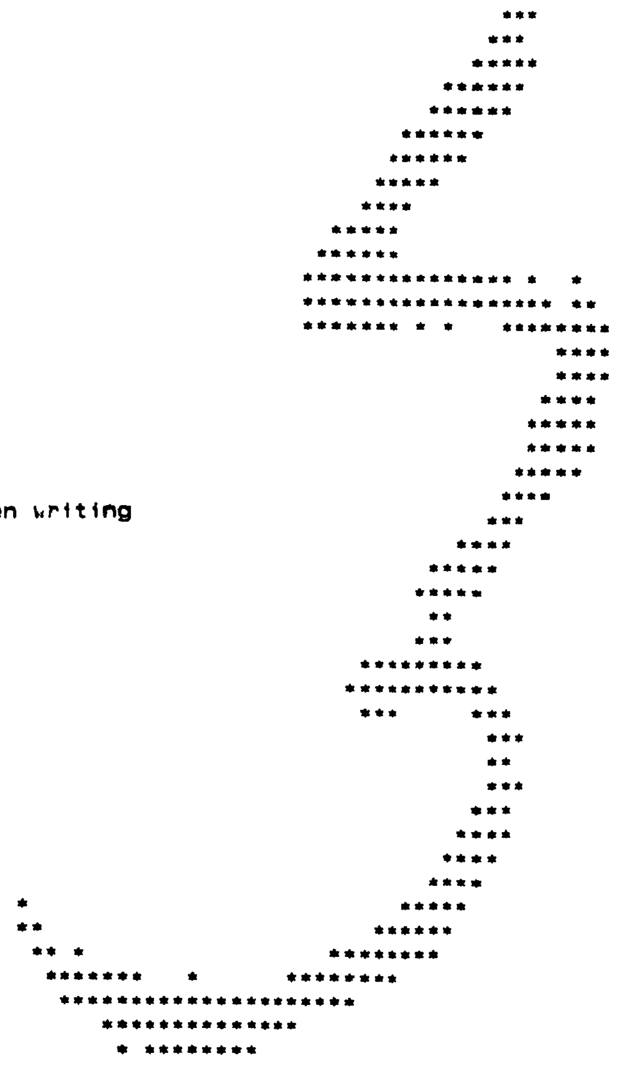
NUMERAL NO.3393 : 2
Thick stroke, loss of detail



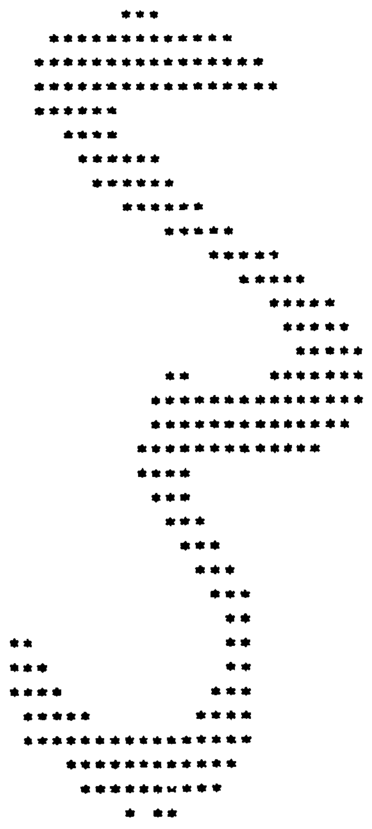
NUMERAL NO.3650 : 2
Careless writing



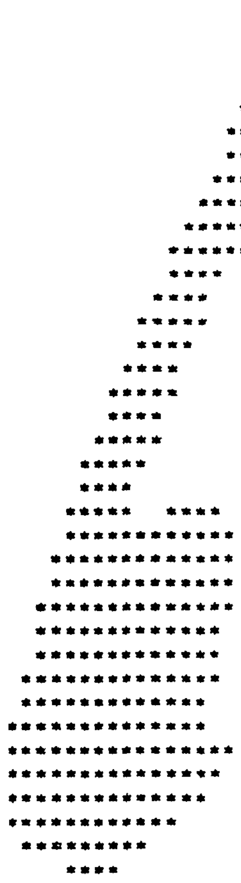
NUMERAL NO.4179 : 3
Extra stroke



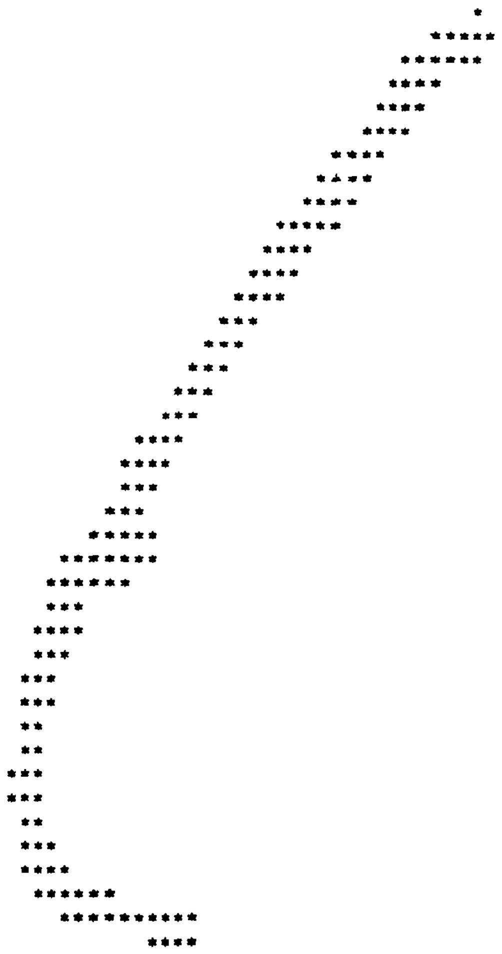
NUMERAL NO.4444 : 3
Fancy writing or change of mind when writing



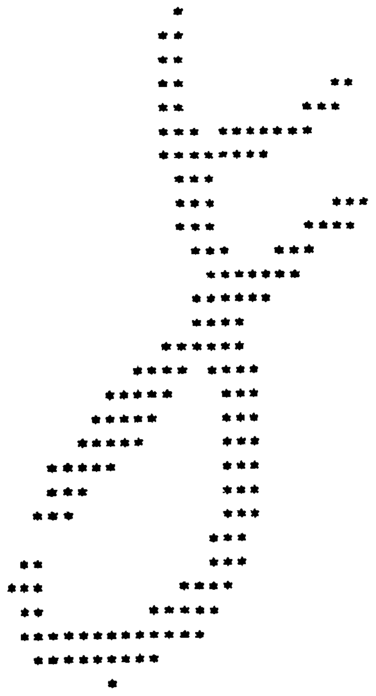
NUMERAL NO.6080 : 6
Thick stroke, loss of detail



NUMERAL NO.7197 : 7
Improper formation of character



NUMERAL NO.7587 : 8
Correction or retracing stroke



In free writing with no constraints imposed, the number of pattern classes that are created can increase dramatically. For example, the skeleton representation of numeral 4 may have from 0 to 4 fork points:

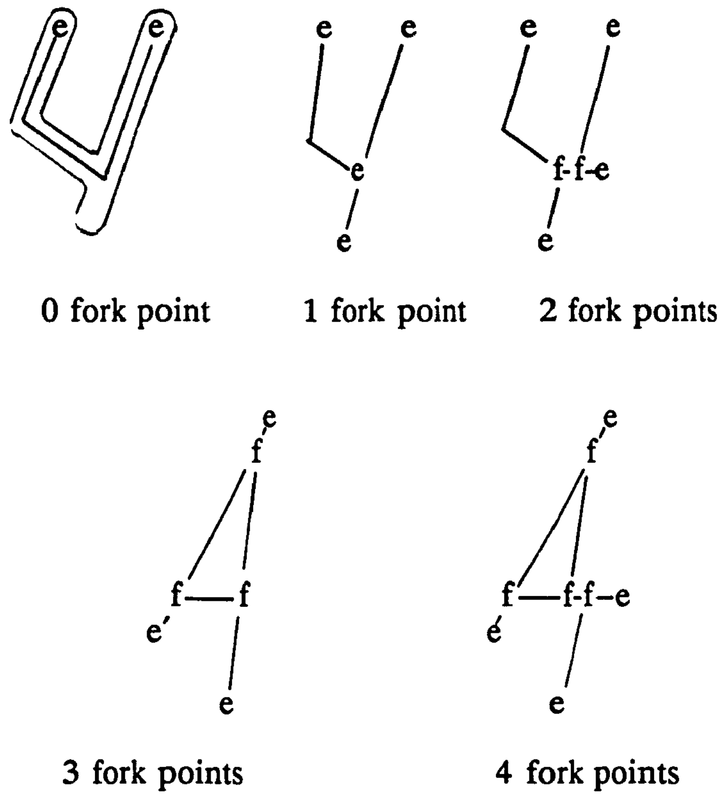


Fig. 2.2 : Different configurations of numeral 4

With extra fork points produced by noise and poorly formatted strokes, the number of possibilities may increase even further. This variety of character forms and shapes will confuse most structural methods. Hence a method must be very sophisticated to be able to handle most cases.

Likewise, statistical methods and methods that use global analysis can also be confused by the variation in the black pixel positions of the character image. Take the non-parametric methods for example, the large number of pattern classes will create a large number of hyperplanes and increase the probability that many hyperplanes have equal distances from the unknown vector.

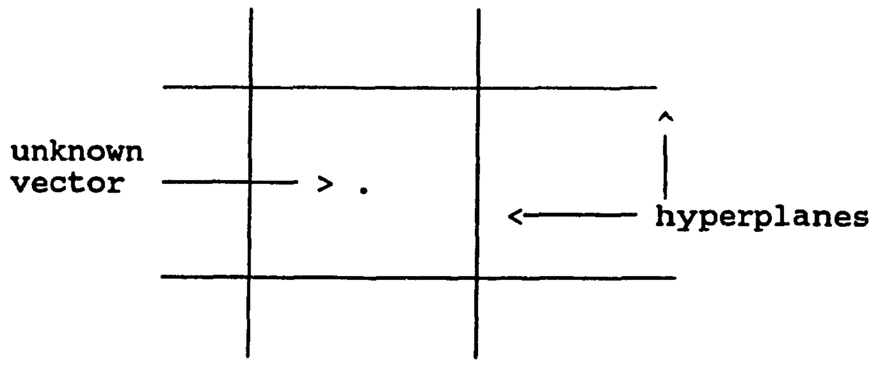


Fig. 2.3 : Confusion due to a large number of hyperplanes

2.2 GOALS OF THIS THESIS

2.2.1 The role of human intelligence in recognition

For accurate recognition, ideally all the character 's features should be extracted such that it is possible to reconstruct the original image from the features, or at least an image close to the original one that can be recognized by human. In most pattern recognition methods, only the features considered to give accurate and unambiguous recognition within the specified context need to be extracted, it is not important that the original image can be reconstructed from those features. By the specified context, the method knows beforehand what type of characters it is dealing with, such as: numerals only, upper case letters only, etc..

In a method that does not use human intelligence in recognition, the character may be transformed into mathematical entities, or grammar tokens, these representations can not be visually perceived by human. When the recognition rate needs to be improved, it will be found that an elaboration or modification of the method using mathematical representation or grammar tokens will not lead to improvement, in fact, there may be no more direction to go from there. A method using human intelligence to guide the feature extraction process allows the method to be constantly refined by addition of more human knowledge; hence the recognition rate can be improved to be near or surpass the human recognition rate.

2.2.2 Goals of this thesis

Unconstrained handwriting is a real-life situation. To recognize them the method cannot consider only the ideal situations. All the elements of this method should aim to exploit the real features from the real data. Hence, the goals of the thesis are:

1/ To have the algorithm behave like human: Human recognizes things by storing the knowledge about features of known objects in memory, this is what we call experience. This experience will be used to recognize unknown objects. The computer would store knowledge about features from training in its memory, and use this knowledge to recognize unknown characters.

2/ To be objective in approach: Once a researcher decides to disregard any samples, some degree of subjectivity will be introduced into the method, consequently the features selection will also be subjective. When features are not subjective (i.e. dependent on the human expert and the training set, but do not reflect the general features of all numerals), the method will perform well with a training set, but when applied to a test set, the recognition rate will deteriorate. To assure objectiveness, the training samples would be selected randomly, no visual inspection would be made to reject any character from the training set, no matter how distorted or how poorly it has been written. The rejection would occur in the recognition process, where the algorithm would compute that the features found on those poor samples are self-conflicting and do not result in an acceptable confidence level. The method therefore would automatically handle non-representative samples without human intervention.

3/ To be able to handle overlapping features: One of the major problems in pattern recognition is overlapping features (for example 1 and 7, 7 and 4, 0 and 6, 4 and 9). The statistical recognition methods can have some clustering method to produce a boundary between overlapping features. The syntactical methods have to use a much more complicated grammar to handle it. In this method, the decision would be made from consideration of all features, represented by their actual occurrences in the training set. Since many features would be involved, the overlapping aspect would be handled automatically and flexibly.

4/ To promote machine learning so that the machine can participate in extracting and organizing the features without guidance from the human expert. In this study, the primary features would be automatically generated by the computer which would also organize secondary features and record statistical data on the features.

5/ To reach an acceptable degree of accuracy.

6/ To have an acceptable efficiency: it would not incur too high a computational cost.

2.3 DESCRIPTION OF THE METHOD

In most recognition methods, in the training stage, the samples are examined by the human expert. From examinations and analyses, a number of features will be selected, then methods to extract these features will be designed. The classifier will have the knowledge given to it by the human expert, in the form of a decision tree, a grammar, a number of templates or hyperplanes... In the recognition stage, the computer compares the features of the unknown characters with the decision tree, or the grammar, set of templates or hyperplanes.., and a decision on the assigned class is given.

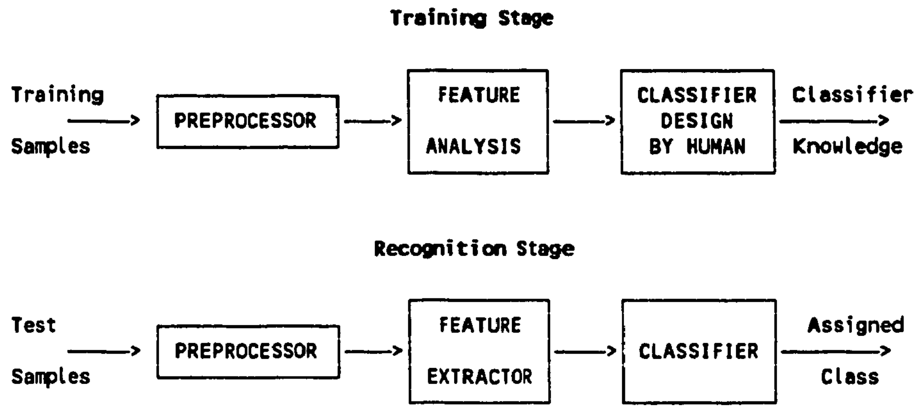


Fig. 2.4 : The usual approach in recognition

In this method, building the classifier knowledge was a co-operative effort from both the human expert and the computer. The human expert suggested the features that the computer should look at, the computer inspected the training samples for the features, organized the data on the frequency of occurrences of the features, and used these data to recognize a sample. Examining the inference process that the computer made and its recognition result, the human expert would add more features in order to improve the recognition rate, he might also decide to take out some previous features if those features did not contribute to a better recognition rate. Therefore, the computer was not only a recognition tool, but also a useful partner with the human expert in the design of the classifier.

2 recognition methods were used in this study:

1/ In the inference method, the knowledge base kept the frequency of occurrences of the 10 numerals that satisfied a rule (i.e. having a feature). The list of 10 values of frequency of occurrences could be considered as a list of hypotheses, each value represented a likelihood measurement of a numeral being present in the training set. Recognition was an inference process using set operations to find the set of known numerals in the knowledge base that had all the features of the unknown numeral, and from this set of hypotheses, a decision on the most probable assigned class was made using Bayes' Rule.

2/ Another knowledge base was used to contain entries about **pattern classes**, from the most common to the most unusual. Each pattern class consisted of samples that had the **same combination of features**. The knowledge base could be 'purified' by filtering out pattern classes which had a number of occurrences lower than a threshold. By changing the threshold values, different degrees of reliability could be achieved. Recognition was done by a structural classifier using feature matching.

A block diagram of the overall process is given in Fig. 2.5.

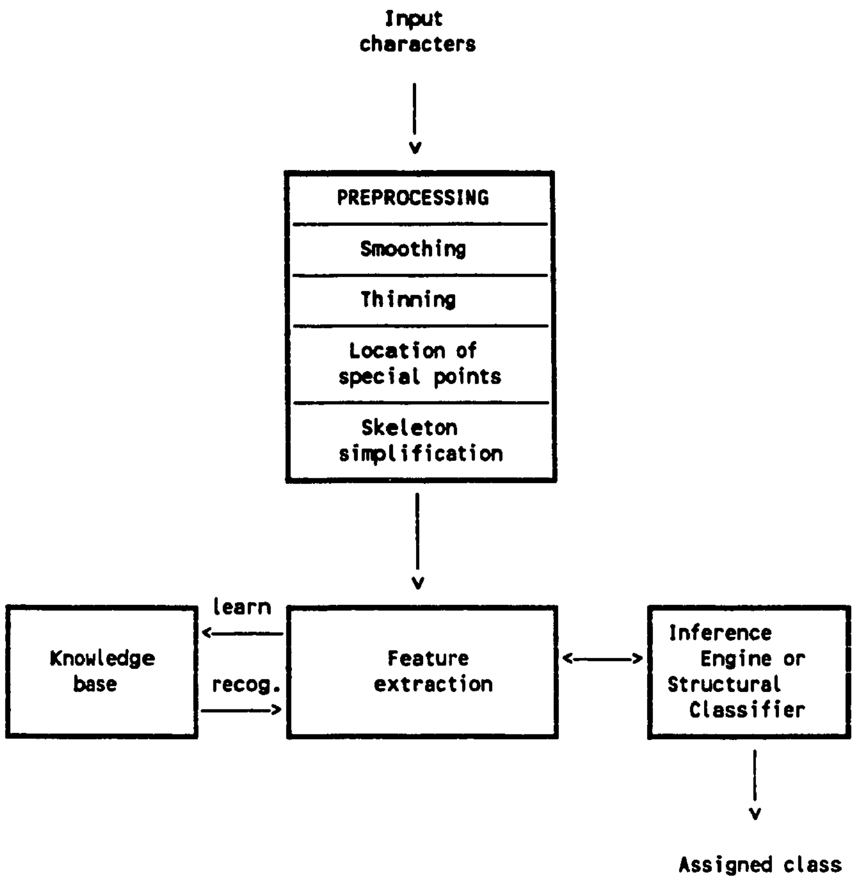


Fig. 2.5 : Block diagram of the method

3. PREPROCESSING

The data base was collected from dead letter envelopes by the U.S. Postal Services at different locations in the United States. The samples were digitized at a resolution of approximately 166 dots per inch, then binarized, enhanced and segmented. The above preprocessing steps are described in detail in [SAL88]. Further preprocessing was done in this study.

Preprocessing helps to obtain simpler patterns that will yield more discriminative features in the feature extraction process. Depending on the feature extraction and the recognition method that are used, some preprocessing techniques are favored over others. As this method is based on human perception which usually interprets the character image as a combination of strokes, thinning was used in order to obtain a stroke representation of a character. Before thinning, edge smoothing as well as hole filling were applied to produce a better skeleton. The effect of normalization (mainly size reduction) was also investigated.

The skeleton obtained from thinning consists of the following special points connected together by stroke segments:

E : end point (tip of stroke)

F : fork point (a point having 3 strokes coming from it)

X : cross point (a crossing point of 2 strokes).

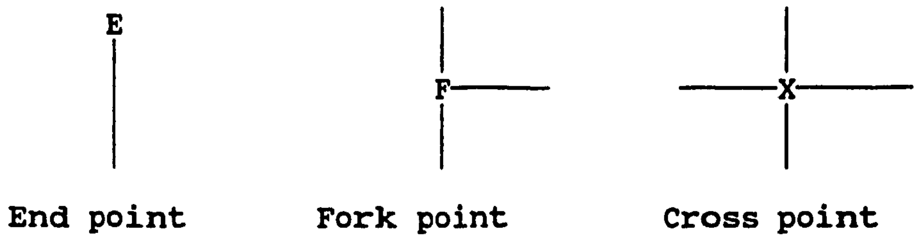


Fig. 3.1 : Special points in a skeleton

3.1 NOISE IN SKELETON

4 major types of noise on the produced skeletons are identified:

1/ A white noise on a black background will be amplified and extra fork points are generated. In this report we call this type of noise F-type (Fig. 3.2a).

2/ Touching pixels from 2 strokes generate extra fork points. This type of noise is also called F-type (Fig. 3.2b).

3/ A protrusion along the edge of stroke or a sharp turn in a stroke produces a tail (1 fork + 1 end point). This type of noise is called E-type (Fig. 3.2c).

4/ A loop is thinned into a single stroke. This type of noise is called L-Type.

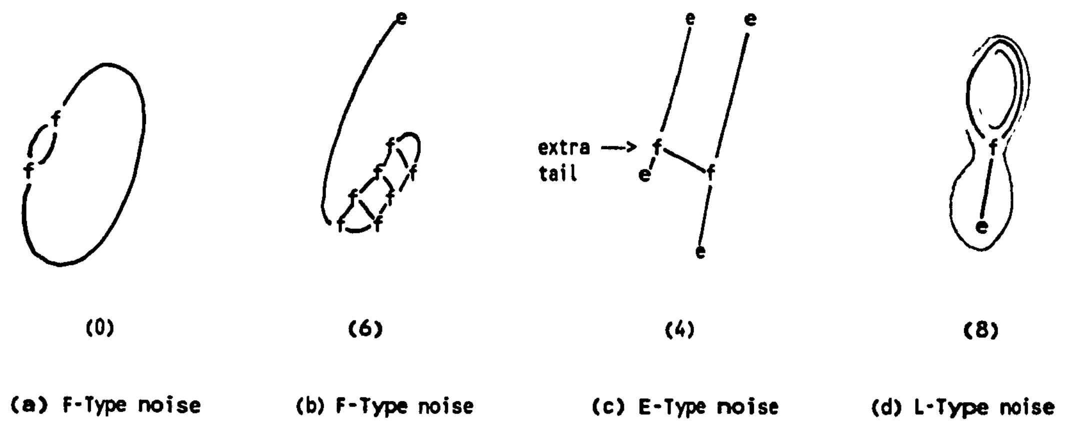


Fig. 3.2 : Undesirable noise on skeletons

The E-type noise caused by a sharp turn in a stroke helps to identify an important feature. On the other hand, an E-type noise caused by extra stroke or in a middle of a straight stroke only makes the skeleton more complicated. E-type noise is comparatively easy to handle in the later stages.

The F-type noise is more difficult to handle, it produces recognition errors and causes the recognition algorithm to be more complicated.

The L-Type noise is actually a loss of information from the original image. It will produce more substitution errors.

The above types of noise can be reduced by 3 methods: size reduction, noise removal from the original image, and edge smoothing.

3.2 SIZE REDUCTION

Size reduction can be considered as a 2-dimensional transformation that may reduce the effect of irregularities on the edges of the strokes, making it easier to get a simple pattern (Fig. 3.3). It can also remove some white noise in a black background (Fig. 3.4).

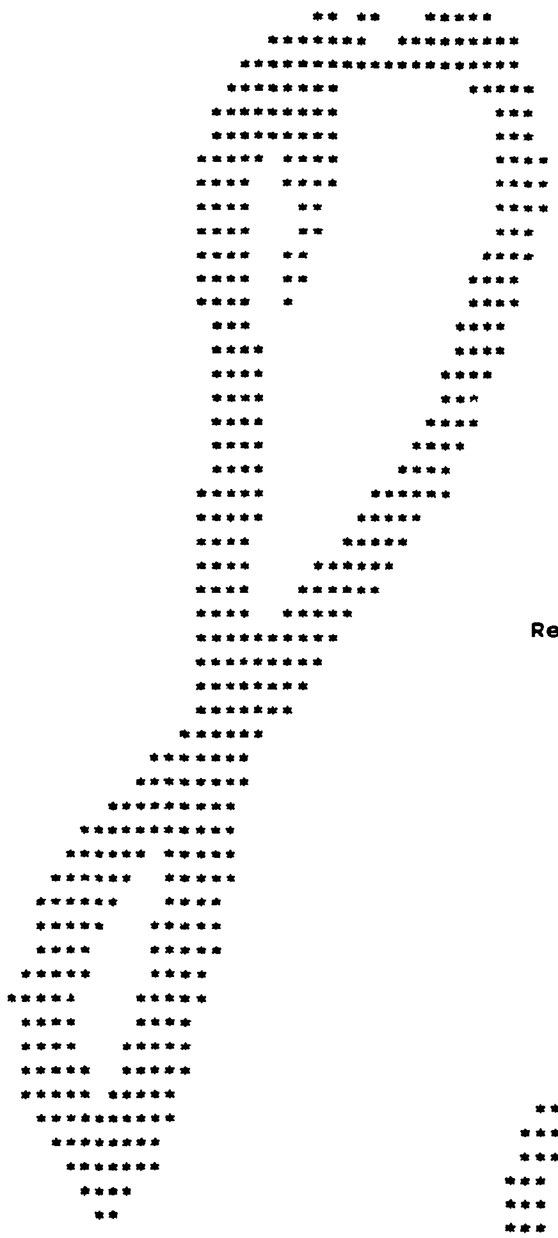
Size reduction decreases the number of black pixels in the pattern, and can be considered as a partial thinning operation that reduces the thickness of a character in a uniform way. The thinning method of inspecting 3x3 window always follows an order in scanning and removes pixels in the same order. It may erode the character in one direction more than the other, not uniformly like size reduction.

An additional benefit of size reduction is that the smaller pattern will take less CPU time in processing.

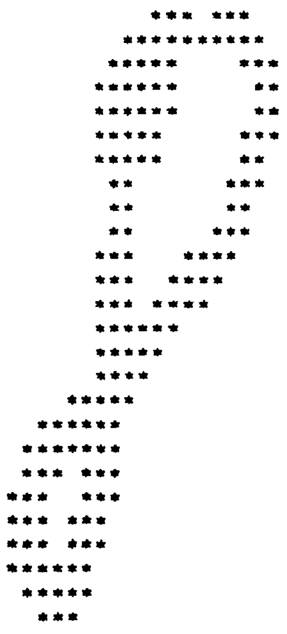
Size reduction will reduce the E-type noise on the skeleton. However, when the

Fig.3.3 : Size reduction helps to get a simple pattern

Full Size 51 x 51



Reduction to 26 x 26



Produced skeleton

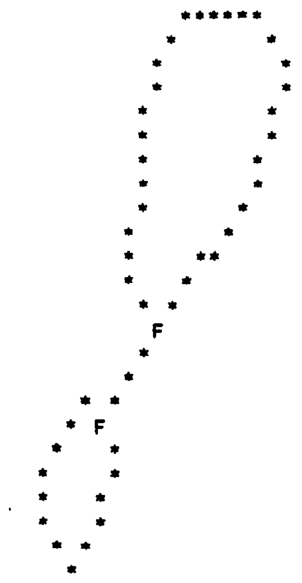


Fig.3.4 : Size reduction removes white noise

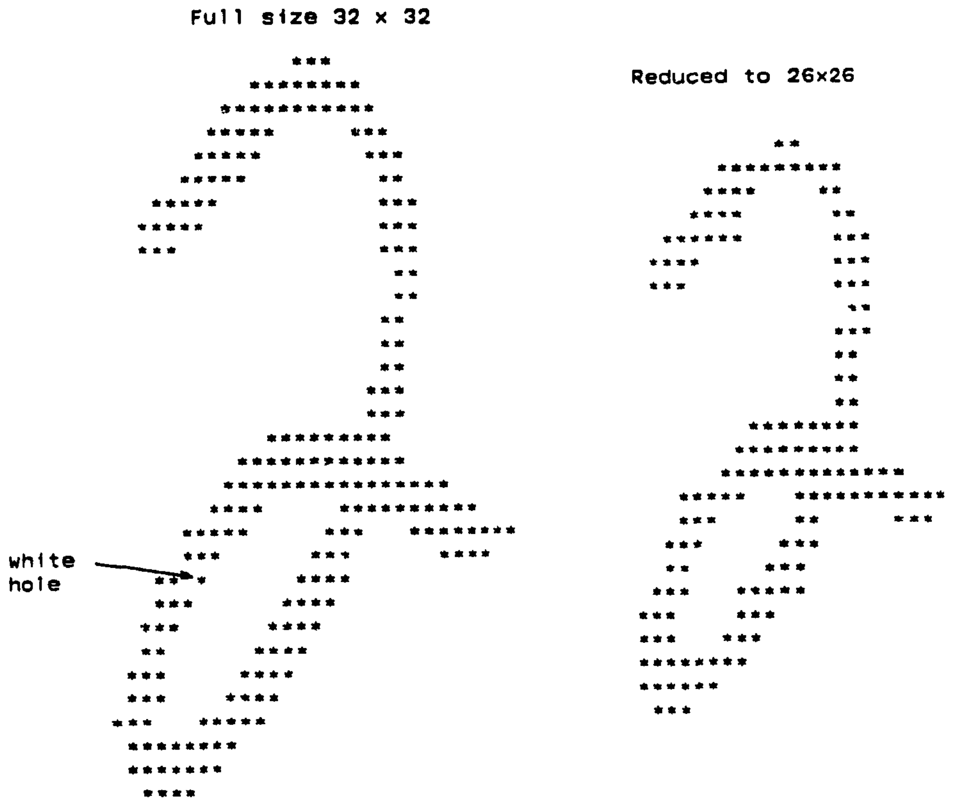
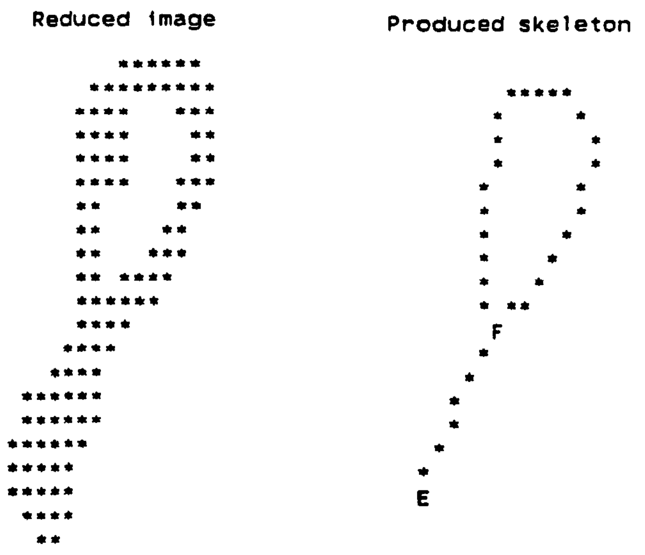


Fig.3.5 : Over reduction will fill up a loop
(The full-size character shown in Fig.3.3 is reduced to 21x21)



strokes are brought closer to one another by size reduction, they may become close enough to introduce F-Type noise. Size reduction may also fill up a valid white area inside the pattern, and change a loop into a single line, producing L-type noise (Fig. 3.5).

3.3 NOISE REMOVAL FROM ORIGINAL IMAGE

The image is scanned by a window of 3x3 pixels. The window has 9 pixels, divided into **majority** pixels and **minority** pixels, the majority pixels being the ones that have a larger count in the window than the minority pixels. If there is only 1 minority pixel and 8 majority pixels, the minority pixel is considered as noise and changed to a majority pixel (Fig. 3.6). The operation is called **noise removal.**

```
      .  .  .                 *  *  *
      .  *  .                 *  .  *
      .  .  .                 *  *  *

    Black noise             White noise
```

Fig. 3.6 : Noise of 1 minority pixel in the window

3.4 EDGE SMOOTHING

If the noise removal operation is extended to those cases where there are 2, 3 or 4 minority pixels in a 3x3 window, in addition to some noise being removed, then the irregularities at crossing points or bending points in the image tend to even out, so that thinning will give a simpler pattern without extra fork points or tails (F-type and E-Type noise). We call this operation **edge smoothing.** The undesirable effects of edge smoothing are:

- when changing white pixels to black, it may overfill a small loop and change the loop to a line and create an L-type noise,
- when changing black pixels to white, it may create a discontinuity in a thin stroke.

The advantages and disadvantages of size reduction and edge smoothing are very similar. Edge smoothing is more flexible because the rules to switch a pixel from white to black or vice versa can be varied to give different results. On the other hand, edge smoothing requires more computing time.

3.5 EXPERIMENTS IN PREPROCESSING

Different degrees of size reduction and different criteria for noise removal and edge smoothing were attempted to determine a set of conditions that produces a simple skeleton retaining all the geometrical features of the original image. The skeletons produced by different preprocessing parameters were inspected visually and some measurement of the amount of noise (**Error Count**) in the skeletons was made. It was found that some preprocessing methods intensify one kind of noise while other methods intensify others. By inspecting the Error Counts (EC) in a wide solution space, optimal areas which give the right combination of preprocessing parameters can be found.

200 samples from the data base (20 for each numeral) were extracted and put through the following steps:

1/ Different size reduction: no reduction, or reduced to smaller sizes.

2/ Noise removal.

3/ Different degrees of smoothing (including no smoothing).

4/ Thinning using an algorithm by Shinghal and Naccache [NS84].

For the first set of experiments (Set 1), there was only noise removal, so that the effect of size reduction could be seen more clearly. In the next 4 sets, the rules were extended to produce edge smoothing.

From the results of the first set, 112 samples out of the 200 samples gave similar skeleton results for full size as well as when they were reduced to 31x31, 26x26, 21x21 and 16x16. There were 88 samples that gave different skeletons for different reduction sizes. These 88 samples were used in different trial runs using different combinations of size reduction and edge smoothing.

After each run, the skeletons obtained from the samples were visually inspected, and any features on the skeletons considered as noise were recorded.

The noise was classified as E-Type, F-type or L-Type as discussed in Section 3.1. Since the effects of noise on the difficulty and accuracy of recognition were different, each type of noise was given a different weight. The assignment of the weight factors was intuitive and subjective, as follows:

- E-Type noise is easiest to handle, weight factor = 1.
- F-Type noise makes recognition much more difficult, weight factor = 2.
- L-Type noise can cause misrecognition, weight factor = 3.

The number of noise was multiplied by the weight factors and added together to give a value representing the Error Count (EC) of the run.

The results of the runs on Set 1 (with noise removal, but no smoothing) is given in Table 3.7.

Table 3.7 : Results of Set 1 (200 samples)
(No smoothing applied)

	Full Size	Normalized size		
		26x26	21x21	16x16
Total no. of noise/errors	21	18	19	24
L-type	0	1	5	11
F-type	17	16	14	13
E-type	4	1	0	0
Error Count	38	36	43	59

3.5.1 Description of different runs

Each set of experiments has a different way of smoothing the image. For convenience, the number identifying the set of experiment and the smoothing method used is the same (eg. Set 2A uses smoothing method 2A).

Each smoothing method inspected the pixels in a 3x3 scanning window, and used different rules to switch color of the center minority pixel. We distinguish the following criteria to decide on smoothing operation:

- Pixel count basis: smoothing is considered if the number of the minority pixels in the scanning window is 1,2,3 or 4. In the case of 1, we have noise removal.
- Pixel connectivity: the majority pixels surrounding the minority pixel in the center are inspected to see whether they are connected together or not.

```
* . *          * * *
* . *          . . *
. * .          . . *
```

(a) 4 minority pixels 5 majority pixels are not connected

(b) 4 minority pixels 5 majority pixels are connected

Fig 3.8 : Connectivity condition

- Symmetry of color: the smoothing rule is applied to both black and white pixels, or just applied to one type of color only.
- Propagation of smoothing: with propagation, when a minority pixel is changed to have the majority color, it is considered to have the new color in following inspections, thus the change can be propagated. With no propagation, when a pixel changes color, it is not considered as having the new color in next inspections.

The sets of experiments were selected so that these factors, together with the size of the image would be considered. The sets and the methods used are summarized in Table 3.9. Summary of results are given in Table 3.10, graphs of Error Count for different image sizes and different smoothing methods are plotted in Figures 3.11 to 3.14.

Tahle 3.9 : Summary of smoothing methods

Methods	2	2A	2B	2C	3	3A	3B	3D	3E	3F	4	4A	4B	4D
No.of minority pixels														
2	x	x	x	x	x	x	x	x	x	x	x	x	x	x
3					x	x	x	x	x	x	x	x	x	x
4											x	x	x	x
Propagation		x	x			x	x		x			x		
Connectivity of majority pixels	x	x	x	x	x	x	x			x	x	x	x	
Symmetry of smoothing			x	x			x			x			x	
Minimum error count	18	14	19	17	17	22	21	11	14	18	19	19	20	16

Table 3.10 : Summary of results

Notes : * denotes the minimum error count in the row

Size used in thinning	Full	31x31	26x26	21x21	Total
Experiment Set					
1	38		36*	43	
2	21	25	18*	35	99
2A	20	21	14*	34	89
2B	19*	22	22		
2C	20	25	17*		
3	23	24	17*	35	99
3A	22*	22*	22*	40	106
3B	21*	22	22	40	105
3D	11*	15	24	36	86
3E	14*	14*	25	45	98
3F	18*	21			
4	21	21	19*	35	96
4A	25	27	19*	40	111
4B	22	25	20*	40	107
4D	16*				

Fig. 3.11 : Error count of Set 1
Noise removal only

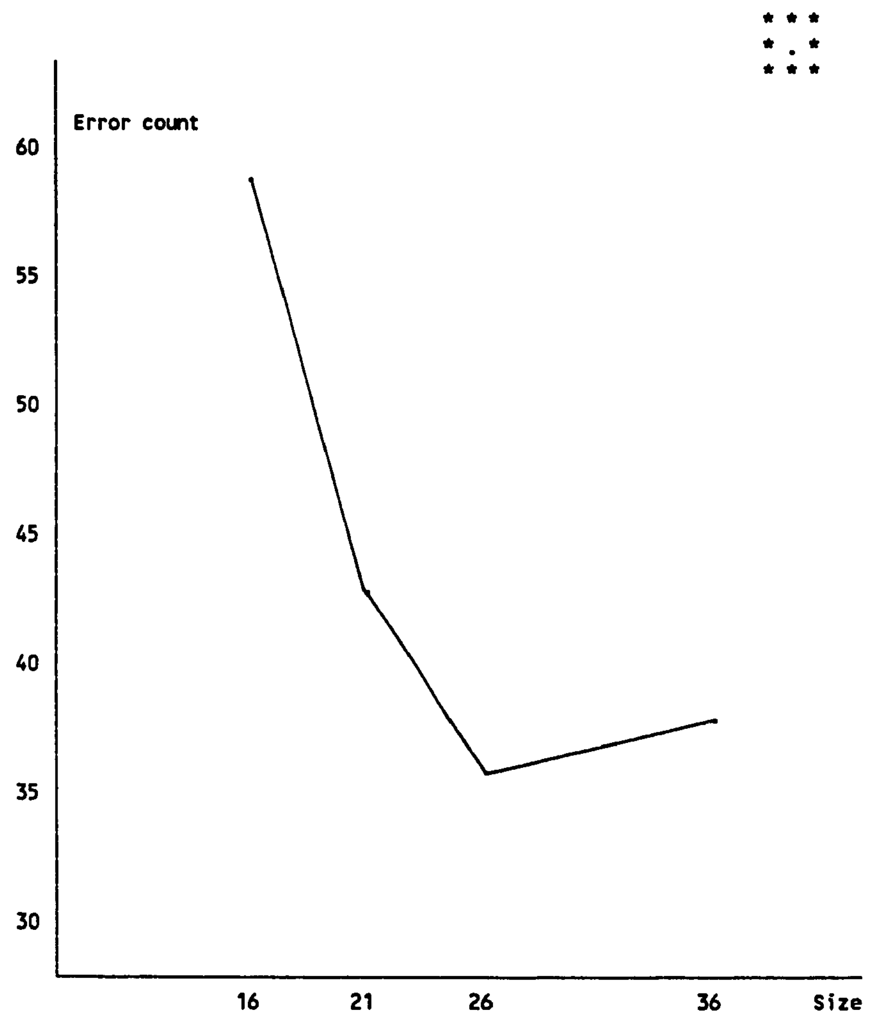


Fig. 3.12 : Error count of Set 2

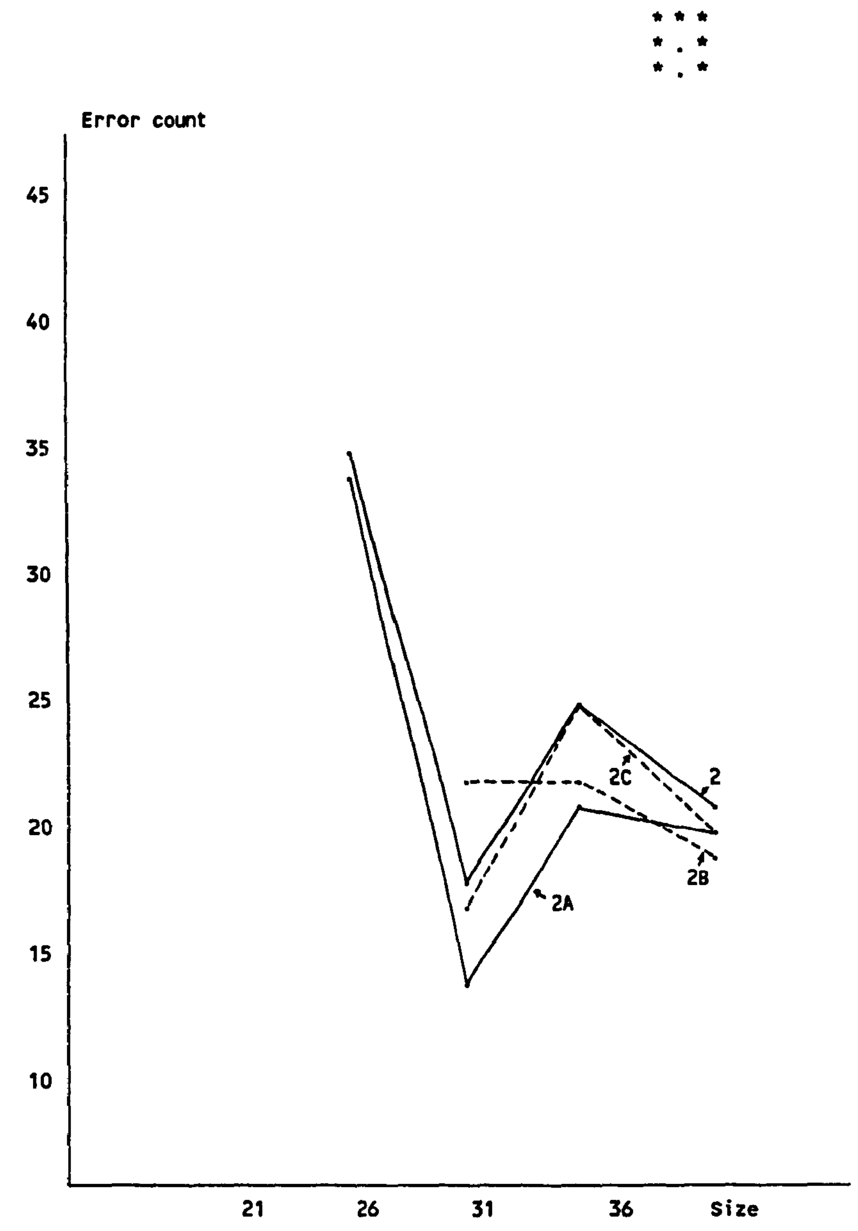
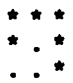


Fig. 3.13 : Error count of Set 3



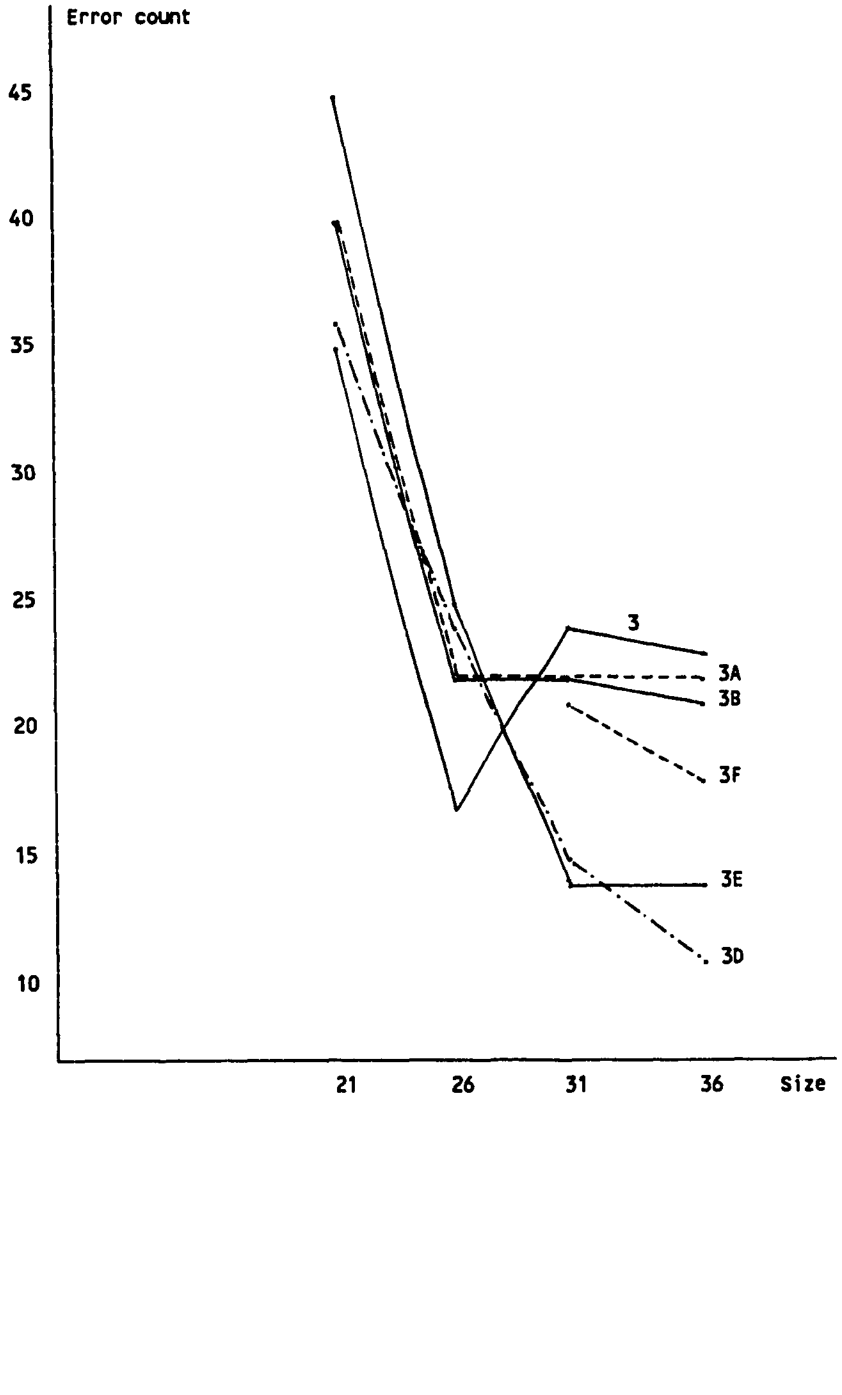
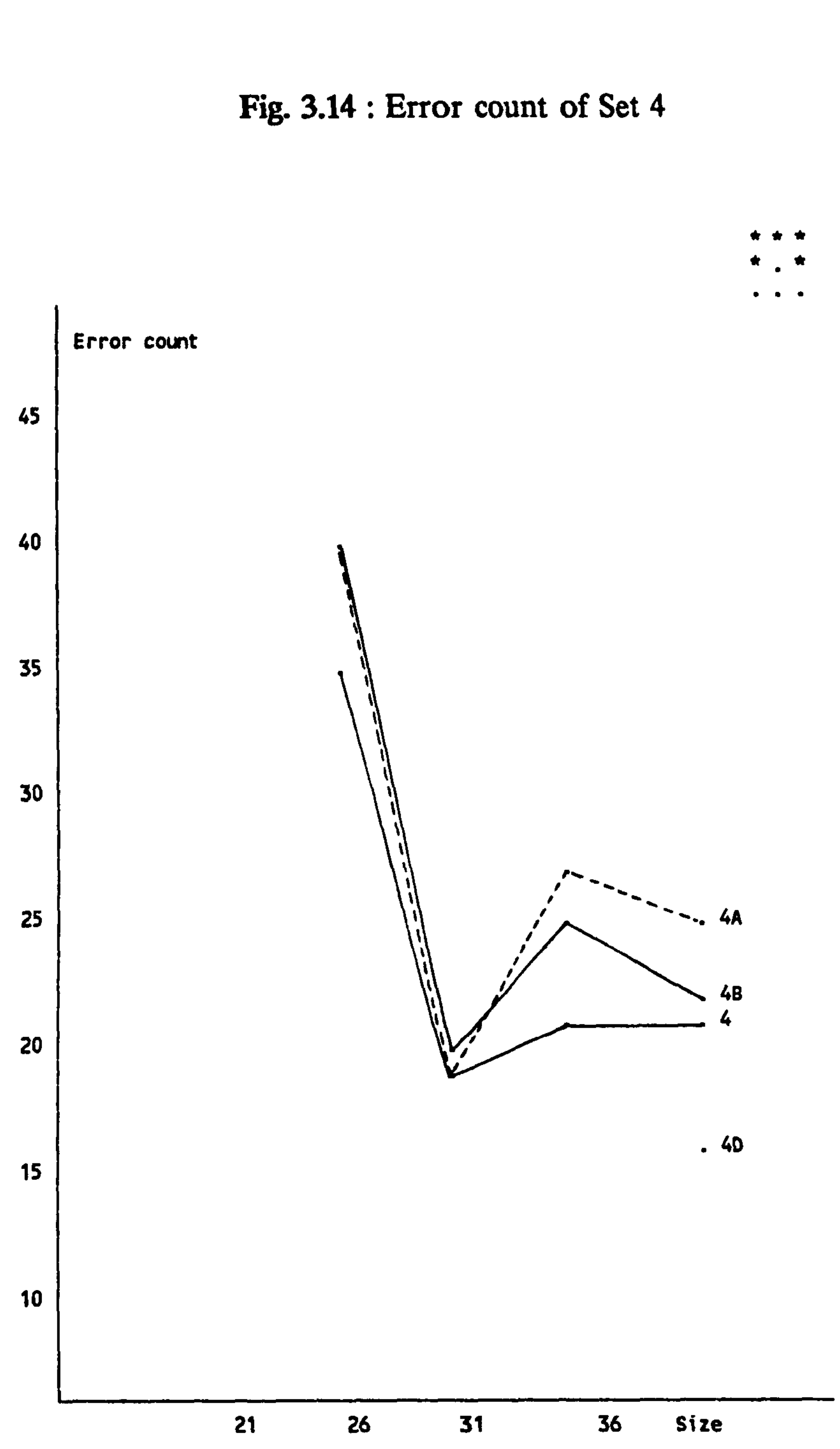


Fig. 3.14 : Error count of Set 4



3.5.2 Result Analysis

The calculated 'Error Count' is subjective and not very accurate, because:

- It is not certain whether the weight factors are correctly assigned.
- Human errors and inconsistencies may occur when a skeleton is examined to decide whether it contains errors or not. If there is an error, the severity of the error cannot be decided quantitatively.

Bearing in mind this limitation, the following observations are only relatively conclusive:

a/ Effects of image size: From Table 3.10, it is found that for a certain degree of smoothing, there is an optimum size. For example:

+ With no smoothing (Set 1), the best size is about 26x26,
+ With more smoothing, the minimum error count is obtained using full size (Set 3D, error count = 11).

b/ Effects of applying the same rule for both black and white pixels (symmetry of smoothing): In Tables 3.9 and 3.10, comparing the error counts between Sets 2 and 2C (error counts 18 and 17), Sets 3 and 3B (error counts 17 and 21), Sets 4 and 4B (error counts 19 and 20), where in the first set the rules for black and white pixels are the same, and in the second set the rules for 2 types of pixels are different, we do not see any real advantage in applying the same rule for both black and white pixels. At high degree of smoothing and the 2 types of pixels are treated the same (Set 3F), the image is smoother, but because of expansion of white areas, thin places in the image become broken.

c/ Effects of propagation on smoothing: Propagation increases the degree of smoothing. At a large image size, its effect is not very noticeable, but at a smaller size, it may overfill enclosed areas and create more L-Type noise.

d/ Effects of connectivity restriction: At first, it was thought that by introducing the connectivity requirement, smoothing would be more discriminatory. But when removing the restriction, the smoothing still gave surprisingly good skeletons at larger image sizes (set 3D which did not apply the connectivity requirement had error count of 11, while set 3 using the same other parameters, and applied the connectivity requirement, had an error count of 17). The connectivity requirement also adds extra computing cost to smoothing.

3.5.3 The optimum solution

From the solution space obtained from different sets of experiments, there are 2 areas which give lower error count:

+ Smoothing method 2A with image size 26x26 (error count = 14).
+ Smoothing method 3D with original image size (error count = 11).

However the second solution is a better choice than the first because:

+ Variation in this area is not very sensitive, so there is less chance of error.
+ A bigger skeleton is more accurate and gives more room for further processing.

To minimize the L-Type noise of the second solution, we used smoothing method 3D for image with size > 24. When filling white hole and the maximum size of image was smaller than 24, method 2A was used (ie. if there are 2 white pixels in the 3x3 window, one white in the center, the center white will be switched to black with propagation).

3.5.4 Conclusions on preprocessing steps before thinning

There are an essential number of features or special points that should be present in the skeleton. Having more than the optimal number of features leads to unwarranted complication in recognition, having less will increase misrecognition.

Smoothing the image by size reduction or by inspecting a window and changing the color of a pixel will help to create a skeleton with the essential features and minimum amount of noise.

Although the measurement of errors in all the experiments were not accurate, the results strongly suggested that size reduction is not as good as edge smoothing. Hence finally method 3D was selected as the preprocessing method before thinning for samples with size > 24x24 and method 2A (where smoothing is more limited than 3D) was selected for samples with a smaller size.

3.6 THINNING

The thinning algorithm used in this study was developed by Naccache and Shinghal at Concordia University [NS84]. It is noticed that in some cases, this method gave excessive erosion and the skeleton sometimes was not in the center of the original strokes. However in general, satisfactory skeletons were obtained. Since the main thrusts of this study are on feature extraction and recognition, no special effort was made to find the best method that gives the highest speed or the most representative skeletons.

After thinning, the skeletons were analyzed to locate the special points: end points, fork points and cross points; these points served as reference points for all subsequent processing: simplification of skeleton and feature extraction.

3.7 SIMPLIFICATION OF SKELETON

It is possible to reduce the number of special points in the skeleton to achieve a simpler representation and therefore reduce the number of pattern classes that the recognition algorithm has to handle:

3.7.1 Joining 2 neighboring end points

If 2 end points are close to each other and there are no other special points in the vicinity, they are joined together by an approximate straight line. This is implemented by inspecting a window with horizontal and vertical distance of (0.10 * the diagonal dimension of the image) from each end point.

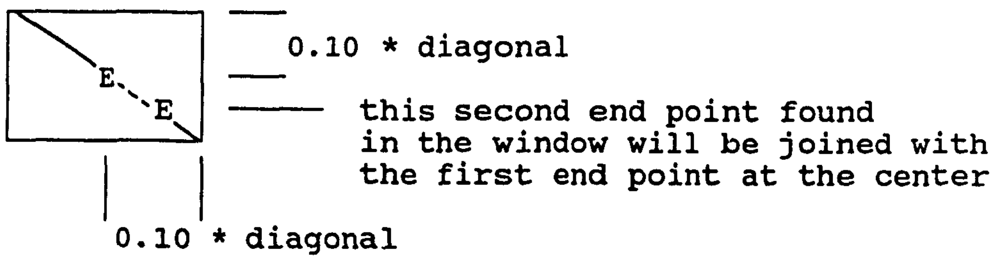

0.10 * diagonal
E
E
this second end point found in the window will be joined with the first end point at the center
0.10 * diagonal

Fig. 3.15 : Window to join 2 end points

This process helps to join broken strokes in numerals 0,2,8.. (Fig. 3.16), but it has undesirable effects when it joins the middle stroke of numeral 3 to the top or the bottom stroke (Fig. 3.17).

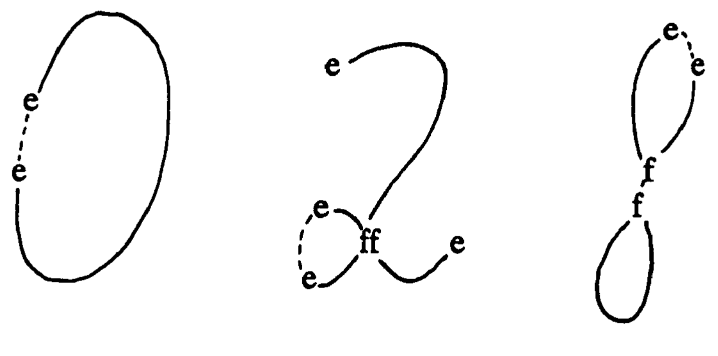


Fig. 3.16 : Joining simplifies skeletons of 0,2,8

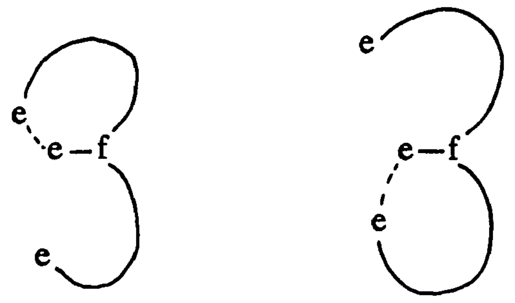


Fig. 3.17 : Undesirable effects when joining neighboring end points

The threshold (0.10 * diagonal) is the maximum that can be used without causing the undesirable side effect on numeral 3 as explained above.

3.7.2 Removing short tails

A short tail in a skeleton consists of a fork point and an end point close to it. A window with horizontal and vertical boundaries equal to (0.07 * the diagonal dimension of the image) from each fork point is investigated. If there is a path connecting the fork point to the end point within the window, the pixels along the path are erased, the end point is removed and the fork point becomes a normal point.

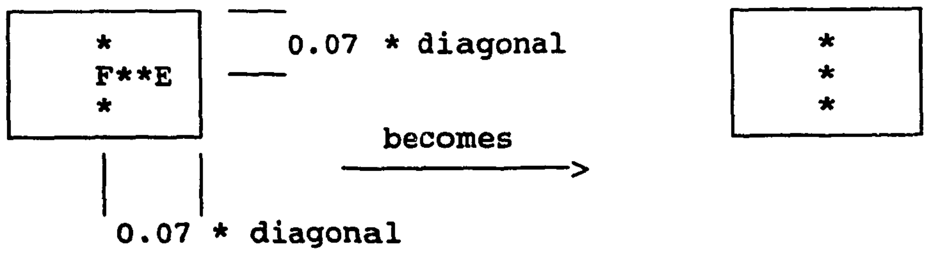


Fig. 3.18 : Removal of a short tail

This operation also has some side effects and removes meaningful strokes, such as the ones illustrated in Fig. 3.19.

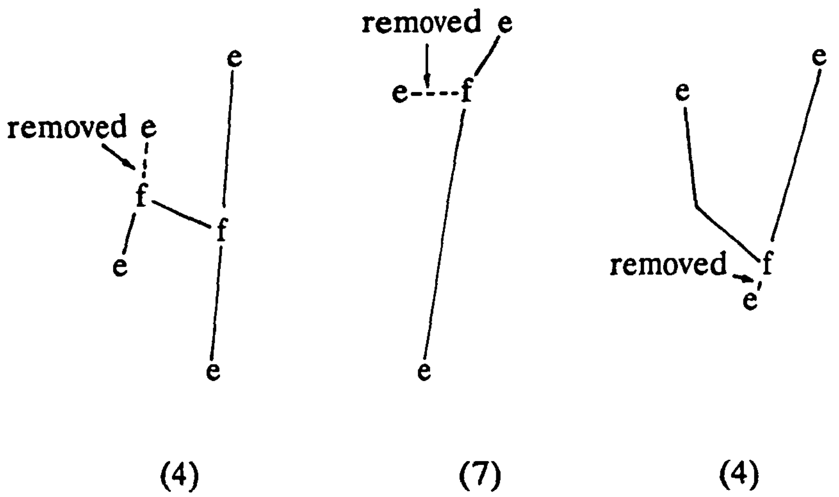


Fig. 3.19 : Side effects of tail removal

The ratio 0.07 of the diagonal has been selected to minimize this undesirable effect.

3.7.3 Removing neighboring forks points

Neighboring fork points are usually caused by white holes in the image. In such cases the skeleton can be simplified. For example, the following simplification can be made:

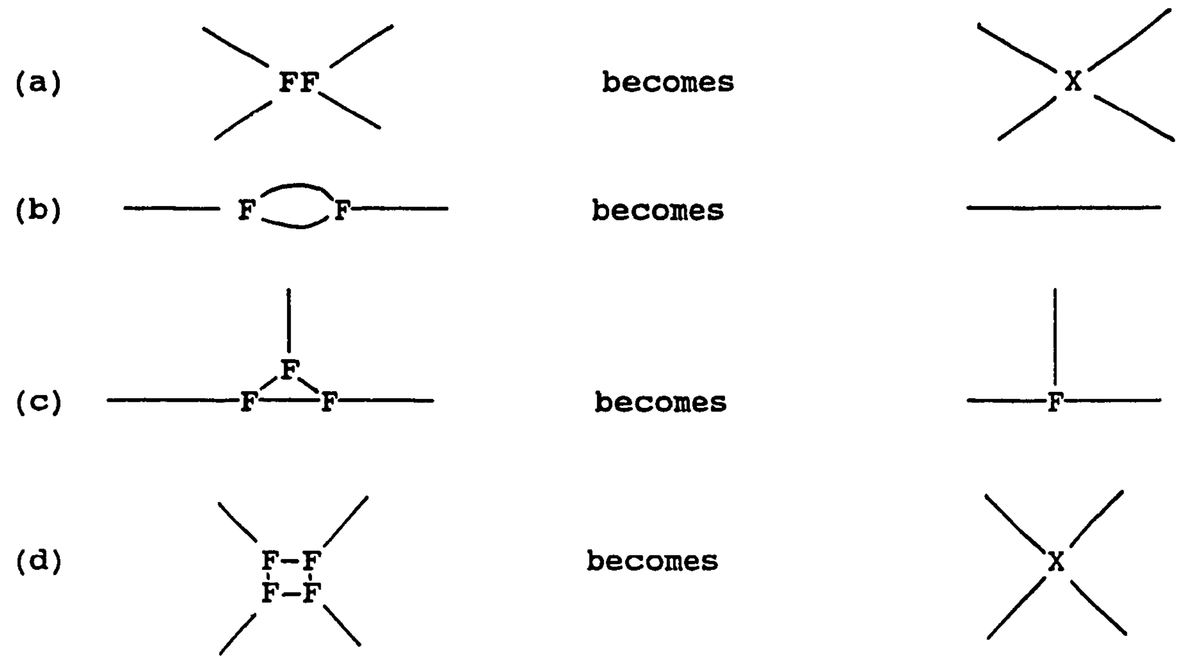


Fig. 3.20 : Simplification of fork points

These simplifications were not implemented in this study because of the following reasons:

- the above types of configuration do not occur very frequently. When they do, there is logic in the feature extraction process to handle the most frequent ones,
- with other factors considered, (such as deciding the threshold below which 2 fork points are considered close to each other), it may turn out that some new confusion will appear and the recognition accuracy may not improve,
- these errors may be specific to the smoothing and thinning method used in this study and may become negligible if a better smoothing or a better thinning method is used,

- programming the method to detect the configuration and to perform the simplifying operation is complicated. It involves tracing all possible paths from each fork point inside a window having the considered fork point at the center. The logic must deal with all possible combinations of special points that may be present in the window (end points, fork points, cross points). The costs in programming and in program execution are high while it is not sure whether much benefit could be gained from this operation.

3.8 THE ROLE OF PREPROCESSING

In this study, more attention has been paid to the use of a knowledge base constructed by the computer under the guidance of a human expert, and study was not done extensively to improve preprocessing.

After the knowledge base has been fully developed, stable recognition results were reached and the recognition errors were inspected in detail, it is felt that a sophisticated preprocessing method is probably a very effective way to transform an unconstrained and poorly written character into a properly formed character so that finally a good skeleton can be obtained. In addition to using the knowledge base in the classification stage, expert modules can also be attached to the preprocessor to identify a familiar configuration and give appropriate thinning tailored to that configuration.

A recent study by **Brown et al** [BFW88] has treated preprocessing extensively. The authors introduced a number of new concepts and techniques to obtain good skeletons that retain maximum features of the original patterns, such as:

- Implementation of easy skeleton simplification like configuration (a) in

Fig. 3.20.

- Stroke concept and uniform erosion to avoid thinning a filled loop into a straight stroke (the concept of 'unthinnable region').
- Using expert modules to handle special configurations.
- Selective removal of spurious tails: tails are considered for removal only when they occur at some designated areas in the pattern image.

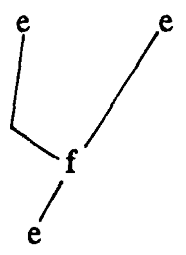
4. FEATURES AND FEATURE EXTRACTION

4.1 FEATURE GENERATION BY COMPUTER

One of the desirable goals is to automate feature selection and extraction as much as possible. The computer can make measurements on the pattern and store the features as a character array, where each character represents a measurement or the presence of a feature. In order to explore the effectiveness of features generated by computers, some experiments were conducted using various types of features which can be observed by the computer without human intervention.

4.1.1 Sequence of special points

The skeleton from a numeral was scanned from top to bottom, left to right, and all the special points were recorded in a 'feature string', for example, the feature string of the following skeleton is 'EEFE'



This feature string has a close relation with the position of the special points and the number of strokes in the numeral. Using this feature string, from a training set of 8500 samples, the computer automatically generated a knowledge base of 87 features. This knowledge base was used to recognize 530 unknown samples. The recognition results were:

Recognition	64.91%
Substitution	34.53%
Rejection	0.57%
Reliability	65.28%

This feature is very simple and need no human intelligence or knowledge, the approach is the same as neural networks, yet it achieved the same recognition rate as neural networks (33-78%) as reported in [PLHS88] with much less computing cost.

4.1.2 Sequence representing character profile

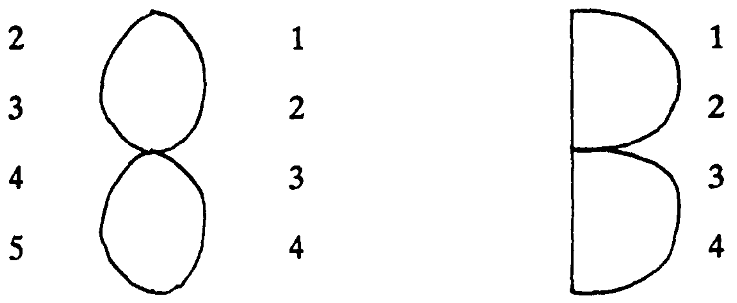
Profiles are also very representative of a numeral. Profile features are not affected by white noise inside a stroke, and extracting profile features needs less computing time than thinning because the computer considers only the pixels at the edges of the numeral in 1 pass, while thinning must go through all black pixels in many passes.

To implement machine learning using profile features, a profile can be represented by the sequence of its direction from top to bottom. For example, consider the right profile of a 3:

```
\ 1
/ 2
\ 3
/ 4
```

If we represent the stroke to the right by an odd number, stroke to the left by an even number, and each change in direction by an increment in the number, then the right profile of 3 will be represented by '1234'.

We can obtain the profile string for the left and right profiles, and also represent the variation in width of the image in the same way (Fig. 4.1). The width is defined as the horizontal distance between the leftmost and rightmost black pixels of the character.

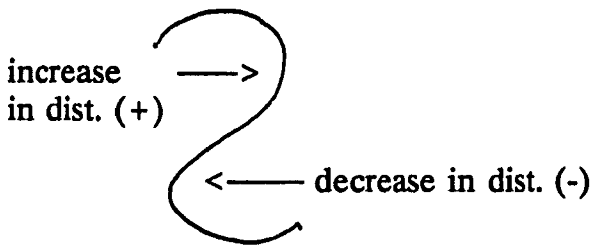


Left profile = '2345'
Right profile = '1234'

Width profile = '1234'
representing the variation
in width of numeral 8

Fig. 4.1 : Profile strings of numeral 8

In addition to the edge direction, the jump detected on the edge can be added to the profile string. A jump is represented by a '+' if there is a sharp **increase** in the distance from the numeral edge to the border of the rectangular frame enclosing it. It will be represented by a '-' if there is a sharp **decrease** in the distance from numeral edge to the frame border. The jump representations are superimposed to the direction profile to give a composite profile, as shown:



Left profile = '23'
Right profile = '12'

with the jump superimposed:

Left profile = '2+23' (the + jump comes in the middle of the direction represented by 2)
Right profile = '12-2' (the - jump comes in the middle of the direction represented by 2)

Fig. 4.2 : Profile strings containing jump representation

The computer was used to extract the profile features and build the knowledge base from a training set of 1000 samples. Then the knowledge base was used to recognize the same set, the recognition rate achieved was 70.7%.

4.1.3 An alternative method using profile features

In the above methods using profile strings, the direction of the numeral edge was used. But the string did not include the information of the relative lengths of each edge segment, therefore different profiles could be represented by the same string. As an illustration, the following 2 numerals have the same right profile string '1234' (Fig. 4.3)

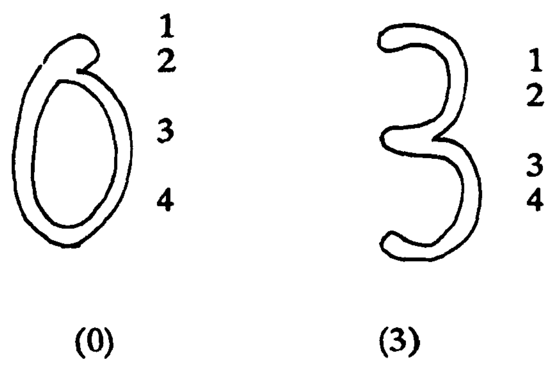


Fig. 4.3 : 2 different profiles represented by the same string '1234'

Also, the fluctuation in the profile direction would give a complicated profile string without capturing the true profile feature (Fig. 4.4)

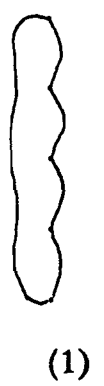
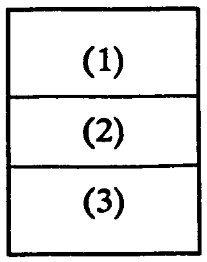


Fig. 4.4 : Fluctuation in profile direction that does not represent the true profile feature

To overcome the above deficiencies, and also to make the feature extraction faster, another scheme of profile representation was attempted, in which both the profile and the position of the jumps were quantified with discrete measurements over equal portions of the character's length. The width of the characters and the presence of holes inside the character were also measured. This was done as follows:

The image is divided into 3 equal horizontal bands as shown:



For each band the following measurements are made:

- Average distance of the left edge of the character from the **left** border (L).
- Presence and direction of a **jump** in the **left** profile.
- Average distance of the right edge of the character from the **right** border (R).
- Presence and direction of a **jump** in the **right** profile.
- Average **width** of character (W).
- Presence and direction of a sharp **change in the width** on each row.
- Average width of the white **hole** inside the character (H). The width of a hole at a row is defined as the number of white pixels that are enclosed between the leftmost black pixel and the rightmost black pixel of the image on that row.
- Presence and direction of a sharp **change in the width of the hole.**
- The **ratio of maximum width/length** of the character over its entire length (this feature is helpful in recognizing numeral 1).

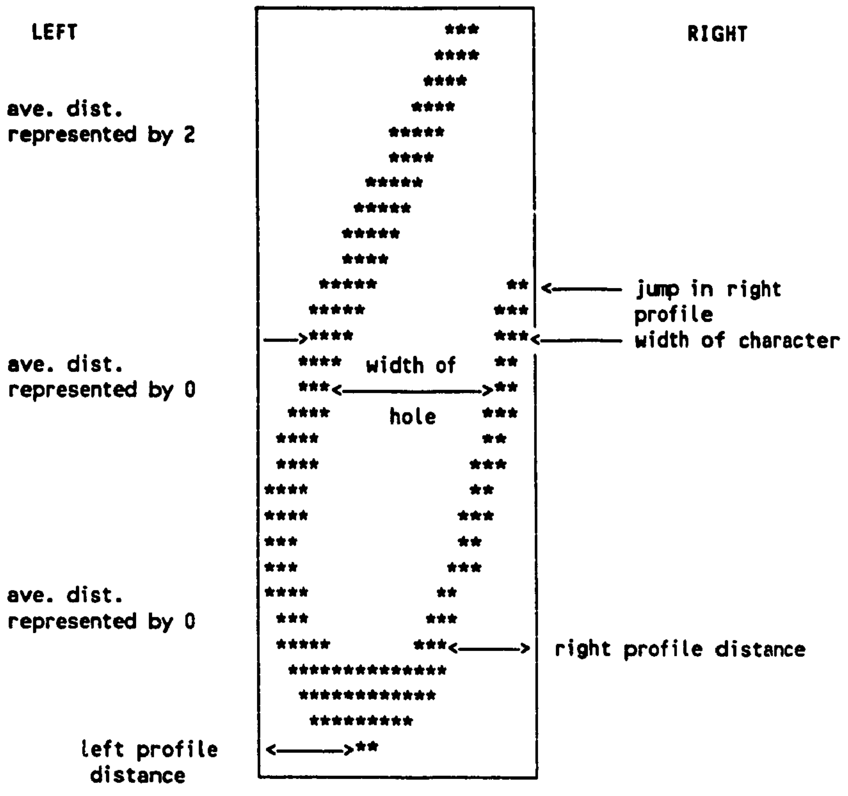
The average distance is quantified into a number (0,1,2) using 2 thresholds. For example, assume that the thresholds used are: T1 = 0.2 * the width of the image and T2 = 0.4 * the width of the image. If average distance is less than T1, it is represented by the digit 0, if it is less than T2 and greater than or equal to T1, it is represented by the digit 1, if it is greater than or equal to T2, then it is represented by digit 2.

The presence of a sharp change (jump) is represented by a '+' or a '-' depending on the direction of the change. There is a 'jump threshold' to decide whether a jump is detected or not.

The ratio of maximum width/length is represented by a letter where:

Rounded Ratio =	0.1	Letter =	'A'
	0.2		'B'
	... and so on.		

With the above representations, a sample has 5 features which are represented by 5 feature codes. Each code is a string of 6 characters, the first 3 represent the distance measurements. In the last 3 positions, a presence of a '+' or '-' represents a jump in the corresponding band. An example of the 5 feature codes is given in Fig. 4.5.



Left	'200 '
Right	'101 - '
Width	'022 + '
Hole	'022 + '

Ratio max. width/length = 0.6 represented by 'G '

Fig. 4.5 : Measurement and generation of feature codes by computer

By changing the thresholds, it is possible to obtain a different number of features in the knowledge base.

Variation of distance threshold

Since a measurement is assigned a number 0,1 or 2, we try to have the assignments to be distributed equally into 0,1 and 2 while keeping the jump threshold constant. When the assigned numbers are evenly distributed among the 3 ranges, there are more combinations of feature codes. It is hoped that by increasing the number of feature codes (i.e. having more features), the number of pattern classes also increases, and classification (recognition) will be more accurate. The iterations using different threshold values are illustrated in Table 4.6. The number of recognition errors on a training set of 1000 samples is also given for each iteration. The maximum recognition rate was about 96.5%.

Table 4.6 : Iteration by changing distance threshold values

Sample size = 1000

Run no.	Number of feature codes						No. of errors	Recog. rate
	Left	Right	Width	Hole	MW/L Ratio	TOTAL		
1	208	138	273	71	17	707	54	94.6
2	208	148	273	135	17	781	45	95.5
3	208	148	269	177	17	819	37	96.3
4	208	156	268	187	17	836	36	96.4
5	208	141	265	222	17	853	42	85.8
6	208	153	244	214	17	836	45	95.5
7	208	155	270	214	17	864	38	96.2
8	208	154	270	214	17	863	37	96.3
9	208	154	268	214	17	861	38	96.2
10	208	155	271	214	17	865	35	96.5
11	208	155	267	214	17	861	40	96.0
12	208	155	266	214	17	860	36	96.4

The knowledge base from the set of thresholds in the last iteration (Run number 12) was used to recognize a test set of 1000 samples, the number of errors was 227, recognition rate was 77.3%.

The recognition rate on a larger training set (2000 samples) was nearly the same as with a smaller set of 1000, being 94.6% (107 errors). However, the total number of observed features increased from 860 to 1158 (35% increase).

Variation of the jump threshold

The threshold to recognize a jump should also affect the number of observed features and the recognition rate. In the following experiments, the threshold of a jump is = threshold ratio * the diagonal dimension of the image. Results on a training set of 2000 samples are shown:

Table 4.7 : Variation in jump threshold

Threshold ratio (based on diagonal dim.)	No.of feat.codes	No. of errors	Recog. rate
0.15	1167	10	94.6
0.20	1012	147	92.6
0.25	855	207	89.6
0.30	702	262	86.9
0.35	566	324	83.8

Investigation of recognition rate of a test set

By changing thresholds, it is possible to have more features and better recognition rates with a training set. However, the real test of a recognition method is its recognition rate on a set of unknown samples. Various runs were done using different jump thresholds on a training set of 2000 samples and a test set of 2000 samples. The results are summarized in Table 4.8.

Table 4.8 : Recognition rates using different jump thresholds

Run no.	No. of feature codes	TRAINING Subst. error	TRAINING Reject	TRAINING TOTAL	TRAINING %	TESTING Subst. error	TESTING Reject	TESTING TOTAL	TESTING %
1	139	504	0	504	74.5	627	11	638	68.1
2	240	464	0	464	76.8	625	14	639	68.0
3	1167	107	0	107	94.6	333	134	467	76.6
4	1265	104	0	104	94.8	358	137	495	75.2
5	1217	110	0	110	94.5	338	137	475	76.2
6	1110	118	0	118	94.1	344	127	[illegible]1	75.4
7	1172	97	0	97	95.1	324	140	464	76.8
8	999	148	0	148	92.6	340	116	456	77.2
9	859	201	0	201	89.9	395	92	487	75.6
10	1309	94	0	94	95.3	331	159	490	75.5
11	1120	122	0	122	93.9	341	114	455	77.2
12	1017	149	0	149	92.5	370	116	486	75.7
13	1219	101	0	101	94.9	337	140	477	76.1
14	909	175	0	175	91.2	388	92	480	76.0
15 *	711	168	0	168	91.6	385	113	498	75.1
16 *	740	164	0	164	91.8	386	148	534	73.3

Note: Runs 1-14 : 3 jump features on 3 equal horizontal portions.
* Runs 15-16 : 2 jump features on 2 equal horizontal portions.

The relationship between the number of features codes and the recognition rate in training and testing is plotted in Fig. 4.9.

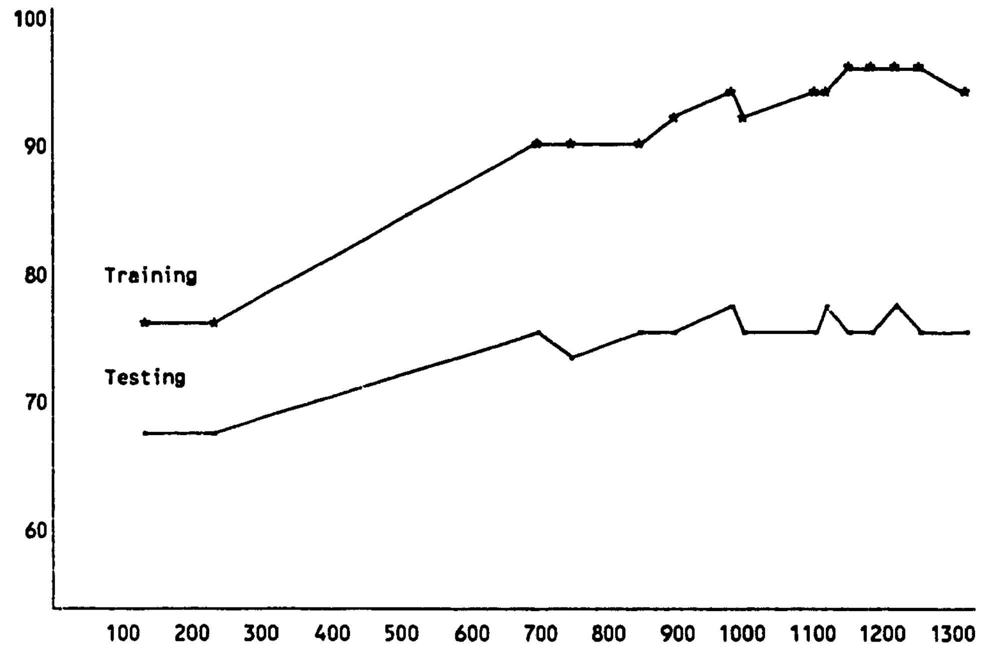


Fig. 4.9 : Relationship between recognition rate and the number of features codes

4.2 CONCLUSIONS FROM TEST RUNS USING COMPUTER GENERATED FEATURES

4.2.1 Human intelligence is an important element in feature extraction

From all the runs, the maximum rate that could be achieved using computer-generated features was about 75% for a test set, although it was possible to get a recognition rate of 95% for a training set.

The results point to the fact that the features generated by computer are dependent on the training samples and do not capture the truly discriminating features of the numerals. To have discriminating features, special relationships between different measurements on the numeral must be identified. This task requires human intelligence.

4.2.2 Requirements for a feature

Consistent and general: a feature should be applicable to all samples of the same numeral. The difference in recognition rate of the training set and a test set is an estimate of the generality of the features. Using general features, the recognition rate for the test set should not be much different from that of the training set.

Informative (meaningful): a feature is easily perceived and understood by human. Features obtained from mathematical transformation are often not meaningful in human perception.

Discriminative: a feature should help to resolve the confusion between different numerals.

Useful: given a possible set of hypotheses, the application of a new feature must produce a new set of hypotheses which is smaller than the previous one and some numerals within the set should become more dominant. The term **hypothesis set** of a feature refers to a set of likelihood measurements for the 10 numerals having that feature in the training set. Hypothesis set will be more fully discussed in Chapter 5.

4.3 COMBINATION OF COMPUTER-GENERATED AND HUMAN-BASED FEATURES

The method implemented in this thesis is a combination of computer generated features and features derived from human perception. Both the original image and the skeleton were used in feature extraction. The strategy of using both the skeleton and the original image is fairly new and was used in only a few recent works on character recognition. In Kuan and Srihari's work [KS88], the image was decomposed to strokes using run length analysis techniques, and the contour profile information was used as auxiliary features. In Mitchell and Gillies's work [MG88], the skeletons and 'flesh regions' were used in feature extraction.

We distinguish 2 types of feature:

- The computer generated feature is the **primary feature** used for the coarse classification of numerals, it is a feature code consisting of characters representing the special points in the skeleton that are found in a scan from top to bottom, left to right. For example the following numeral has the feature 'EEFE':

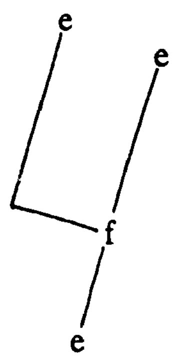


Fig. 4.10 : Primary feature

- A **secondary feature** originates from human perception. It is called secondary feature because it always goes together with one primary feature. For example:

  The primary feature is 'EEFE' (feature 12). If there is a straight stroke on the right hand side (as in Fig. 4.10), then the numeral has feature 104.

Each primary feature has many secondary features, but each secondary feature has one and only one primary feature. Because secondary features are dependent on primary features, the feature extraction process can consider first the primary feature and then considers only the secondary features that are dependent on it. Features are only extracted when needed.

A complete list of primary and secondary features is given in Appendix 1.

4.4 PURPOSE AND USE OF FEATURES

A feature has the purpose of increasing the likelihood of one candidate while decreasing the likelihood of others. The features therefore have one or more of the following effects:

- They produce some **coarse discrimination** by reducing a large hypothesis set into a smaller set where the likelihoods of its members are more discriminative than in the larger set.
- They produce **fine discrimination** to arrive at the final recognition by creating either a hypothesis set with one or more dominant numerals, or a hypothesis set that does not contain one or more numerals.

A feature may also have a side effect, i.e. it may decrease the likelihood of a candidate which is the true recognition candidate. For example, if we just look for a profile of 3 on the right hand side, the following numeral may have a hypothesis set where 3 is dominant, and may be classified by the computer as numeral 3.

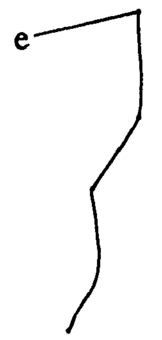


Fig. 4.11 : Side effect of features

However, before applying the rule that uses the profile of 3, if we specify a precondition that the rule is applied only if the stroke on the right is not relatively straight; then, the final hypothesis set will be more dominant in 7 and return 7 as the result.

To reduce the side effects:

- the knowledge base must have considered a large number of rules, so that the features are used in a balanced manner,
- a feature that has strong discriminative power but found to give a skewed hypothesis set must be balanced with preconditions, or in some cases, must be discarded.

Where preconditions exist, the features related to a precondition must be examined in a sequential order.

4.5 GUIDELINES FOR THE EXTRACTION OF FEATURES

Writer dependency

The features as well as the noise in the handwritten characters are in some way related to the writer's habits. An analysis of these relationships is outside the scope of this thesis, however writer's habits were considered when the features were selected. For example:

- When reading or writing, the sequence is from top to bottom, left to right. For left-handed people, the local sequence in forming a single character may be from right to left, but the overall movement of characters is also from left to right, and reading also follows the same sequence. Therefore, the special points in the skeleton (end points, fork points, cross points) were scanned in the order top to bottom, left to right:

```
——————————————>
——————————————>
——————————————>
```

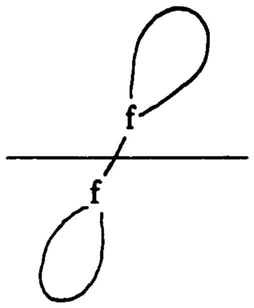
  instead of the sequence:

```
|  |  |
|  |  |
|  |  |
|  |  |
v  v  v
```

- People have different slants in writing, therefore, the left and right profiles were considered, but not the top or bottom profile, because the top and bottom profiles are affected by slants. For example the right profile was used to recognize numerals 3,5,8:

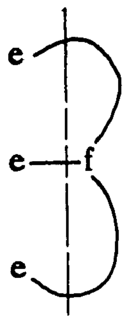
(3) (5) (8)

- Because of the effect of slants, intersection of a numeral is effective in the horizontal direction, but not in the vertical direction. In the following illustrations, the horizontal intersection will help to recognize numeral 8; on the other hand, the vertical intersection for numeral 3 is meaningful, but not effective, it may be effective only if the numeral is rotated to offset the slant, as shown below:



(8)

No. of intersections = 1



(3)

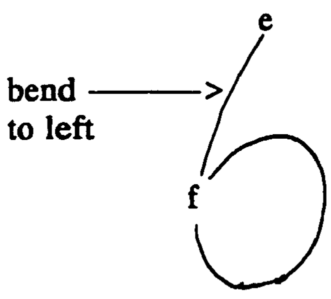
No. of intersections = 3

4.6 DESCRIPTIONS OF DIFFERENT TYPES OF SECONDARY FEATURES

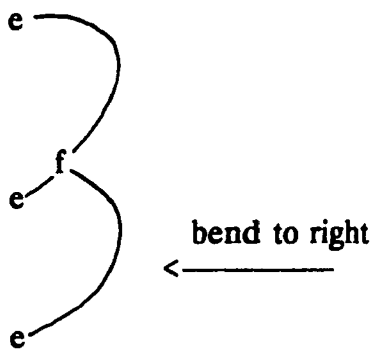
The features used in the knowledge base can be grouped into different types:

4.6.1 A bend between 2 points (BE)

This feature is a curvature between 2 special points, for example a stroke of 2 or 6 before a fork point, or a stroke of 3 after a fork point.



(6)

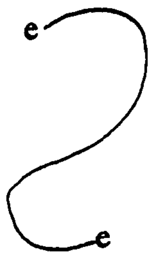


(3)

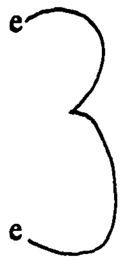
4.6.2 Horizontal curve at top or bottom (CV)

Several numerals have curves connected to an end point at the top or bottom, coming from the left or right direction. For example:

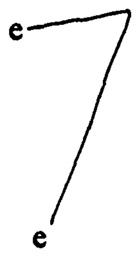
- Numeral 2 has a curve at top right and a curve at bottom left,
- Numeral 3 has a curve at top right and a curve at bottom right,
- Numeral 5 has a curve at top left and a curve at bottom right,
- Numeral 6 has a curve at top left,
- Numeral 7 has a curve at top right,
- Numeral 9 has a curve at top left or top right.



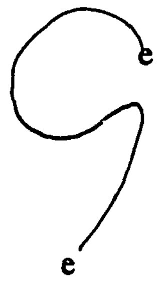
(2)



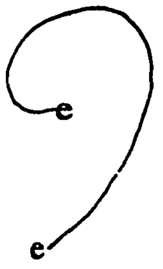
(3)



(7)



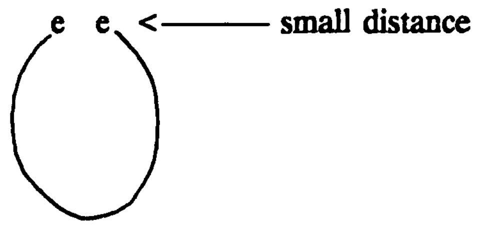
(9) curve at top left



(9) curve at top right

4.6.3 Distance between 2 points (DI)

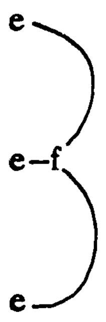
The distance between 2 special points is a general feature that was used frequently, for example to identify broken strokes in a numeral which has 2 end points near each other as shown below:



(0)

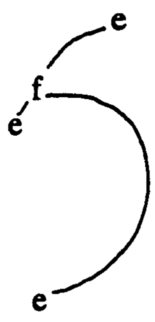
4.6.4 Direction of a stroke from a special point (DR)

One application of this feature is to distinguish 2 numerals 3 and 5. From the end point at the top, numeral 3 has a stroke heading right, while numeral 5 has a stroke heading left.



(3)

stroke from first end point heads to right

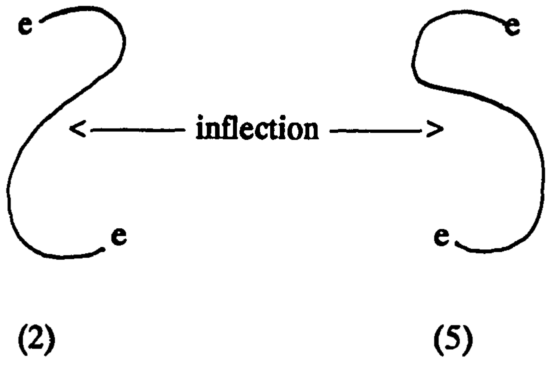


(5)

stroke from first end point heads to left

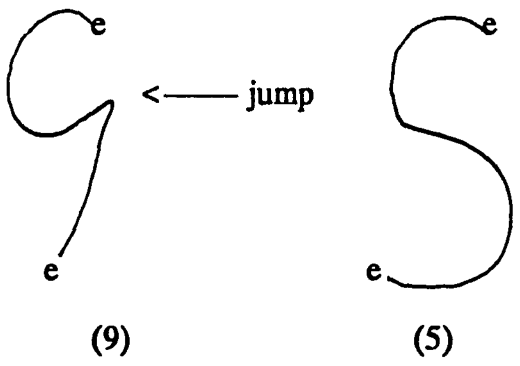
4.6.5 Inflection (IN)

An inflection is a change of direction in a stroke joining 2 special points from top to bottom. This feature is present in numerals 2 and 5 as shown below:



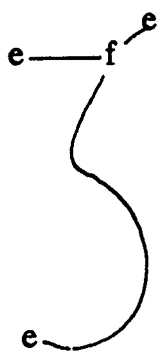
4.6.6 Jump at left or right profile (JP)

A jump is a discontinuity in the profile of a numeral. For example, a high jump on the upper right hand side helps to distinguish numerals 9 and 5.

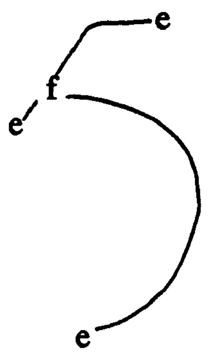


4.6.7 Comparing lengths between 2 strokes (LE)

Example: the relative lengths of the strokes connecting the fork point to the 2 top end points help to distinguish between 3 and 5:



(3)
Fork point nearer to right end point

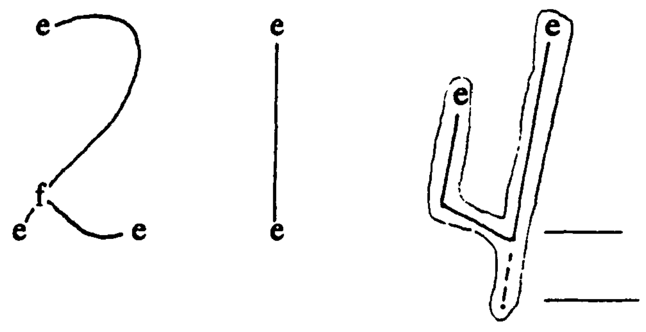


(5)
Fork point nearer to left end point

4.6.8 Location in numeral frame (LO)

The location of a special point with respect to the top, bottom, left, right borders can help with recognition. For example:

- A fork point of numeral 2 is usually nearer to the bottom border,
- Numeral 1 has 2 end points on or near the top and bottom borders,
- Numeral 4 may have the bottom tail removed by preprocessing. In such case, the lowest point of the skeleton is far from the bottom border of the original image.



(2) (1) (4)

the lowest point of the skeleton is far away from the bottom border of the original image

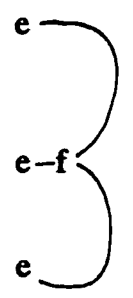
4.6.9 Profile on the left or right (PR)

The profile features (described in Section 4.1.2) help to identify 3,5 (right profile), 6 (left profile), 0,8 (left and right profiles) ...

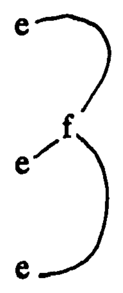
4.6.10 Relative position between 2 points (RP)

Examples where the relative position of special points is used for recognition:

EEFE or EFEE : numeral 3 usually has a fork point on the right of all end points.



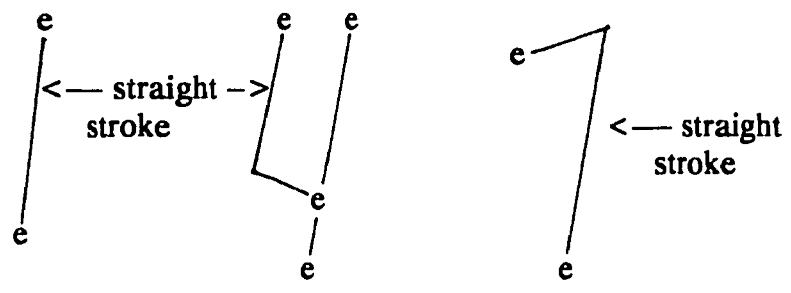
(3)



(3)

4.6.11 Straight stroke between 2 points (SS)

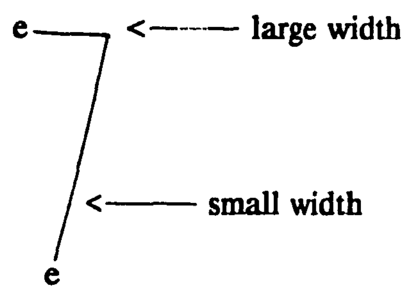
This feature was used to identify a numeral that has a straight segment in its skeleton, for example numerals 1,4 or 7.



(1) (4) (7)

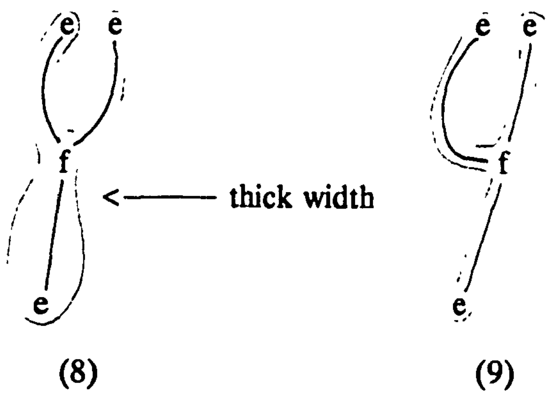
4.6.12 Width of numeral at different rows (WI)

For example, a large width at the top and a small width at the bottom help to identify numeral 7:



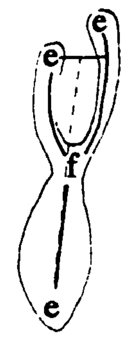
4.6.13 Width of stroke (WS)

The width of a stroke is helpful to identify the presence of a loop which is filled during writing or preprocessing, for example in numerals 6 or 8. In the following illustration, the thick width at the bottom stroke distinguishes numeral 8 from numeral 9.

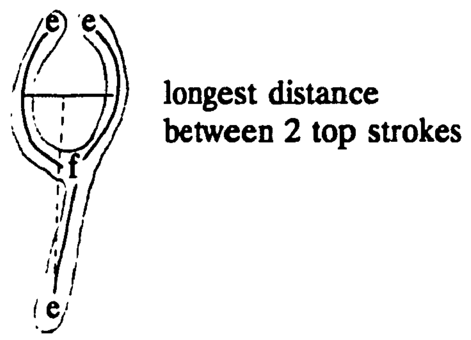


4.6.14 Symmetry (SY)

The symmetry feature helps to distinguish numeral 8 from 9, or numeral 8 from 6, as illustrated in the following example:



(8)
symmetrical



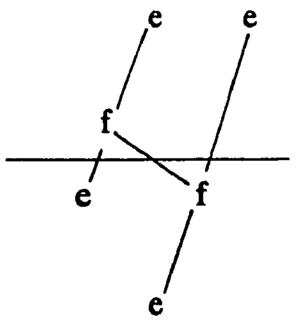
(9)
not symmetrical

A center point is identified at the top half of the numeral, it is the middle point of the longest distance between the 2 strokes at the top half. An imaginary line between this center point and the bottom end point is drawn. If the fork point is close to this line, the numeral is considered to be symmetrical in shape.

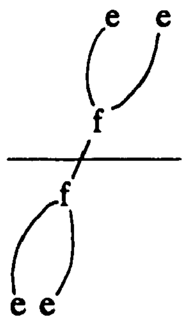
4.1.15 Horizontal intersection at a row (X)

The number of intersection points between an imaginary horizontal line and a numeral can help to distinguish one numeral from another. For example:

- Distinguishing 4 and 8 (having primary feature EEFFEE)

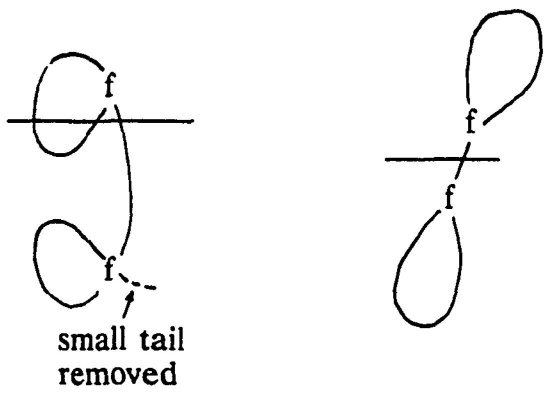


(4) intersections = 3



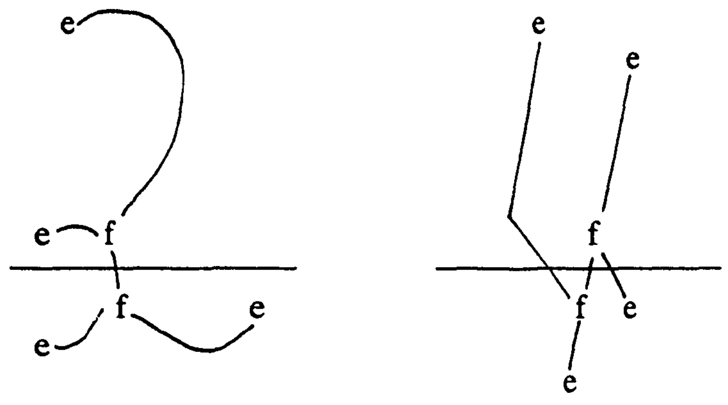
(8) intersections = 1

- Distinguishing 2 and 8 (having primary feature FF)



(2) intersections = 3 (8) intersections = 1

- Distinguish 2 and 4 (having primary feature EEFFEE)



(2) intersections = 1 (4) intersections = 3

4.6.16 Combination of above features (CO)

This is a composite feature that combines more than one of the above listed features.

4.7 PRECONDITIONS AND DEPENDENCY OF FEATURES

As an illustration, the feature string 'EE' which occurs most frequently is analyzed for dependency. The 63 secondary features dependent on the feature string

'EE' are grouped into different levels, all features in the same level can be considered simultaneously, provided that all the features at the higher levels have been considered. From the dependency relationship, we can see that with parallel processing, feature extraction can be done in 6 steps, the maximum number of parallel processes in a level is 17 for this example. With sequential processing, 64 steps are required to extract the features.

Table 4.12 : Dependency of features

NOTE: Notation 184 (22,65,140) means that feature 184 is dependent on 3 features 22,65,140. This notation is not used in level 2, but implicitly all features in level 2 are dependent on feature 4 in level 1.

Level	1	2	3	4	5	6
F	4					
E		5				
A			136 (5)			
T			140(5,13)			
U				184 (22,65,140)		
R				357 (13,22,40,73,140)		
E					391 (103,357)	
S			352 (5,38)			
		6				
			13 (6)			
				26 (13)		
					48 (26)	
						257 (48,52)
					49 (26)	
					171 (6,26,40,49)	
					182 (6,26,40,49,133,140)	
						206 (31,73,182)
						368 (182)
					214 (6,26,31,40,49)	
						229 (214)
						230 (214)
						265 (49,52,151)
						284 (48,49)
						321 (49,151)
				31 (13,22)		
					390 (31,73)	
				384 (13,22,133)		
				385 (13,14,255)		
			131 (6)			
				151 (6,22,38,131,133,140)		
					294 (151)	
				160 (38,131)		
				183 (38,131)		
					275 (22,65,140,183)	
				350 (5,38,131,133)		
				353 (131,133)		
					372 (352,353)	
		14				
			40 (14)			
				83 (5,22,26,38,40,65,73)		
				84 (5,22,26,38,40,65,73)		
					411 (83,84,131,133)	
				103 (5,40)		
					233 (103)	
				287 (40)		
					317 (287)	
				351 (6,40)		
		22				
			409 (22)			
			410 (22)			
		38				
			318 (38)			
		52				
			270 (52)			
		65				
		73				
			74 (73)			
				309 (13,74)		
		133				
			406 (133)			
		255				
		258				
		303				
		388				
Count	1	13	12	17	13	8

4.8 DIFFICULTIES IN FEATURE SELECTION

4.8.1 Feature selection is data-dependent and time-consuming

During training, features were added to the knowledge base using 3 criteria:

- The human expert judged that the rule for the feature is general and applicable to the majority of cases.
- The feature had to produce an improved recognition rate as expected. If the improvement was not as expected, for example when a feature was added with the purpose of resolving confusion between numerals 2 and 3, but the recognition results showed that the improvement actually occurred for some other pair, for example numerals 4 and 9, then the improvement was just a coincidence, and the effect of the added feature had to be examined carefully.
- The added feature should not be in conflict with existing features, i.e. it should not cause substitution errors that have been resolved by some previous features. If such substitution errors were detected, the conflict had to be resolved by modifying the new feature, or the old feature, or both. This was the most time-consuming task in training, because the conflict might propagate to more than 2 features. Therefore, adding a new feature always involved a great deal of conflict resolution, especially when the knowledge base already contained a large number of rules.

Addition of features was incremental and therefore this process is dependent on the training data. There were many possibilities of selecting a new feature and deciding which of the old features it was dependent on. It is not possible to prove that the growth of the knowledge base during training is an optimum one and the rules are inter-related in a systematic way.

4.8.2 Fuzziness in establishing thresholds for a feature

One of the major problems in pattern recognition is establishing thresholds. This problem appears in all stages of the recognition process. In preprocessing, we use a threshold to erase or to keep a spurious tail, or to join or leave intact a break in a stroke. In feature extraction, all features are based on thresholds. Take for example a broken numeral 0 such as the one shown in Section 4.6.3. If the threshold for the distance between 2 end points is too low, some valid numerals 0 are rejected, but if it is too high, some numerals 6 will be recognized as numeral 0. With statistical analysis, a threshold can be established to minimize the number of errors, but no matter how the threshold is selected, some substitution errors always occur.

Because of this dilemma, even when the character is well-formed and free from noise, there could still be a number of errors caused by having a fixed threshold.

To minimize the problem with threshold, this study used 2 approaches:

1/ For a feature, usually there were 2 rules with 2 thresholds, one to accept the numeral to a pattern class, the other to reject the numeral from a pattern class. For the example above, there were 2 rules:

- Rule 26 : true if (a) the distance between 2 end points is less than 0.3 * the diagonal dimension of the image and (b) horizontal distance between them is less than 0.3 * the width of the image and (c) vertical distance between them is less than 0.3 * the height of the image. This rule was used to recognize numeral 0.
- Rule 184: true if the distance between 2 end points is greater than 0.5 * the diagonal dimension of the image. This rule was used to reject numeral 6 from being misrecognized as 0.

Since there were 2 rules, any sample which did not satisfy the 2 thresholds would fall into a different pattern class and be recognized by other features.

2/ A large number of features were used so that the results of feature matching would interact with one another. Using the inference recognition method, the result of this interaction of features would produce a hypothesis set containing a number of hypotheses. If there is a dominant hypothesis which has a higher confidence level than others, there is a good probability that the hypothesis is correct. If there is no dominant hypothesis, the sample is rejected as being unrecognizable (this will be discussed in detail in Section 5.4).

5. TRAINING AND RECOGNITION

5.1 TRAINING

Each sample from the training set was examined for the primary feature and then matched against all the secondary features that are dependent on the primary feature. The results of training can be represented by a matrix as in Fig. 5.1.

Fig. 5.1: Results of feature extraction

Feature Sample	1	2	3	4	5	6	.... m
1	x			x			
2		x	x		x		
3	x		x				
..							
n		x		x		x	

Notes

- The primary feature has an identification number which is smaller than all its secondary features.
- The presence of a feature in a sample is represented by an 'x' in the corresponding row and column of the matrix.

This matrix of samples and features can be summarized in 2 ways and put into a knowledge base of pattern classes and a knowledge base of hypothesis sets.

5.2 KNOWLEDGE BASE OF PATTERN CLASSES

A **pattern class** is defined as a class of all the numerals which have exactly the same set of features. The number of samples that belong to the same pattern class is called the **size** of the pattern class. The rows in the above result matrix can be grouped and represented as entries about pattern classes. Each entry for a pattern class contains:

- The pattern class **identification number,**
- The **feature vector**, implemented as an array of characters. A feature is represented by a column, this column is blank if the pattern class does not have this feature, or contains a character 'X' if it has the feature.
- The **label** of the pattern class, which is the identity of the sample (0..9). Because of the possible limitation of features, samples of different numerals may produce the same feature vector. The use of the label will create different entries of pattern classes for different numerals with the same feature vector.
- The **size** of this pattern class.

The above entries form the knowledge base of pattern classes.

5.3 KNOWLEDGE BASE OF HYPOTHESIS SETS

For a column representing a feature in the result matrix, we can count the number of occurrences of the 10 numerals having that feature in the training set and obtain a set of 10 values. As the basic assumption is that the training set closely represents all the input samples, this set of values also represents the weight or the probability of an unknown sample, having that feature, being one of the 10 numerals. The set S(F) of the frequencies of occurrences of 10 numerals having the feature F in the training set is called the **hypothesis set.**

$S(F) = (n_0 \text{ of } 0\text{'s}, n_1 \text{ of } 1\text{'s}, \ldots n_9 \text{ of } 9\text{'s})$

where $n_0 .. n_9$ are the frequencies of occurrences of the numerals 0..9 having feature

F in the training set.

The above information forms a knowledge base of hypothesis sets. Each feature (primary or secondary) has an entry in this knowledge base containing:

- Feature **identification**: the primary feature string and a number representing the secondary feature,
- A short **description** of the secondary feature. In case of primary feature, the description is an asterisk (*).
- The **hypothesis set** of the feature.

Examples of knowledge base entries are shown in Table 5.2.

The knowledge base of hypothesis sets can be derived from the knowledge base of pattern classes but the inverse is not true. Both of them are created from the result matrix obtained from training.

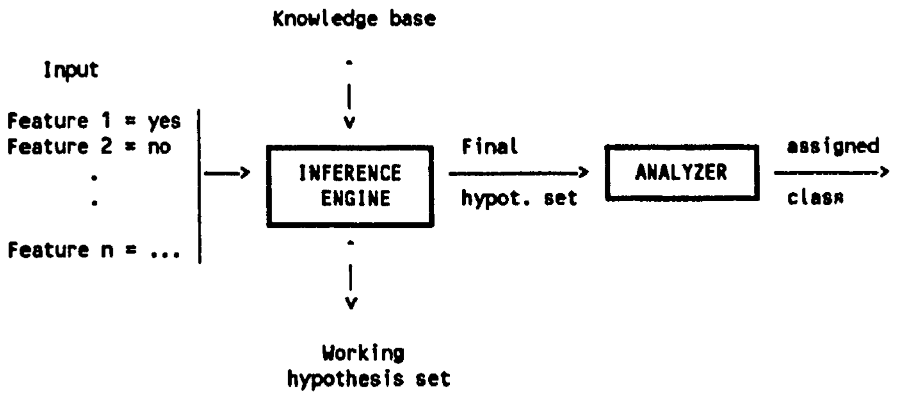
5.4 RECOGNITION BY INFERENCE

The knowledge base of hypothesis sets can be used by a inference engine to deduce the identity of an unknown sample. The inputs to the inference engine are the features of the unknown numeral. Its working variable is a hypothesis set, which lists the probability measurements of different hypotheses. During the inference process, each result of feature inspection is compared with the knowledge base, and according to the result of the comparison, the hypothesis set is modified. When all inputs have been considered, the analyzer examines the final hypothesis set and assigns a class to the unknown numeral.

Table 5.2 Contents of knowledge base of hypothesis sets

Primary Feature	Secondary Feat.Id.	Number of occurrences 0	1	2	3	4	5	6	7	8	9	Total	Description
EF	1	125	0	34	12	0	8	424	1	18	0	622	*
blank	2	847	0	0	0	0	0	0	0	0	0	847	*
FE	3	34	1	2	9	1	0	0	2	15	230	294	*
EE	4	210	1793	127	278	6	320	60	788	0	33	3615	*
EE	5	47	15	0	1	0	185	4	0	0	23	275	CURVE AT TOP LEFT
EE	6	0	1554	0	0	0	0	0	5	0	0	1559	SMALL MAX. WIDTH TO DISTINGUISH 1
EFEE	7	3	4	178	466	4	42	5	61	1	3	767	*
EFEF	8	0	0	33	32	0	6	1	1	3	0	76	*
EFFE	9	0	0	235	8	2	4	56	2	3	8	318	*
FFFE	10	1	0	11	0	0	0	0	0	1	3	16	*
EEX	11	1	0	1	0	0	0	1	0	11	0	14	*
EEFE	12	2	2	70	214	384	11	16	35	18	62	814	*
EE	13	0	0	3	220	0	20	0	3	0	1	247	PROFILE OF 3/8 AT RIGHT
EE	14	7	8	62	57	0	43	26	2	0	0	205	LARGE WIDTH AT BOTTOM
FE	15	1	0	0	7	0	0	0	0	3	3	14	PROFILE OF 3/8 AT RIGHT
EEFFEE	16	0	0	18	6	64	0	1	2	7	0	98	*
EEFE	17	0	0	0	0	247	0	1	0	1	39	288	PROFILE OF 4 ON LEFT
EEFE	18	2	2	6	139	133	8	2	34	17	22	365	2 END POINTS AT TOP
EEFEFE	19	0	0	2	4	133	0	0	0	0	0	139	*
EEFE	20	0	0	0	0	257	0	10	0	3	1	271	STRAIGHT STROKE ON A 4
FEFE	21	2	0	0	1	3	0	0	4	0	2	12	*
EE	22	2	2	0	3	1	307	0	9	0	26	350	INFLECTION POINT IN A 5
EEFFEEFE	23	0	0	0	2	3	0	0	0	0	0	5	*
EEFE	24	0	0	1	180	1	4	0	0	1	0	187	A LIMB OF 3 AFTER FORK
EF	25	91	0	0	0	0	0	1	0	0	0	92	FORK POINT CLOSE TO END POINT
EE	26	146	0	0	0	0	0	0	0	0	0	146	SMALL DISTANCE BETWEEN EE
EEEF	27	0	0	4	1	0	0	6	0	0	0	11	*
EE	28	0	0	0	0	0	0	0	0	0	0	0	NUMERAL LEFT OF 2 END POINTS
EEFF	29	3	0	28	16	0	0	4	0	186	0	237	*
EFFE	30	1	0	1	0	0	3	10	0	55	0	70	*
EE	31	30	409	7	55	4	7	0	743	0	6	1261	NUMERAL RIGHT OF 2 END POINTS
EFEE	32	0	0	0	0	0	0	0	0	0	0	0	LAST END POINT AT RIGHT
EEFE	33	1	1	0	30	4	6	0	32	0	15	89	FORK POINT NEAR TOP
EFEE	34	3	4	1	290	1	36	1	60	0	2	398	FORK POINT NEAR TOP
EEFF	35	0	0	17	1	0	0	4	0	1	0	23	LARGE DISTANCE BETWEEN 2 END POINTS (+2,+6)
EFEE	36	0	2	1	82	0	0	0	43	0	1	129	FORK ON RIGHT
FF	37	15	0	1	0	0	0	0	0	221	0	237	*
EE	38	62	14	105	0	0	0	39	5	0	0	225	CURVE AT BOTTOM LEFT
FEFF	39	0	0	0	0	0	0	0	0	11	0	11	*
EE	40	0	1597	0	0	0	0	0	0	0	1	1598	STRAIGHT STROKE BETWEEN EE (+1)
EFFFEE	41	0	0	1	0	0	0	1	0	4	0	6	*
....													

Fig. 5.3 : Recognition using the inference engine



All features used by the inference engine of this method are represented in the following form: feature identification and a Boolean variable (Yes/No) signifying whether the unknown numeral has the feature or not. With this simple form of representation, the inference is flexible and can handle all types of features used in syntactical recognition.

5.4.1 Inference operations

The inferencing process is a matching process: Given an unknown sample, find from the knowledge base the entire set of numerals in the training set that have all the features of this unknown sample.

When an unknown sample is found to have a primary feature F_0, then it must belong to set $S(F_0)$. This set is a universal set for feature F_0 because all samples in the training set having feature string F_0 would belong to this set.

The unknown sample is also checked against all the secondary features which have F_0 as the primary feature. For each secondary feature F_n, there are 2 possibilities, the sample has feature F_n, or it does not have it.

- If the unknown sample has feature F_n, then there is a hypothesis set $S_n = S(F_n)$ of known numerals in the training set having the same features as this unknown sample.
- If the unknown sample does not have feature F_n, the hypothesis set is the complement of set $S(F_n)$ with respect to the universal set $S(F_0)$, hence the hypothesis set will be $S_n = S(F_0) - S(F_n)$.

By checking each feature, the sets S_0, S_1 .. S_n will be computed. The final hypothesis set S_f of all known numerals in the training set having all the features of the unknown sample will be the intersection of the sets $S_0..S_n$:

$$S_f = S_0 \wedge S_1 \wedge S_2 \ \dots \wedge S_n$$

The final hypothesis set is given to the Analyzer to decide on the classification result. The final hypothesis set corresponds to all the features that have been examined, in other words it corresponds to the feature vector.

The inferencing process is illustrated in Appendix 2. In this illustration the processing is sequential. However an inference step can be done as soon as a result is returned from one feature extraction step, hence the inferencing process can be in a mixed sequential-parallel manner like feature extraction (as discussed in Section 4.7).

5.4.2 The Analyzer

The final hypothesis set S_f of an unknown sample is of the form:

$$S_f = (n_0 \text{ of 0's}, n_1 \text{ of 1's} ..) \qquad (5.1)$$

Percentages p_i $(i=0..9)$ of all hypotheses are calculated, where:

$$p_i = 100 * n_i / (n_0 + n_1 + .. + n_9) \qquad (5.2)$$

The hypothesis having the highest percentage is called the **dominant hypothesis** and is selected as the recognition result. If a sample has more than 1 hypothesis that have the same highest percentage, it will be rejected. A sample is also rejected if the final hypothesis set is a null set.

A minimum threshold of percentage value can be established to control the reliability of recognition. A sample is rejected if the highest percentage is lower than the threshold.

This method of decision is based on common sense reasoning. Of all the numerals in the training set which have all the features of the unknown sample, we would naturally feel that the one which occurs most frequently would be most likely to be the right answer. It can be proved that this decision follows Bayes' Rule:

Let - $p(i)$ be the a-priori probability of class i,

- $p(X|i)$ be the probability of observing a feature vector X in class i,
- $p(i|x)$ be the probability that a sample having feature vector X belongs to class i,

Then Bayes' Rule states:

$$p(i|x) = \frac{p(X|i)\, p(i)}{\sum_{k=0..9} p(X|k)\, p(k)} \qquad (5.3)$$

Since the denominator is the same for all i's, it can be omitted when different values of $p(i|x)$ are compared. Then:

$$p(i|x) = K\, p(X|i)\, p(i) \qquad (5.4)$$

where:

$$K = \frac{1}{\sum_{k=0..9} p(X|k)\, p(k)}$$

Let - N_i = number of numerals i in training set

- $N_t = N_0 + .. + N_9 = 8500$ = size of training set
- n_i = frequency of occurrences of numeral i in the final hypothesis set S_f (i.e. having feature X)

The probability of numeral i in the training set is:

$p(i) \quad = N_i/N_t$

If we assume that the a-priori distribution of numeral in the test set is the same as the training set, then the a-priori probability of numeral i in the test set is also:

$p(i) \quad = N_i/N_t$

The probability of a feature vector X being observed in numeral i from the training set is:

$$p(X|i) = n_i/N_i \qquad (5.5)$$

The probability that a sample having feature vector X being a numeral i, using equation (5.4) is:

$$p(i|x) = K\,(n_i/N_i)\,(N_i/N_t) = (K/N_t)\,n_i \quad (5.6)$$

Let $K' = K/N_t$ = a constant, then:

$$p(i|x) = K'\,n_i \qquad (5.7)$$

Hence if a-priori knowledge of probability of numeral i in the test set is assumed to be the same as in the training set, then the probability that a sample in test set with feature vector X being numeral i is proportional to n_i, the frequency of occurrence of numeral i with the same feature vector X found in the training set.

If the a-priori knowledge of probability of the 10 numeral classes in test set is not known, p(i) is assumed to be the same for all numeral i's, then:

$$p(i|x) = K\,(n_i/N_i) * \text{a constant} = K''\,(n_i/N_i) \qquad (5.8)$$

5.5 STRUCTURAL CLASSIFICATION

5.5.1 Feature matching

In order to use the knowledge base of pattern classes for recognition, each feature vector should correspond to one entry and one label. If there are more than one entries, the situation can be handled in 2 ways:

- A new and well-chosen feature can be added, which produces 2 new different feature vectors with unique labels from the previous feature vector. However, as discussed in Section 4.8.1, adding new features becomes more and more difficult when the number of features is already high. In the training stage, thc addition of features becomes ineffective when:

  + The added feature is trivial and only serves to distinguish 2 pattern classes of poorly-written numerals which occur rarely and cannot be considered representative.
  + The added feature is non-trivial, but there are not sufficient samples in the training set which have that feature, therefore the rule based on that feature is not effective and still produces a wrong hypothesis. One example is given in Fig. 7.14.

- A most dominant label is selected, and all the others are discarded. If there is no dominant label, all the entries having that feature vector are discarded. This is the same situation like when inference is used and the final hypothesis set contains more than one hypothesis. In fact we can verify that the weight n_i of a hypothesis in the hypothesis set and the size of the pattern class is the same quantity. Therefore, the logic of the Analyzer (Section 5.4.2) can be used to select the most dominant pattern class entry.

The knowledge base of pattern classes now contains entries which have only one unique label for each feature vector. An unknown sample can be recognized by comparing its feature vector against this knowledge base. Because the label is unique, there can be no more than one match. If there is a match, the label of the matched entry is the recognition result. If there is no match, the sample is not recognized.

5.5.2 Selecting a confidence level

Pattern classes have different sizes. The class that occurs more frequently will have a higher confidence level. By applying different thresholds in pattern class size and discarding classes having a size smaller than the threshold, the recognition rate decreases but the substitution rate and reliability of recognition increase. Thus the method can be tuned to achieve the desired reliability or desired recognition rate.

We should also consider the fact that each numeral has a different degree of shape complexity. For example 0, 1 and 7 are fairly simple in shape, while 2, 4, 8 may have more complicated shapes. The number of pattern classes tends to increase with the degree of complication in the shape of the numeral, and consequently, the average size of a pattern class will decrease. Therefore it is not beneficial to apply the same threshold of pattern class size for all numerals. One approximate value for a threshold is:

$T_s = S_{ave} * F_t$

where : T_s : Threshold value,

S_{ave} : Average size of a pattern class of the numeral in the training set,

F_t : A tuning factor varying from 0.0 to 2.0 or more.

It should be noted that a tuning factor of 0.0 means all generated pattern classes from the training set which are correctly recognized will be used. With a tuning factor of 1.0, if the distribution in size is normal, 50% of the samples are below the average and the rejection rate will be about 50%.

5.6 COMPARISON BETWEEN INFERENCE METHOD AND STRUCTURAL METHOD

The inference method and the structural method use the same features, they also use Bayes' Rule to select a recognition result. The difference is that the inference engine uses the relative ratio between different $p(i|x)$ values within a pattern class, while the structural method uses the absolute values of $p(i|x)$ which is the probability of the pattern class in the total training set.

The inference method relies on relative ratios between hypotheses. It can make an educated guess even when an unknown sample has a feature vector that does not match any entry in the knowledge base of pattern classes. Therefore the maximum possible recognition rate achieved by the inference method is always higher than the structural method.

On the other hand, when a character is recognized by the structural method, there must be a number of characters that have exactly the same set of features in the knowledge base, and this number must be higher than a certain threshold, hence the structural method gives a higher reliability rate.

The 2 methods complement each other and together they cover the whole range of recognition rate and reliability rate. Depending on the requirement of the user, the system can be tuned to achieve the desired performance.

The inference method is the original method that was used at the beginning of the research. The flexible and simple structure of its knowledge base facilitates experimentation in feature selection and extraction. When a feature is modified or added, its effect on recognition result can be seen directly in the program output (as in Appendix 2), and the proper hierarchy of features, such as feature's preconditions, can be built in a heuristic manner. On the other hand, the effect of a modified knowledge base of pattern classes on the recognition result can be seen globally only.

The structural method appears to be a strong candidate after enough experimental results have been collected. It achieves nearly the same recognition rate as the inference method but using less computing power because it does not perform the set operations (see Table 6.8 for computing costs). Also, if a high degree of reliability is desired, the number of required pattern classes in the knowledge base is smaller (see Table 9.9, Section 9.3.1), the number of features to be extracted will be smaller and less computing cost is needed for feature extraction. The pattern classes can also be analyzed, grouped and organized into a decision tree, feature matching can be changed from sequential search to binary search, resulting in further decrease in the computing cost.

6. EXPERIMENTAL RESULTS - OVERVIEW

The data used in this work are a set of handwritten ZIP codes collected by U.S Postal Services from mail envelopes written by the general public. The number of writers is unknown, but we can assume that from the size of the samples (16,985), there would be approximately 3,397 writers, because each US postal code has 5 digits. The data were digitized, binarized and segmented before being processed by this method.

Of the total 16,985 samples, 8500 were used in training, the remaining 8485 for testing. The numerals are not evenly distributed, the frequencies of occurrence of numerals are as shown:

Table 6.1: Distribution of numerals in the data base

NUMERAL	FREQUENCY OF OCCURRENCE	USED IN TRAINING	USED IN TESTING
0	2500	1250	1250
1	3594	1800	1794
2	1673	800	873
3	2174	1100	1074
4	1439	700	739
5	787	400	387
6	1241	600	641
7	1720	900	820
8	1161	600	561
9	696	350	346
Total	16985	8500	8485

Training was done alternatively with recognition using the inference method. From recognition results, the errors were analyzed and features were modified, added or deleted from the knowledge base. In the beginning, a training set of 2000 and a test set of 2000 samples were used. Features were added to the knowledge base until the recognition rate on the training set could not be improved much further. A larger training set of 8500 and test set of 8485 samples were used and the knowledge base was refined with additional features until recognition rate of the training set reached a stabilized level. A full discussion of how recognition results change as the size of the training set and the test set increase can be found in Chapter 8.

The features obtained from a training set of 8500 samples are:

Table 6.2 : Features from the training set

Primary	87
Secondary	330
TOTAL	407

The above 407 features divide the training set into 1195 pattern classes:

Table 6.3 : Breakdown of pattern classes

	NO. OF CLASSES	NO. OF SAMPLES	AVE. SIZE OF CLASS
Recognition	947	8210	8.67
Rejection	168	201	1.20
Substitution	80	89	1.11
TOTAL	1195	8500	7.11

The distribution of pattern classes having correct recognition is:

Table 6.4 : Distribution of correctly recognized pattern classes

NUMERAL	NO. OF CLASSES	NO. OF SAMPLES	AVE. SIZE OF CLASS
0	81	1220	15.06
1	75	1781	23.75
2	140	765	5.46
3	163	1059	6.50
4	98	683	6.97
5	96	371	3.86
6	69	569	8.25
7	87	869	9.99
8	72	561	7.79
9	66	332	5.03
TOTAL	947	8210	8.67

4 percentage rates are used to measure the performance of a method:

- Recognition rate (or success rate) is the percentage correctly recognized in all the input samples.
- Rejection rate is the percentage rejected as being not recognizable in all the input samples.
- Substitution error rate (or misrecognition rate) is the percentage recognized incorrectly in all the input samples.
- Reliability rate is the percentage that is correctly recognized in all the input samples after excluding the rejected samples.

The recognition results are summarized in Table 6.5, the confusion matrix and recognition results for each numeral are given in Tables 6.6 and 6.7.

Table 6.5 : Summary of recognition results

	Training	Testing
Set size	8500	8485
Recognition	96.59%	94.11%
Substitution	1.05%	2.53%
Rejection	2.36%	3.36%
Reliability	98.93%	97.10%

From this basis, different threshold parameters are used to get different recognition, substitution and reliability rates. The results are discussed in Chapter 9.

COMPUTING COST

The computer used for this work is a CDC Cyber Model 170/835, its processing power is rated at 3 MIPS (million instructions per second). The computing costs for different operations in preprocessing and recognition are given:

Table 6.8 : Computing costs

OPERATION	Ave. CPU sec./sample	
Smoothing	0.290	
Thinning	0.248	
Skeleton simplification	0.123	
* TOTAL PREPROCESSING		0.661
Feature extraction	0.126	
Inferencing	0.173	
Analyzing	0.004	
Structural classification	0.012	
* TOTAL RECOGNITION		
Inference method		0.303
Structural method		0.138
* GRAND TOTAL		
Inference method		0.964
Structural method		0.799

Table 6.6: Recognition results of the training set

Sample size = 8500

OUT -> IN	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	RECOG.	TOTAL
0						1	4		6		19	11	1220	1250
1				1			2	5			11	8	1781	1800
2	1			4	2			1	1	1	25	10	765	800
3	1	1	2			3	1	4			29	12	1059	1100
4			1					2		1	13	4	683	700
5				1							28	1	371	400
6	3	4			3	1					20	11	569	600
7		7	2	6	1					1	14	17	869	900
8	1		4		3					2	29	10	561	600
9					4			1			13	5	332	350
TOTAL	6	12	9	12	13	5	7	13	7	5	201	89	8210	8500

NUMERAL	TOTAL	REJECT	%	SUBST.	%	RECOG.	%	RELIABILITY
0	1250	19	1.52	11	0.88	1220	97.60	99.11
1	1800	11	0.61	8	0.44	1781	98.94	99.55
2	800	25	3.12	10	1.25	765	95.62	98.71
3	1100	29	2.64	12	1.09	1059	96.27	98.88
4	700	13	1.86	4	0.57	683	97.57	99.42
5	400	28	7.00	1	0.25	371	92.75	99.73
6	600	20	3.33	11	1.83	569	94.83	98.10
7	900	14	1.56	17	1.89	869	96.56	98.08
8	600	29	4.83	10	1.67	561	93.50	98.25
9	350	13	3.71	5	1.43	332	94.86	98.52
TOTAL	8500	201	2.36	89	1.05	8210	96.59	98.93
NORMALIZED %			3.02		1.13		95.85	98.83

Table 6.7 : Recognition results of the test set

Sample size = 8485

OUT -> IN	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	RECOG.	TOTAL
0			4	5	1		5	1	11		39	27	1184	1250
1	1		1		2	4		4		1	30	13	1751	1794
2				18	1	2			3	4	33	28	812	873
3			13		1	9	2	3	7		37	35	1002	1074
4	2		5	2			6	3	3	1	23	22	694	739
5	1		2	5			5	1	2		33	16	338	387
6	3	1	4	1	1	1			5		30	16	595	641
7		3	5	7	1				2	2	12	20	788	820
8	7		2	1	1		4	1		2	23	18	520	561
9	1		3	1	7	2		4	2		25	20	301	346
TOTAL	15	4	39	40	15	18	22	17	35	10	285	215	7985	8485

NUMERAL	TOTAL	REJECT	%	SUBST.	%	RECOG.	%	RELIABILITY
0	1250	39	3.12	27	2.16	1184	94.72	97.77
1	1794	30	1.67	13	0.72	1751	97.60	99.26
2	873	33	3.78	28	3.21	812	93.01	96.67
3	1074	37	3.45	35	3.26	1002	93.30	96.62
4	739	23	3.11	22	2.98	694	93.91	96.93
5	387	33	8.53	16	4.13	338	87.34	95.48
6	641	30	4.68	16	2.50	595	92.82	97.38
7	820	12	1.46	20	2.44	788	96.10	97.52
8	561	23	4.10	18	3.21	520	92.69	96.65
9	346	25	7.23	20	5.78	301	86.99	93.77
TOTAL	8485	285	3.36	215	2.53	7985	94.11	97.38
NORMALIZED %			4.11		3.04		92.85	96.81

ANALYSES OF EXPERIMENTAL RESULTS

The following 5 separate chapters analyze the experimental results in detail:

Chapter 7:	Analysis of recognition errors.
Chapter 8:	Dependence on the size of the training set.
Chapter 9:	Performance curve and fine tuning.
Chapter 10:	Additional experiments on data used by Concordia OCR project team.
Chapter 11:	Comparison with other studies.

The analyses and discussions in these chapters have the following purposes:

- To gain some understanding about the data and the characteristics of the problems related to the recognition of totally unconstrained handwritten characters,
- To evaluate the effectiveness of the knowledge-based approach and to add fine-tuning to the method in order to improve the recognition performance,
- To estimate the limitation of this method and other methods in general, and identify a realistic goal on the recognition of totally unconstrained handwritten numerals.

7. ANALYSES OF RECOGNITION ERRORS

Using the inference method, there were 1.05% substitution errors and 2.36% rejects in the training set. Most poorly written samples (illustrated in Section 2.1) were rejected because the hypothesis set for each sample did not have a dominant hypothesis satisfying the threshold condition (see Section 5.4.2).

The major confusions observed in the training set and the test set were:

Table 7.1 : Major confusions

Notes: In this table, the expression 1-7:12 means that there are 12 misrecognition between numerals 1 and 7, where numeral 1 is misrecognized as 7 or vice versa.

Training Set	Test set
1-7 : 12	1-7 : 5
3-7 : 10	3-7 : 13
0-8 : 7	0-8 : 18*
0-6 : 7	0-6 : 8
2-3 : 6	2-3 : 31*
1-6 : 6	1-6 : 1
2-8 : 5	2-8 : 4
4-9 : 5	4-9 : 5
3-5 : 4	3-5 : 14*
4-6 : 3	4-6 : 7
5-6 : 1	5-6 : 7
6-8 : 0	6-8 : 8*

Apart from the problems caused by unconstrained writing as discussed in Section 2.1, and preprocessing limitations (segmentation, binarization, thinning, hole filling, smoothing, connection of broken strokes, tail removal) already discussed in Chapter 3, the substitution errors in the training set were caused by:

- Poor handwriting that can confuse also human readers: confusion between 1 and 7 (Fig. 7.2), 0 and 8 (Fig. 7.3), 0 and 6 (Fig. 7.4), 9 and 4 (Fig. 7.5), 7 and 4 (Fig. 7.6).
- Limitation in the feature extraction:
  - + the profile feature representation does not truly represent the profile: numeral 7 has a right profile of numeral 3 (Fig. 7.7).
  - + an artificially established threshold cannot make a clear distinction between one numeral and another: 1 and 6 (Fig. 7.8), 6 and 4 (Fig. 7.9), 2 and 8 (Fig. 7.10).
  - + The distinguishing features are too subtle to be detected by the computer: 2 and 3 (Fig. 7.11), 3 and 5 (Fig. 7.12).
  - + Side effect of feature: one feature aims at recognizing one numeral, yet it can cause a rejection of another numeral (Fig. 7.14).
- Limitation of training samples: Insufficient representation of a numeral pattern in the training set causes a valid feature to have a low weight in the hypothesis set (Fig. 7.14).
- Segmentation error: 2 touching numerals given to the recognition stage as a single character (Fig. 7.13).

It can be noted that the confusions found in the training set and the test set are consistent, indicating an acceptable degree of accuracy in the method. There are a few exceptions which are marked with an asterisk (0 and 8, 2 and 3, 3 and 5, 6 and 8) where the number of confusions in testing is much higher than training. These exceptions may originate from the fact that there are not enough representative samples in the training set to capture fully all the possible confusions between the

Fig.7.2 : Confusion between 1 and 7

NUMERAL NO.2487 : 1

```
             --
             -E-
              -*-
               -*
               -*-
                -*
                -*
               -*
              -*-
              -*
             -*-
            -*-
          --*-
         --*-
        --*-
       -**-
      -*--
    --*-
   -**-
  -*--
 -*-
-E-
--
```

NUMERAL NO.6727 : 7

```
        -----
         E**--
           -*--
            -*-
            -*-
           -*-
           -*-
           -*-
           -*-
           -*-
           -*-
           -*-
           -*-
          --*-
          --*-
          --*-
         --*-
          -*-
         --*-
         -*-
        --*-
        --*-
       --*-
       --*-
       --*-
      --*--
      --*-
      --*-
      -*-
     -*-
     -*-
     -*
    --*-
    --*-
   --*-
   -*-
  --*-
  -*-
 --*-
 --*-
 --*-
 -*-
--E-
----
 --
```

Fig.7.3 : Confusion between O and 8

NUMERAL NO. 261 : O

```
        --
       -**--
       *  *-
       *  -*-
        *-*F-
        -F--*
       -*-  -*
      -*-   -*
     -*-    -*
    -*-     -*
   -*-      -*
  --*-      *
 --*-      -*
 -*-       -*
-*-        *
-*-       -*
-*-   ----*
--*---****
---***---
  ----
```

101

Fig.7.4 : Confusion between 0 and 6

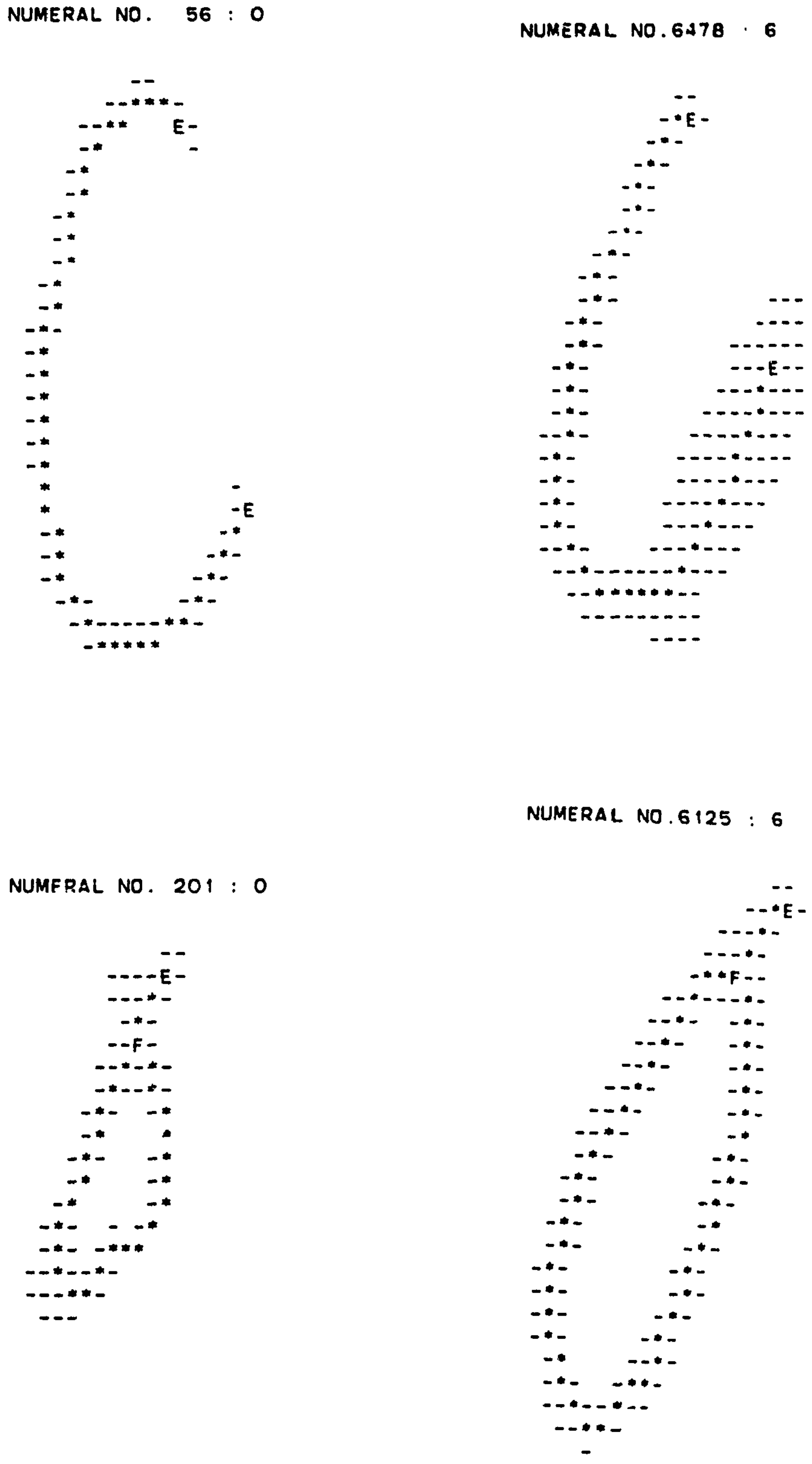


Fig.7.5 : Confusion between 9 and 4

NUMERAL NO.8233 : 9

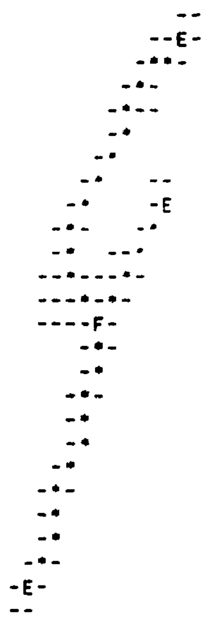
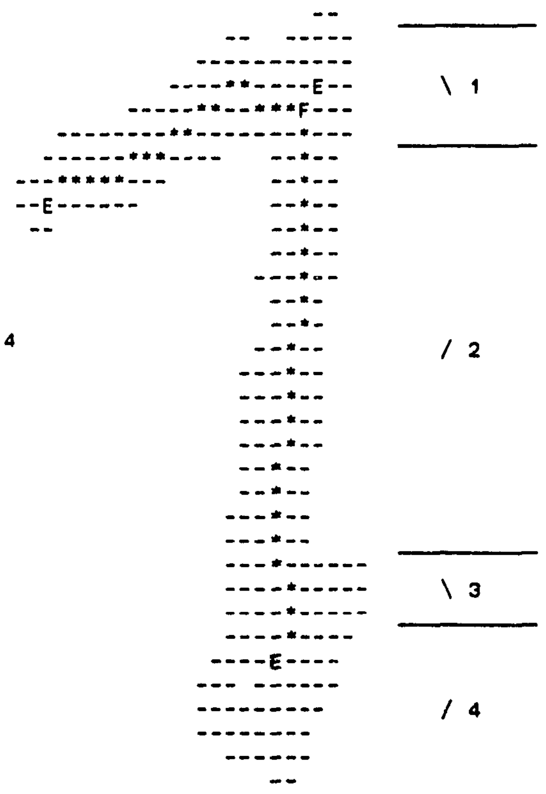


Fig.7.7 : 7 misrecognized as 3

NUMERAL NO.7152 : 7



The right profile is identified by the computer as '1234' which is the profile of numeral 3.

Fig.7.6 : Confusion between 7 and 4

NUMERAL NO.7523 : 7

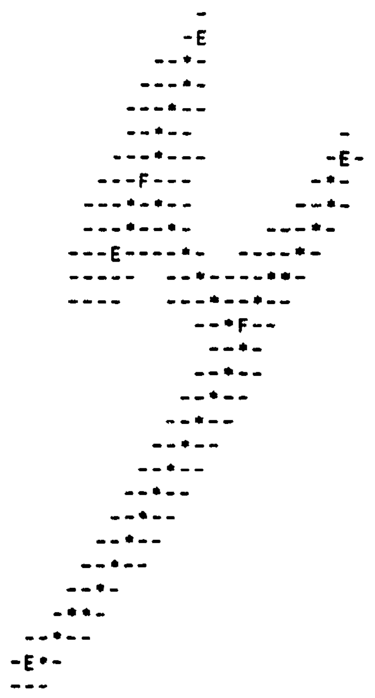


Fig.7.8 : 6 misrecognized as 1

```
NUMERAL NO.6141 : 6

                    -
                  --E-
                 --*--
                --*-
              ---*-
             --**-
            --*--
           --*-
         ---*-
        ---*-
       --**-
      --*--
     ---*----
    ----*------
   -----*------
  ------*------
  -----*-------
 -----*-------
 ----*-------
----E-------
-----------
 -------
  ----
```

Feature used is the thickness of the bottom half, but some samples of numeral 1 also have a thick bottom half

Fig.7.9 : 6 misrecognized as 4

```
NUMERAL NO.6247

        --
       ----
      --E--
      --*--
     ---*--
     --*--
    ---*--
    --*--
   ---*--
   --*--
  --*--
  --*--
 --*--
 --*--       ----
 --*--     -------
--*--    ----------
--*--   -----------
---*---------------
---*-------**E-----
----*---***-------
 ----F**---------
  ---*----------
   --*--------
  --*--
  --*--
  --*-
  --*--
  --E--
   ----
    --
```

Feature used is the thickness of the right limb, but some samples of numeral 4 also have a thick right limb

Fig.7.10 : 2 confused with 8

NUMERAL NO.3198 · 2

```
           --------
          -E********--
                   **-
                    -*-
                     -*
                      -*
                      -*
                      -*           --
                      -*-         -*E
                      -*-      --**
                      --F----**
                     --*-****
                    --*---
                 ----*--
               --****F--
             --**     --*-
            -**       -*-
          -**         -*-
         -*           -*
        -*           -*-
       -*            -*-
      -*            -*-
     -*            --*-
    -*             -*-
   -*             -*-
  -*             -*-
  -*            --*-
 -*           ---*-
 -*          ---*-
 -*        ---**-
-*        --**--
-*-  ---**--
--*--***--
---**---
```

This numeral 2 has a high fork point and a big bottom loop. It can be confused with numeral 8

Fig.7.11 : 2 misrecognized as 3

NUMERAL NO.3441 : 2

```
                --
          --------
         -E*******--
         ---    -*-
                 -*-
                  -*
                  -*
                  -*
                 -*-
       -----------F-
        E********-*-E-
               -F-
              -*
             -*
           --*-
-        --**
-E-------***
 -******
```

Human reader may detect a broken 2, but computer decides it is numeral 3

Fig.7.12 : 3 misrecognized as 5

NUMERAL NO.4292 : 3

```
             --
              E    bend upwards
             -*    to left
             -*
            -*
           -*
           -*
          -*
         -*
        -*
       -*-
       --**----
           ***--
             -*-
             -*
           --*
          -**
         -*-
      --**
    --**
 ---**
-E**
```

A small bend on top can be detected
by human reader, but not by computer

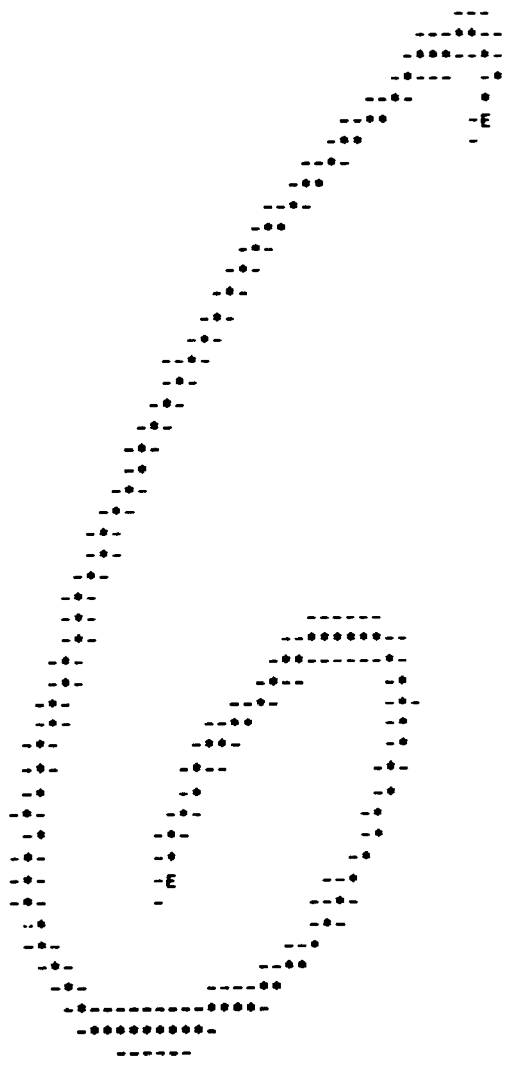
Fig.7.13 : Segmentation error

NUMERAL NO.6113 : 6

```
        --
       ----
       --E-                         ---
       --*-   -----                ----
       --*------------              -E--
       ---F-********---             --*-
      ---*-*--------*--             -*-
     ---*---      --*-             --*-
    ---*---       --*-            --*--
   --**--        --*--            --*-
 ---*---       ---*--            --*--
---*---     ------*--           --*--
--E--      ----*F--             --*--
----       --E*--*--           --*--
            -------*--        --*--
                --*--         --*--
                 --*--       --*--
                  --*--     --*--
  -                --*-    --*--
 -E                --*-    --*-       ------
 -*                 -*-   --*--     ---****--
-*-                --*-- --*--     ---*-----*--
-*-                ---F---*--     --**-    -*-
 -*-             ----*-*--*--     -*--     -*-
 --*-           ---**---*F--     -*-       -*-
  --*--     -----**---  --*-    --*-      --*-
   --**----------*----  --*-    -*-    - --*--
    ---**********---   --*--    --*-   ---*--
         ---------     --*--    --*-----**--
                        --*-------F----*---
                        ---*----**-****---
                          --****--------
                           -----------
```

Fig.7.14 : Limitation of training samples and
side effects of features

NUMERAL NO.6372 : 6



LIMITATION OF TRAINING SAMPLES

Hypothesis set for feature string EE:

```
    0 :  210        5 :  320
    1 : 1793        6 :   60
    2 :  127        7 :  788
    3 :  278        9 :   33
    4 :    6
```

Feature 5 : curve at top left

```
    0 :   47        5 :  185
    1 :   15        6 :    4
    3 :    1        9 :   23
```

Feature 48: first end point is not near top border

```
    0 :   13        5 :    3
    1 :    1        6 :    1
    2 :    8        7 :  256
    3 :   31        9 :   17
```

In the training set, there were only 4 samples of numeral 6 which had a curve at top left, and only 1 numeral 6 which had a first end point not in the top border. Because of features 5 and 48, the final hypothesis set had only 1 occurrence of 6. Using other features, the derived final hypothesis set had no dominant hypothesis, and the sample was rejected. If there were more samples of 6 with feature 5 and 48 in the training set, the sample would be correctly recognized as numeral 6.

SIDE EFFECT OF FEATURES

In the above example, the purpose of feature 5 is to detect a top left curve of numeral 5 or 9. The purpose of feature 48 is to reject numerals that are not 1 (most samples of numeral 1 have the first end point near the top border). The rejection of 6 from the result set of feature 48 is an unwanted side effect.

2 numerals. This explanation is in agreement with our observation on the limitation of training samples noted above (Fig. 7.14).

An analysis of substitution error rates of the training set and the test set shows that more errors occur in the test set for the following cases:

- numerals that have more complicated stroke structures than others (2,3,4),
- numerals that have less samples in the training set than others (5,9),

A comparison of substitution error rates in training and testing is shown in Table 7.15.

Table 7.15 : Comparison of substitution error rates

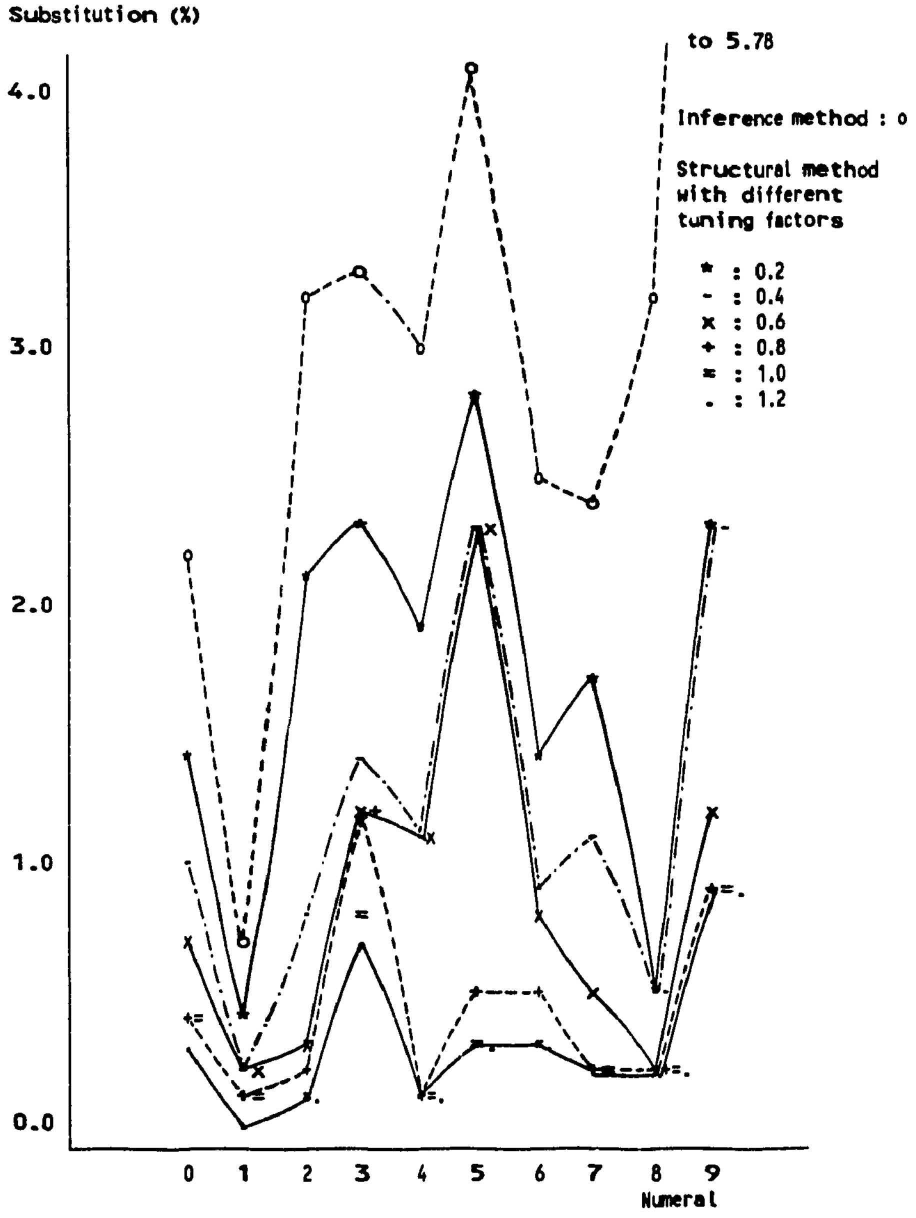
Numeral	% Error Rate Training	% Error Rate Testing	% Increase
0	0.88%	2.16%	245%
1	0.44%	0.72%	164%
2	1.25%	3.21%	257%
3	1.09%	3.26%	299%
4	0.57%	2.98%	523%
5	0.25%	4.13%	1652%
6	1.83%	2.50%	137%
7	1.88%	2.44%	130%
8	1.67%	3.21%	192%
9	1.43%	5.78%	404%

A substitution error rate in testing which is substantially higher than training indicates that training for that numeral has not covered a good proportion of possible shapes of the numeral. This suggests that improvement can be made with a larger training set which, because of its size, contains more features and is more representative of the total population. In this study, we have imposed a constraint that the training set should be at most approximately equal in size to the test set, in order to avoid the situation that the small size of the test set may cause a falsely high recognition rate. This constraint is reasonable because in practice, the recognition method should work on a very large volume of test data, many times greater than the training set. Therefore, the largest size of the training set that was used is 8500, being approximately 1/2 of the total number of samples.

Substitution error rates of different numerals at different reliability thresholds are also given (Fig. 7.16). The tuning factors used in the graph are explained in Section 5.5.2. Although the substitution error rate decreases as higher tuning factor is used, this is not realistic and beneficial, as discussed in Section 9.3.1.

Numeral 9 is the one that has the highest substitution error rate, and it also happens to be the one that has the smallest number of representation in the data base (see Table 6.1)

Fig.7.16 : Error rates of different numerals at different reliability thresholds



8. DEPENDENCE ON THE SIZE OF TRAINING SET

"A character recognition system can only be adequately judged when it has been used in practice for millions of numerals" .. "It is of utmost importance to know exactly what test material was used" .. "We have found by personal experience how easy it is to deceive oneself and others in this kind of experiments".([BEUN73], p.98).

The above excerpts from Beun's study, a classic paper on character recognition are important notes of caution to this study. One question which influences the design methodology and the experimentation in the study is: "Is the data base representative enough so that the method developed using this data base will give acceptable reliability and what kind of accuracy we can expect in practice". This question is partly answered by the recognition rates achieved with the test set. Further confirmation can be obtained by starting training and testing with small training and test sets, and gradually increasing the size of the training and test sets. The variation in recognition rates should give an idea of the reliability of the recognition results.

Another reason to find the dependence of the recognition results on the size of the training set is to identify the factors that may help to improve the recognition rate.

8.1 DEPENDENCE OF NUMBER OF FEATURES ON THE SIZE OF TRAINING SET

When the knowledge base was built, each feature was added only when the human expert judged that the additional feature added more discriminating power to the knowledge base. It is natural to expect that as the training set increased in size, more features must be added to maintain a high recognition rate, and when the training set is small, there may be some features that are useful but not yet encountered.

407 features were finally established with a training set of 8500 samples. Various runs with different smaller sizes of training set were performed to see how many of these features were present in smaller training sets. It is found that the curve with the number of observed features plotted against the size of the training set increases sharply at smaller sizes, but the rate of increase levels off at larger sizes (Table 8.1, Fig. 8.2).

8.2 ANALYSES ON THE NUMBER OF PATTERN CLASSES

When grouping the data in the training set and the test set to pattern classes, it is possible to achieve the following:

- To see the number of pattern classes in the data base and to have some ideas on whether the obtained pattern classes cover the majority of all possible common pattern classes.
- To estimate the proportion of samples belonging to common pattern classes in this data base that may be encountered also in an unknown data base.

Various runs with different sizes of training set and test set were performed, and the following data were collected:

- Number of pattern classes present in the training set.
- Number of pattern classes present in the test set.
- Number of pattern classes present in both the training and the test sets (common pattern classes), and the percentage of samples having common pattern classes in the total data base.

The relationship between the number of pattern classes and the size of the training set is tabulated in Table 8.1 and plotted in Fig. 8.3.

Analysis of the number of samples in common pattern classes and its percentage in the total data base are presented in Table 8.4 and Fig. 8.5.

It is also noted that as the size of the training set increases, the average size of a pattern class that is not a common class (for convenience, we call this **uncommon pattern class**) is relatively constant at about 1.5 samples / pattern class, meaning that the uncommon pattern class is more or less one of a kind. Meanwhile the average size of a common pattern class increases as the size of training set increases, indicating a consolidation of common pattern classes when a larger training set is used. This relationship is tabulated in Table 8.6 and plotted in Fig. 8.7.

The different measurements plotted against the size of training set in Figs 8.3, 8.5, 8.7 also indicate a stabilization of recognition performance when a large size of training set is used.

Table 8.1 : Effect of the size of training set on the number of observed features and the number of pattern classes.

Size of training set (1)	No. of observed features (2)	No. of pattern classes				
		Train. (3)	Test. (4)	Total (5)	Combined (6)	Common (7)
265	198	87	95	182	133	49
530	251	159	152	311	215	96
848	274	223	221	444	309	135
1060	290	270	258	528	364	164
1697	317	351	358	709	482	227
2121	327	427	448	875	599	277
4242	374	720	721	1441	993	448
8500	407	1195	1163	2358	1640	718

Notes:

- Column (2) = number of features of the total 407 that are found in the smaller training set.
- Column (3) = number of pattern classes found in the training set.
- Column (4) = number of pattern classes found in the test set.
- Column (5) = Column (3) + Column (4)
- Column (6) = number of pattern classes that appear in the training set OR the test set.
- Column (7) = number of pattern classes that appear both in the training set AND the test set, (7) = (5) - (6).

Fig. 8.2 : Effect of the size of training set on the number of observed features

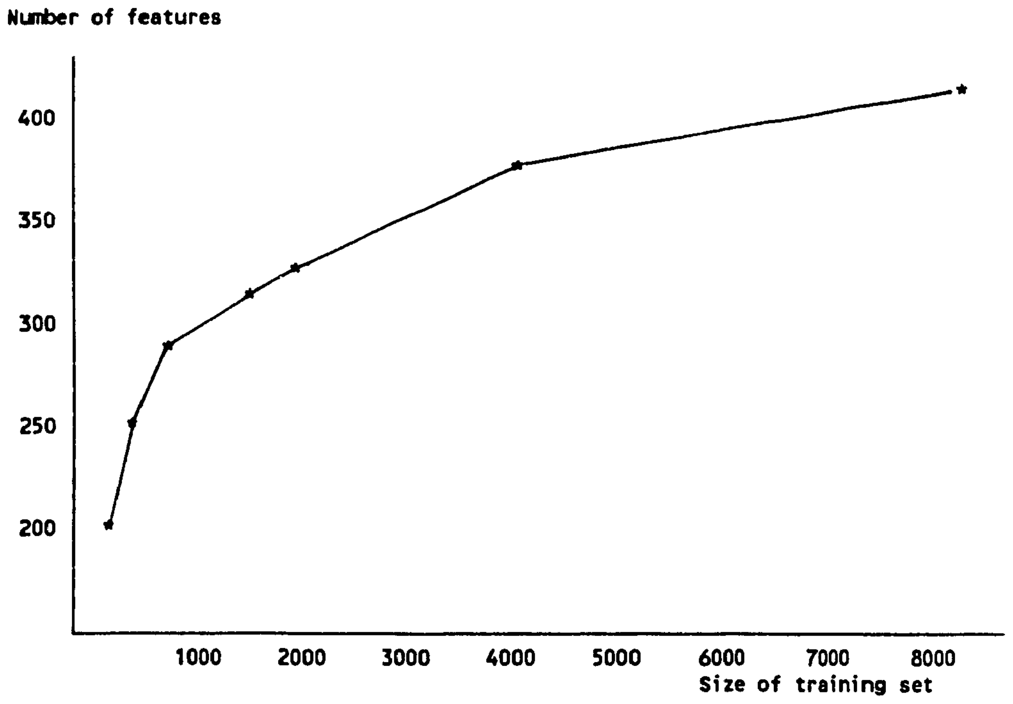


Fig. 8.3 : Effect of the size of training set on the number of pattern classes.

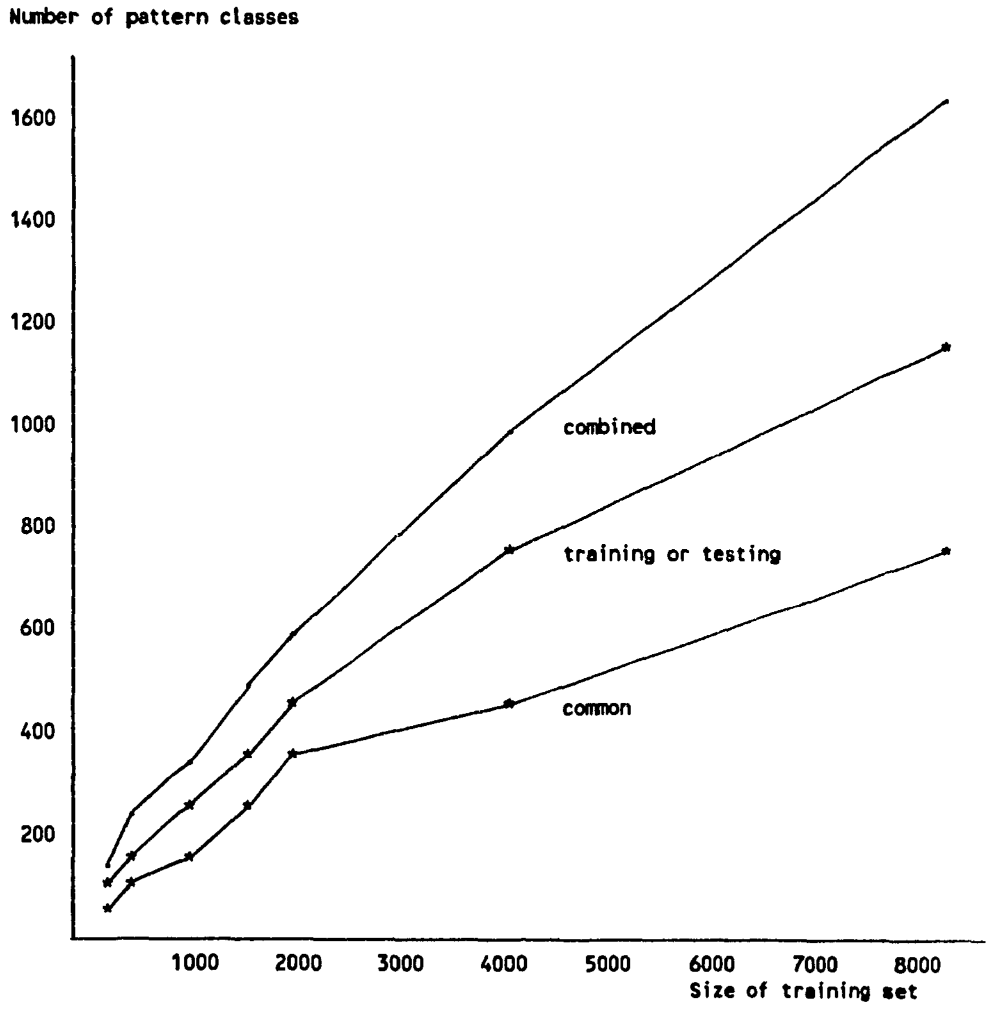


Table 8.4 : Statistics on pattern classes

Size of			No. of patt.class.		No. of samp. having common pattern class.			% in common classes
Train. set (1)	Test set (2)	Total (3)	Common (4)	Uncommon (5)	Train. (6)	Test (7)	Total (8)	(9)
265	265	530	49	84	213	187	400	75.47
530	530	1060	96	119	445	441	886	83.58
848	848	1696	135	174	712	707	1419	83.67
1060	1060	2120	164	200	913	908	1821	85.90
1697	1697	3394	227	255	1519	1503	3022	89.04
2121	2121	4242	276	323	1932	1886	3818	90.00
4242	4242	8484	448	645	3842	3851	7693	90.68
8500	8485	16985	718	922	7818	7902	15720	92.55

Notes :

- Table 8.4 is related to table 8.1. Column (4) is column (7) in Table 8.1, Column (5) is derived from column (6) - Column (7) of Table 8.1.
- Column (9) = Percentage of samples in common pattern classes
  = Column (8) * 100 % / Column (3)

Fig. 8.5 : Percentage of samples in common pattern classes

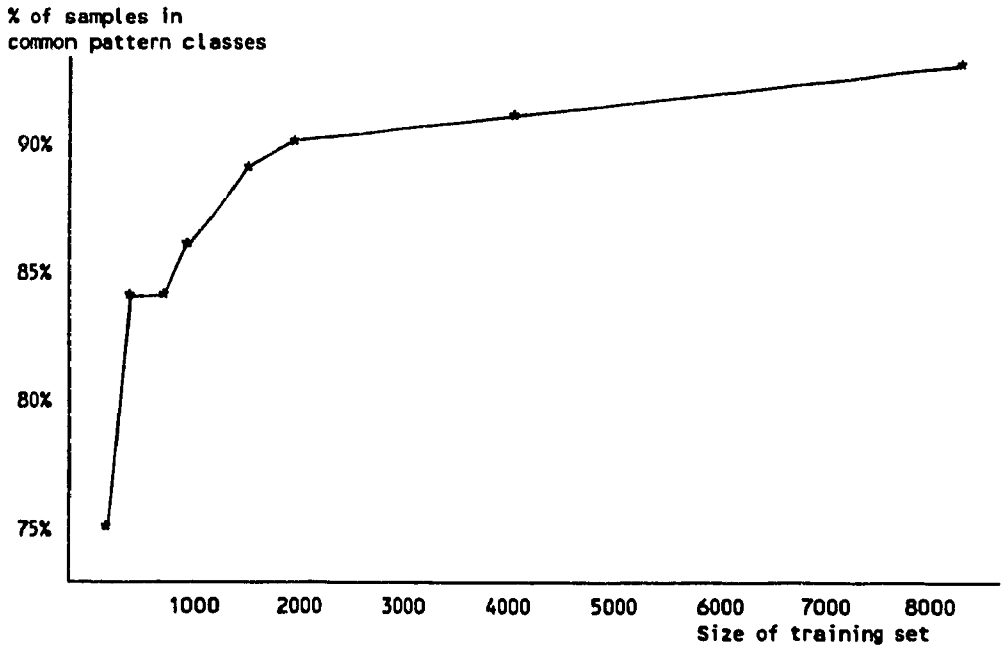
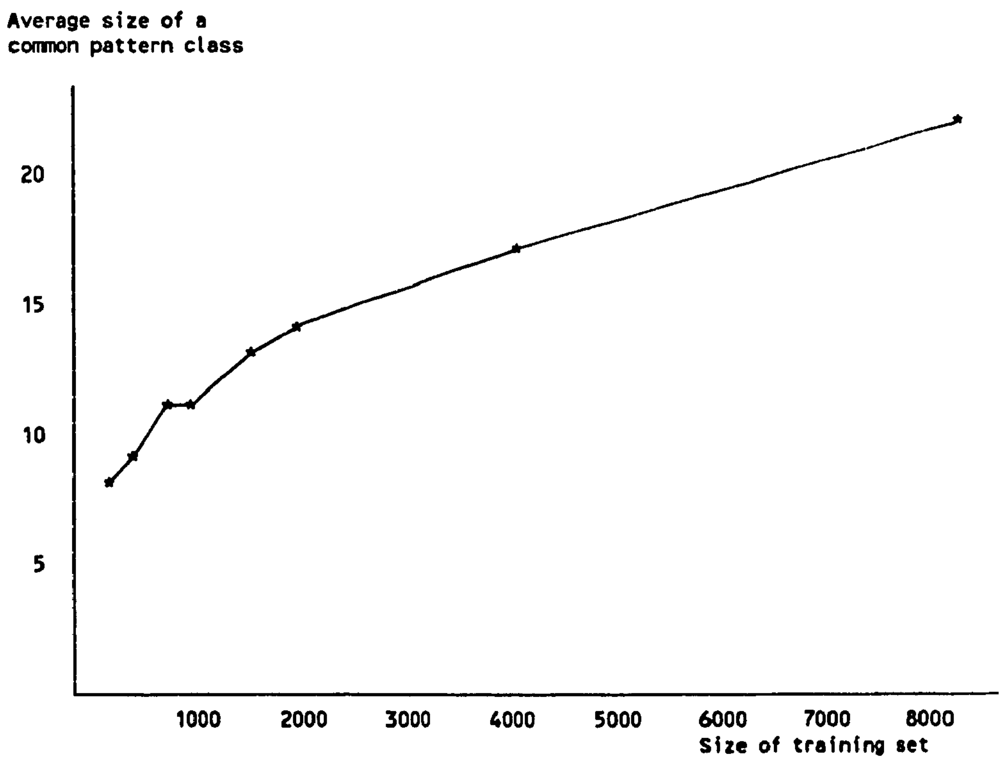


Table 8.6 : Average size of a pattern class

Size of	Common pattern classes		Uncommon pattern classes	
Train.set (1)	Number (2)	Average size (3)	Number (4)	Average size (5)
265	49	8.16	84	1.55
530	96	9.23	119	1.46
848	135	10.51	174	1.59
1060	164	11.10	200	1.49
1697	227	13.31	255	1.49
2121	276	13.83	323	1.31
4242	448	17.17	545	1.45
8500	718	21.89	922	1.37

Notes : Column (2) = Column (4) in Table 8.4
Column (3) = Column (8) in Table 8.4 / Column (4) in Table 8.4
Column (4) = Column (5) in Table 8.4
Column (5) = (3) - (8) then divided by (5) in Table 8.4

Fig. 8.7 : Average size of a common pattern class



8.3 ANALYSES OF RECOGNITION RATES

The recognition rates are influenced by the quality and quantity of features, and the number of generated pattern classes. Therefore the recognition rates should be related to the size of the training set in a very similar way like the number of features and the number of pattern classes.

From Table 8.8 and Fig. 8.9, it can be seen that as the size of the training set increases, the recognition rates for the test set increases and tends to stabilize. The difference between the recognition rate of the training set and the test set decreases. There is also a correlation between the recognition rate of the test set and the proportion of samples having common pattern classes in both the training set and the test set (Fig. 8.10).

Table 8.8 : Effect of the size of training set on the recognition rates

Size of train. set	No. of feat.	Training				Testing			
		Recog.	Sub.	Rej.	Reli.	Recog.	Sub.	Rej.	Reli.
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
265	198	99.62	0.00	0.38	100.00	72.08	2.26	25.66	96.95
530	251	99.62	0.00	0.38	100.00	83.77	2.08	14.15	97.58
848	274	98.23	0.12	1.65	99.88	83.96	2.24	13.80	97.40
1060	290	97.64	0.28	2.08	99.71	86.23	2.74	11.04	96.92
1697	317	97.94	0.23	1.83	99.86	89.69	2.47	7.84	97.32
2121	327	97.36	0.42	2.22	99.57	89.77	2.83	7.40	96.95
4242	374	97.34	0.75	1.91	99.23	92.41	2.45	5.14	97.42
8500	407	96.59	1.05	2.36	98.93	94.11	2.53	3.36	97.38

Notes:

- Columns (3) and (7) : percentage of correct recognitions.
- Columns (4) and (8) : percentage of substitution errors.
- Columns (5) and (9) : percentage of rejection.
- Columns (6) and (10) : reliability, being the percentage of correct recognitions over the number of samples that are not rejected.

Fig. 8.9 : Effect of the size of training set on the recognition rates

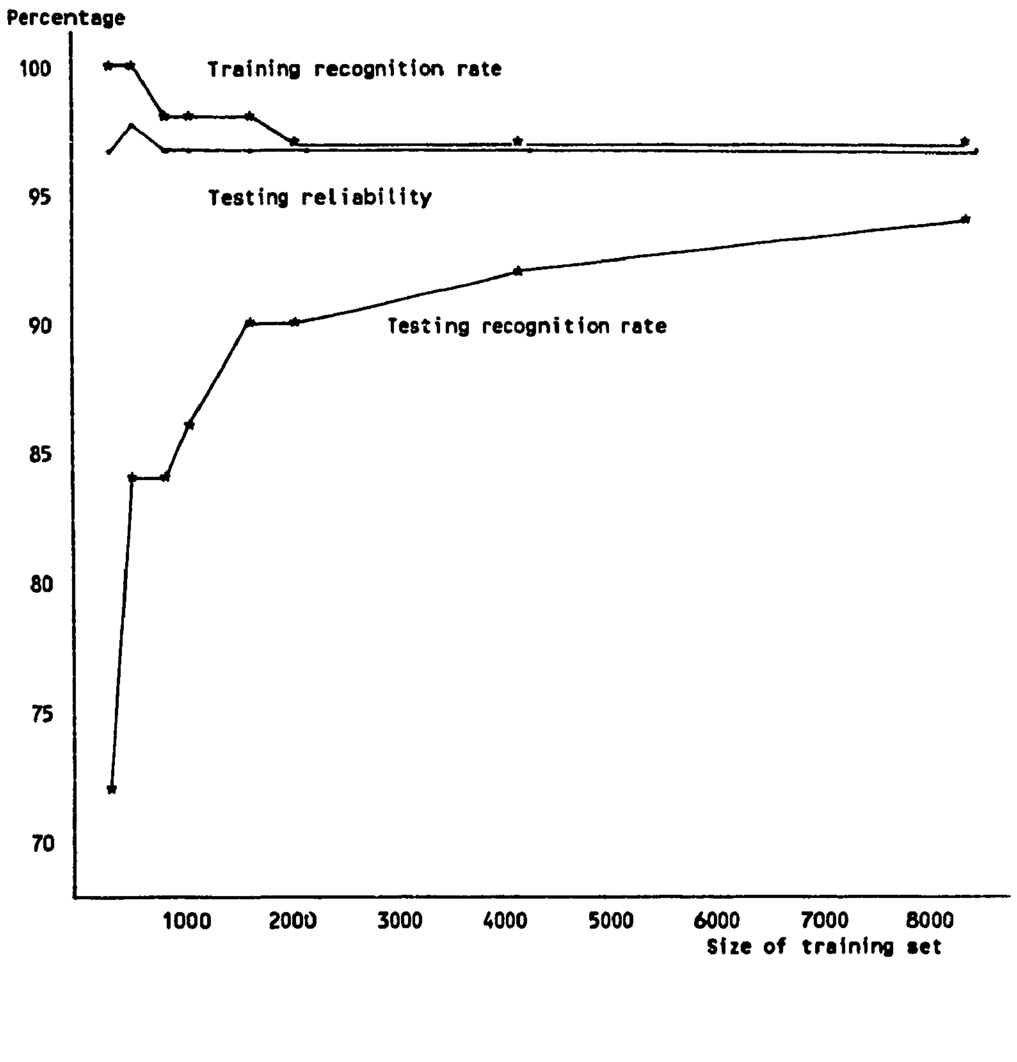
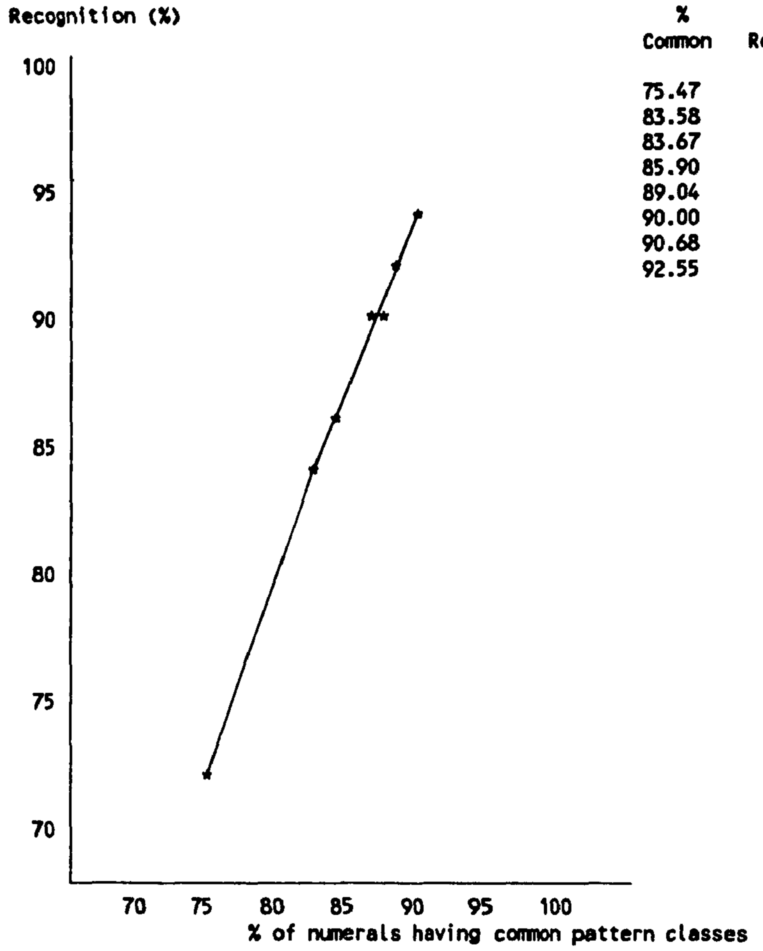


Fig. 8.10 : Relationship between recognition rate
and percentage of samples belonging to common pattern classes



% Common	% Recognition
75.47	72.08
83.58	83.77
83.67	83.96
85.90	86.23
89.04	89.69
90.00	89.77
90.68	92.41
92.55	94.11

8.4 CONCLUSIONS ON EFFECT OF TRAINING SET SIZE

From the above analyses, the size of the training set and the number of writers are important in building a knowledge base that is representative and capable of giving a good recognition rate when applied to unknown samples. A larger training set size will reveal a larger number of features and cover more of the pattern classes that may exist and eventually yield a higher recognition rate.

Furthermore, in the inference method, even when no additional features are added to the knowledge base, an increase in the size of a training set gives more statistical accuracy to the knowledge base and improves the recognition rate. The increase in recognition rate is mainly due to a decrease in rejection rate, but this increase is obtained at the cost of having a lower reliability rate and higher substitution error rate. This is confirmed in the following test:

A knowledge base of hypothesis sets was extracted from a training set of 8500, using only 274 features (out of the total number of features of 407). It was used to recognize a test set of 8485 samples. The same number of features were applied to a training set of 848 and a test set of 848. The recognition rate of the test sets for the first run using larger training set was better than for the second run using smaller training set although the same number of features were used in both runs.

Table 8.11 : Larger training set gives better recognition rate for inference method

Number of features used : 274 (out of 407)

	Training and Test set Size		Difference
	848	8500	
Recognition	83.96%	85.97%	+2.01%
Substitution	2.24%	2.73%	+0.49%
Rejection	13.80%	11.48%	-2.32%
Reliability	97.40%	96.91%	-0.49%

The structural method is entirely dependent on the established pattern classes which depend on the number of features, which depends on the training set. When the size of the training set is larger, it is more likely that the generated pattern classes are better representatives. Hence we also expect the recognition rate to improve when training is done with a larger set.

This behavior is also shown in the performance graphs shown in Figures 10.11 and 10.12 in Chapter 10.

9. PERFORMANCE CURVE AND FINE TUNING

9.1 PERFORMANCE CURVE

4 percentage rates are used to measure the performance of a method: recognition rate, rejection rate, substitution error rate and reliability rate (Chapter 6). Of the 4 measurements, knowing only 2 of them will enable us to derive the remaining 2. Therefore the performance of a recognition method can be conveniently represented as a curve in a 2-dimensional graph using 2 measurements, in this analysis they are substitution error rate versus recognition rate. The inference method and the structural method in this study cover a continuous range of recognition rates, their representative results are plotted in Fig. 9.1. Performance curves of individual numerals, which differ considerably from one to another, are also given in Fig. 9.2. Explanations on how the data are obtained are presented in subsequent chapters.

9.2 FINE TUNING OF THE INFERENCE METHOD

9.2.1 Changing the parameters of the analyzer

By varying the threshold on the minimum percentage of the dominant hypothesis in the final result set in the range of 0 to 100%, it is possible to get different combinations of recognition rate and substitution error rate (Table 9.3).

Fig. 9.1 : Performance curve of the inference method and the structural method

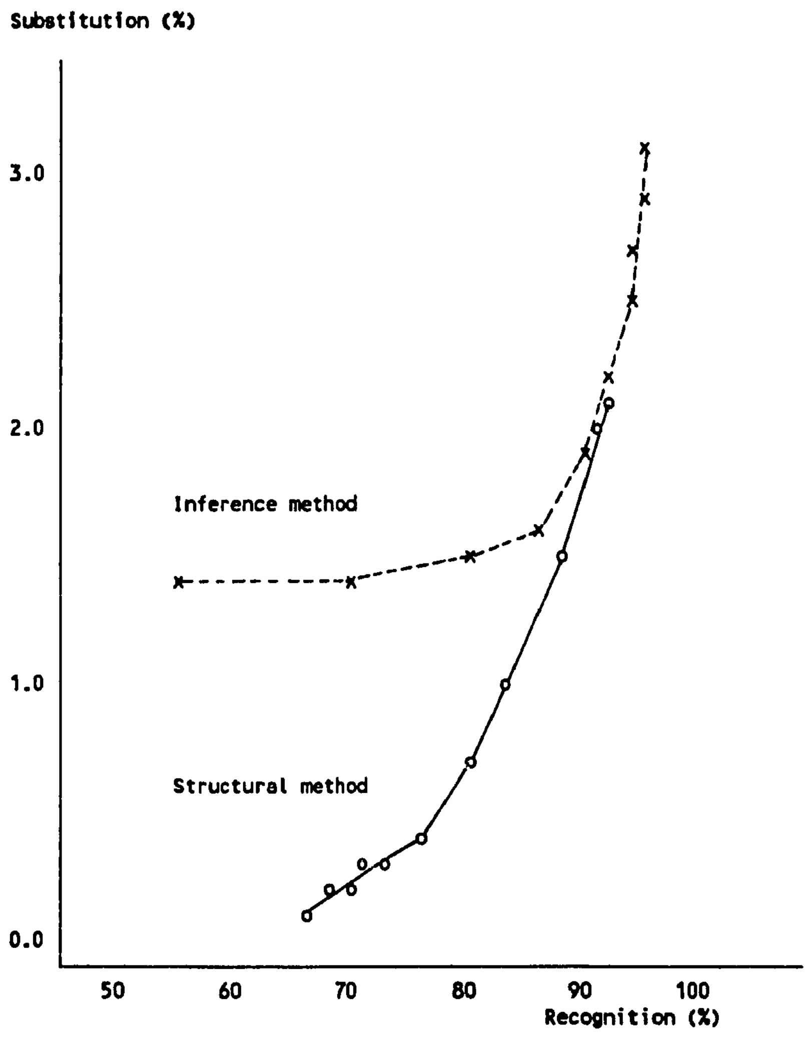


Fig. 9.2 : Performance curves of different numerals structural and inference method

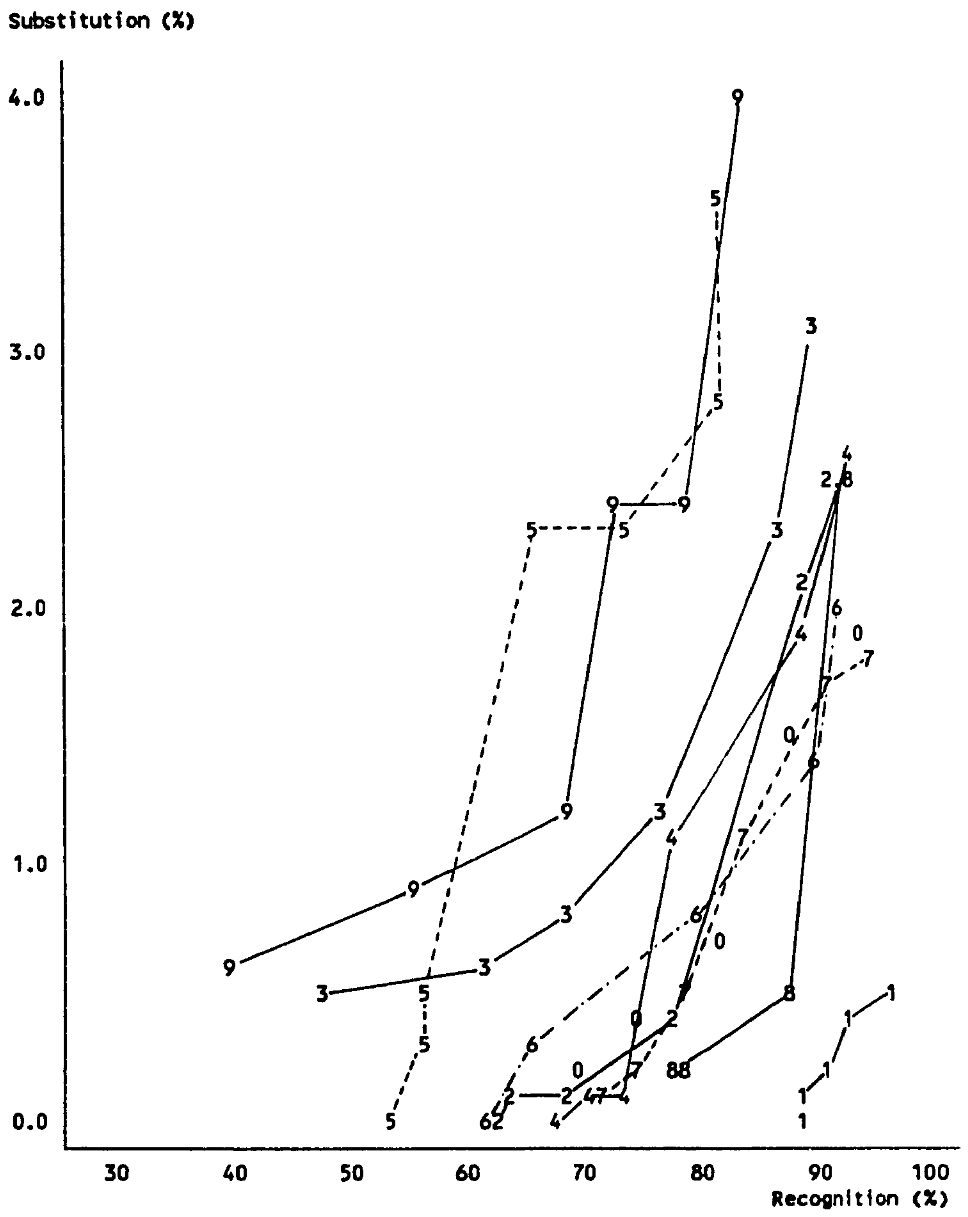


Table 9.3 : Variation in recognition rates using inference method

Test set: 8485 samples

Threshold percent	Rej.	(%)	Subst.	(%)	Recog.	(%)	Reliability. %
1	168	1.98	261	3.08	8056	94.94	96.86
11	168	1.98	261	3.08	8056	94.94	96.86
21	168	1.98	261	3.08	8056	94.94	96.86
31	169	1.99	260	3.06	8056	94.94	96.87
41	188	2.22	246	2.90	8051	94.89	97.04
51	254	2.99	228	2.69	8003	94.32	97.23
61	331	3.90	214	2.52	7940	93.58	97.38
71	449	5.29	188	2.22	7848	92.49	97.66
81	650	7.66	163	1.92	7672	90.42	97.92
91	1050	12.37	134	1.58	7301	86.05	98.20
96	1547	18.23	124	1.46	6814	80.31	98.21
99	2454	28.92	118	1.39	5913	69.69	98.04
100	3624	42.71	116	1.37	4745	55.92	97.61

The minimum substitution error rate that can be achieved with the inference method is 1.37%. The maximum reliability rate that can be achieved is 98.21%, which occurs when the threshold percentage is 96%. At a higher threshold of 100%, although the substitution error rate drops slightly, the recognition rate drops even further, and the maximum reliability rate goes down from 98.21% to 97.61%. The data are plotted in Fig. 9.1.

9.2.2 Exclusion of bad samples from the training set

For recognition using inference method, it is expected that by including poor samples in the knowledge base, it is possible to introduce information that is not representative or even wrong into the knowledge base, but without experience of the poorly written samples, the method will not have the knowledge to deal with them, either to recognize them correctly or to reject them. To see the effect of excluding the poor or unusual samples on the knowledge base, in addition to Set 1 which contains all 8500 samples, training was done on 2 other sets:

- Set 2: 8411 samples, all misrecognized samples were excluded.
- Set 3: 8210 samples, all misrecognized samples and rejected samples were excluded.

The knowledge bases obtained from Set 2 and Set 3 were used in recognition of the test set (8485 samples). The performance curves of 3 runs using the knowledge from 3 sets are given in Figures 9.4 and 9.5.

Fig. 9.4 : Performance curves using different training sets

(Inference method)

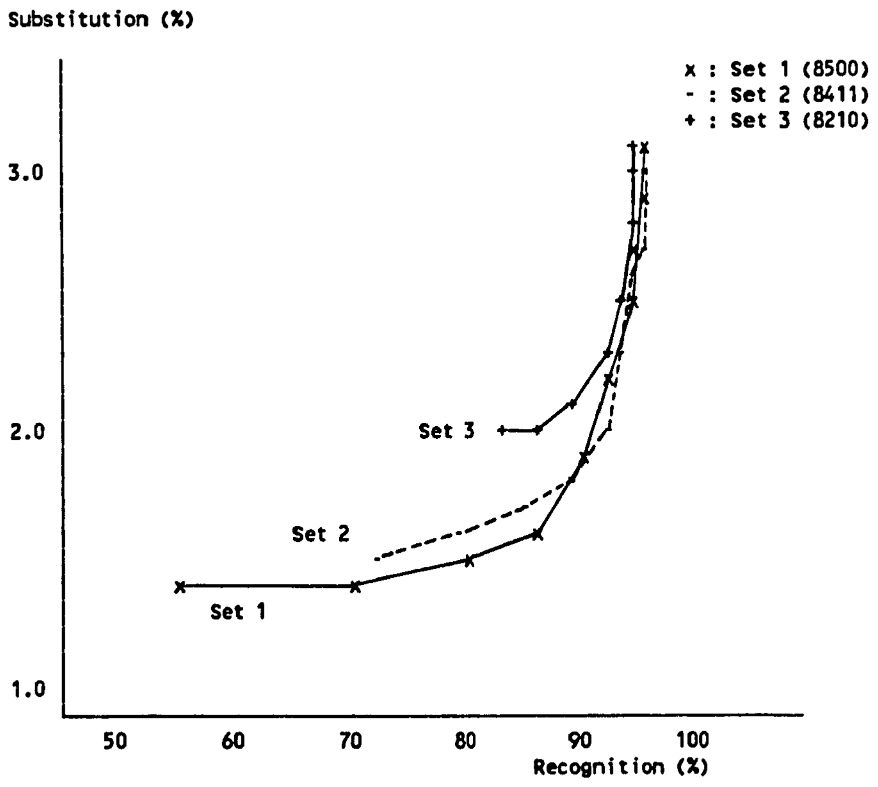
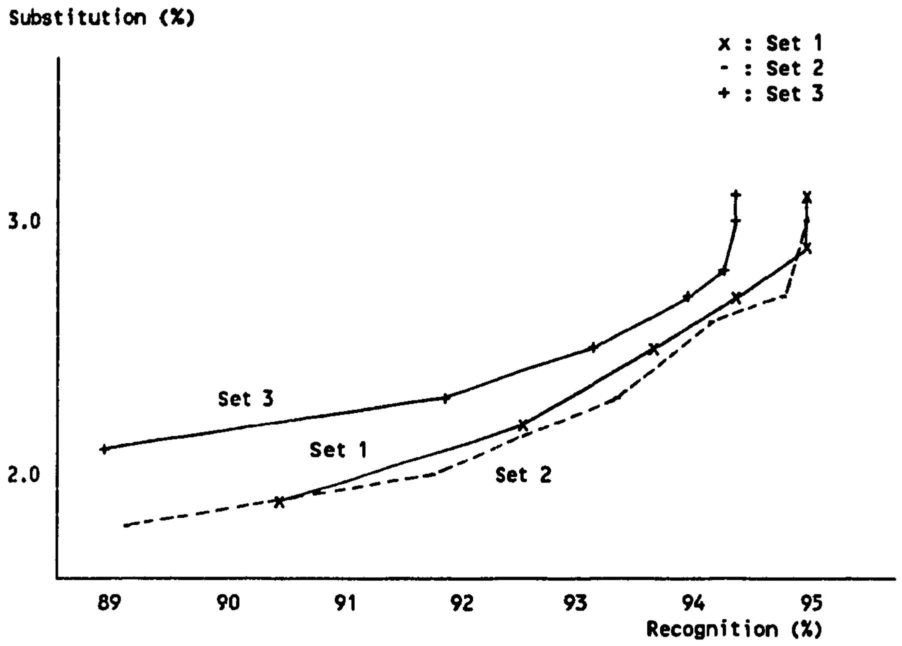


Fig. 9.5 : Comparison of performance curves
at a more detailed scale
(Inference method)



The graphs show that the knowledge base from Set 1 gives better results than Set 2 or Set 3 in most cases. We can see that when bad samples are excluded from the training set, the inference method usually gives a poorer recognition rate. However, in the range of recognition rate of 90-95%, the knowledge base using Set 2 actually gives better recognition rate than Set 1.

Table 9.6 : Set 2 gives better recognition rate at a special range

Training Set	No. of Feat.	Training				Testing			
		Recog.	Sub.	Rej.	Reli.	Recog.	Sub.	Rej.	Reli.
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
All samp.	407	96.59	1.05	2.36	98.93	94.11	2.53	3.36	97.38
Exc.subs errors	407	98.01	0.12	1.87	99.88	94.47	2.49	3.04	97.44
Exc.subs & rej.err.	399	99.67	0.00	0.33	100.00	94.08	2.80	3.11	97.10

Notes: Explanation of columns 2-10 is the same as Table 8.8.

It is also noted that when samples which were misrecognized by the computer were excluded from the training set, the number of observed features decreased and the number of pattern classes generated from the features also decreased considerably. This implies that bad samples mostly belong to the uncommon pattern classes which have a small frequency of occurrence (Table 8.6, Section 8.2).

Table 9.7 : Effect of excluding poor samples from training set on the number of observed features and pattern classes

Training set	No. of observed features	No. of pattern classes				
		Train.	Test.	Total	Combined	Common
(1)	(2)	(3)	(4)	(5)	(6)	(7)
All samples	407	1195	1163	2358	1640	718
Exclude subs. errors	407	1024	1082	2106	1436	670
Exclude subs. & rej. errors	399	751	934	1685	1118	567

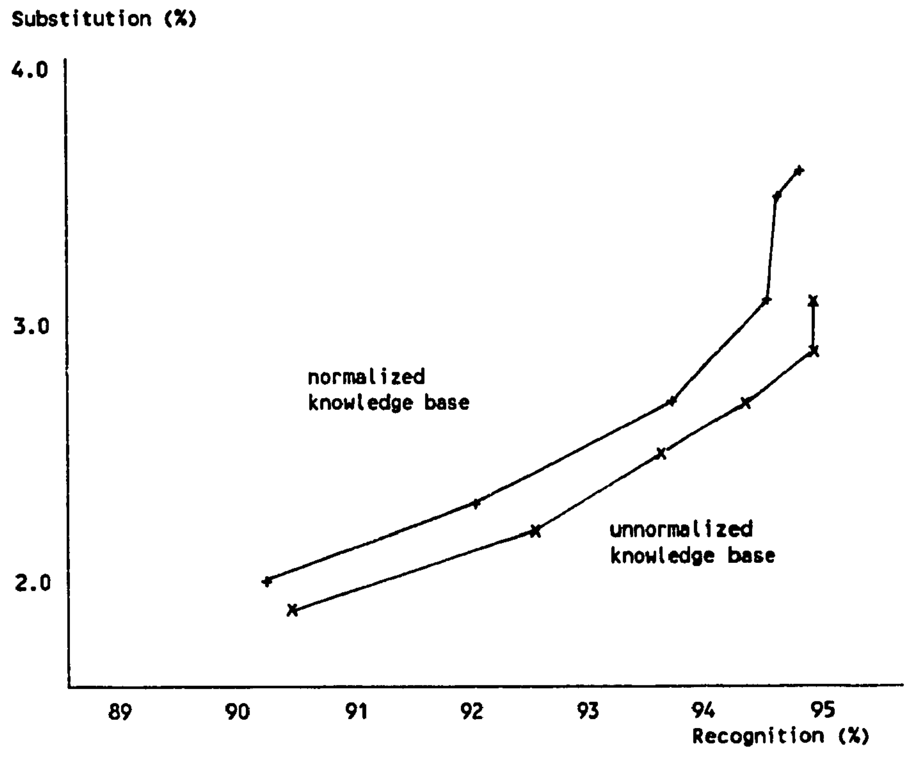
Notes: Explanation of columns 2-7 is the same as Table 8.1.

9.2.3 Effect of a-priori knowledge about the distribution of samples

In equation 5.8, when the a-priori knowledge of distribution of the numerals is not known, each value in a hypothesis set is adjusted by a factor of $1/N_i$ where N_i is the frequency of occurrences of numeral i in the training set.

To exclude this a-priori knowledge, the knowledge base of hypothesis sets was normalized, each occurrence of numeral i was multiplied by the factor of $2000/N_i$ such that the values in hypothesis sets represented occurrence measurements on a training set of 20,000 samples with each numeral having 2000 samples. The normalized knowledge base was then applied to the test set. The recognition rates in comparison with the un-normalized knowledge base are given in Fig. 9.8. In a high recognition range (90-95%), there was a sharp decrease in performance when a-priori knowledge was not used.

Fig. 9.8 : Effect of not using a-priori knowledge of numeral distribution on recognition rate (Inference method)



9.3 FINE TUNING OF THE STRUCTURAL METHOD

9.3.1 Variation of the tuning factor

As explained in Section 5.6, we can select a threshold for the size of pattern class using a tuning factor, varying from 0.0 up to 2.0. The threshold is equal to the average size of a pattern class of a numeral, multiplied by the tuning factor. Recognition rates using different tuning factors are given in Table 9.9 and plotted in Fig. 9.1. The maximum recognition rate on a test set is 91.81% with reliability of

97.78%, the maximum reliability rate can be higher than 99.8% with a recognition rate of less than 65%.

It is possible to increase the tuning factor and obtain a lower substitution error rate. However at some limit, it is not beneficial to increase the tuning factor, because:

- the increase in rejection rate is very high to gain a very slight improvement in substitution rate,
- there are samples in the database which will be misrecognized, even by the best human recognizer, for example the numeral 2 (No.3178) shown in Fig. 2.1 would be recognized as 7 by any human or machine recognizer. This type of error will remain even if the tuning factor were made very high.

Table 9.9 : Recognition results of structural method

Tuning factor	No.of pat. classes	Recog.%	Subst.%	Reject%	Reliability%
0.0	947	91.81	2.09	6.10	97.78
0.1	881	91.09	2.03	6.88	97.82
0.2	606	87.94	1.48	10.57	98.34
0.3	497	86.00	1.23	12.78	98.59
0.4	390	83.25	0.95	15.79	98.87
0.5	348	81.57	0.85	17.58	98.97
0.6	287	79.58	0.71	19.72	99.12
0.7	265	78.24	0.54	21.21	99.31
0.8	219	76.24	0.39	23.37	99.49
0.9	207	75.00	0.37	24.63	99.52
1.0	185	73.46	0.31	26.23	99.58
1.1	167	72.35	0.27	27.38	99.63
1.2	157	71.21	0.27	28.52	99.62
1.3	146	70.29	0.22	29.49	99.68
1.4	140	69.79	0.21	29.99	99.70
1.6	126	68.05	0.15	31.80	99.78
1.8	114	65.61	0.13	34.26	99.80
2.0	101	62.10	0.13	37.77	99.79
2.5	78	57.96	0.09	41.94	99.84
3.0	61	54.44	0.05	45.52	99.91
3.5	51	52.60	0.05	47.35	99.91
4.0	42	49.72	0.02	50.25	99.95
5.0	31	46.36	0.02	53.61	99.95
6.0	24	42.86	0.02	57.22	99.94

9.3.2 Exclusion of poor samples from the training set

By using a threshold on the size of the selected pattern class, the method automatically excludes the poor samples from the training set. The recognition results would be unchanged if poor samples are excluded.

9.3.3 Effect of a-priori knowledge about the distribution of numerals

With structural recognition, the knowledge base of pattern class can be normalized by increasing the size of each pattern class with different multiplication factors so that all numerals have the same frequency of occurrences. However, the effect of this normalization is hidden by a larger change that is caused by using the tuning factor, which is established in a very intuitive fashion and is related to the average size of the pattern class. Hence the effect of normalization can not be isolated and was not considered.

9.4 POSSIBILITY OF IMPROVEMENT

To estimate the possibility of improvement on the method, all substitution errors and rejection errors of the training set were examined. They were divided into 2 types: errors that a human recognizer can also make, and errors that a human recognizer would not make. The division was of course subjective, but it would give an insight to the limitation of the method.

Table 9.10 : Improvement on recognition by a human recognizer

Numeral	Improvement on substitution errors		Improvement on rejection errors		Total errors
	Yes	No	Yes	No	
0	3	8	13	6	30
1	5	3	4	7	19
2	4	6	20	5	35
3	6	6	22	7	41
4	2	2	12	1	17
5	0	1	18	10	29
6	6	5	17	3	31
7	7	10	10	4	31
8	5	5	21	8	39
9	3	2	11	2	18
Total	41	48	148	53	290

The total number of errors that a human recognizer would also make is 101 (48 + 53), the percentage is 1.19% (101 / 8500 = 0.0119). The 189 samples that the human recognizer can recognize are samples having noise such as extra and redundant strokes, white hole inside the original image, touching strokes, filled loop and broken strokes. For these samples, the human recognizer can perform intelligent pre-processing to ignore noise, erase redundant strokes, or add missing strokes. He can also concentrate on strong features, and ignore unimportant features to reach a better decision than a computer algorithm.

Considering the fact that this human recognition test was done by the researcher, who is very familiar with the data and has knowledge of the true identity of the samples, the combined substitution and rejection error rate of about 1.2% is probably biased and tends to be low. If the same experiment were done with an average human recognizer who has no knowledge on the identity of each sample (i.e. the label of each sample is not disclosed), the combined error rate of that human recognizer may be considerably higher than 1.2%.

For comparison, a more objective experiment by Beun [Beun73] with 35 human recognizers on the same kind of data base gave a substitution rate of 1.3% and rejection rate of 0.7%, overall error rate was 2%.

If we want to improve the method to have recognition rate as good as a human recognizer, there are 2 alternatives:

1/ If preprocessing is not improved, additional rules must be added so that the method can recognize the above 189 samples which can be recognized by a human recognizer. Because they are new pattern classes that have a very small size (1.1 - 1.2 per class, as given in Table 6.3, Chapter 6), at least 160 new rules must be added to the knowledge base to generate the new pattern classes. Those rules are not general, and there is no guarantee that they will be effective for an unknown data base. On the other hand, the work required to add an additional 160 rules to a knowledge base that already contains 407 rules is extremely high. We have reached a stage of diminishing return. Therefore, this alternative is not practical, and the building of the knowledge base was stopped at this stage.

2/ The other alternative is to have more intelligent image preprocessing to reduce the number of pattern classes, reduce the number of rules and increase the generality of rules in the knowledge base. This is a direction worth pursuing, but much more study is required to investigate either new or more sophisticated pre-processing techniques.

10. ADDITIONAL EXPERIMENTS ON DATA USED BY CONCORDIA OCR PROJECT TEAM

Further experiments were conducted on 6000 samples used by the Concordia OCR Project Team. Those 6000 samples were extracted from the full database of 16,985 samples used in the main experiment work of this study, however the training and test data from 6000 samples were not necessarily subsets of the training and test data used in the main experiments.

The purpose of the additional experimental work was to compare the recognition results of this method with other methods tested by the OCR Project Team, using the same data. These data were also used in [LAM86] and [LS88].

The 6000 samples were divided into 3 sets: Training set A, Training set B and Test set C, each set contained 2000 samples, 200 of each numeral.

The following runs were performed:

1/ Training on 2000 samples of Training set A.

2/ Testing on 2000 samples of Training set B, using knowledge obtained from training with Set A in Run No. 1.

3/ Training on 4000 samples from Training sets A and B.

4/ Testing on 2000 samples of Test set C, using knowledge obtained from training with Sets A and B.

5/ Testing on 2000 samples of Test set C, using knowledge obtained from training with Set A only.

6/ Testing on 4000 samples of Training sets A and B, using knowledge obtained from training of 8500 samples of the main experimental work.

7/ Testing on 2000 samples of Test set C, using knowledge obtained from training of 8500 samples of the main experimental work.

The confusion matrices for 7 experimental runs are given in Tables 10.1-10.7.

Recognition rates for test set C in Runs 4,5,7 using the structural method are given in Tables 10.8-10.10.

The performance curves of Run 4, Run 5, Run 7 are plotted together with the performance curve of the main run (test set 8485 samples, training set 8500 samples) in Fig. 10.11.

The relationship between substitution error rate and the number of pattern classes used in structural recognition is plotted in Fig. 10.12.

OBSERVATIONS OF ADDITIONAL EXPERIMENTAL WORK

- The recognition results obtained in the above runs are consistent with results reported in the main experimental work. For example:

  + The relationship of recognition rates and the size of training set as plotted in Fig. 8.9 is reproduced in Fig 10.13, the new points coming from additional runs are consistent with the plotted curves.

  + The reliability rates are also above 97%, the substitution rates vary from 2.0 to 2.5, the recognition rates and reject rates are dependent on the size of the training set, the larger the training set, the higher the recognition rate and the lower the reject rate.

+ The performance curves resemble the performance curves of the main experiment. As the size of the training set increases, the curve shifts to the right i.e. better performance (Fig. 10.11).

+ The graph of substitution error rate versus the number of accepted pattern classes plotted in Fig. 10.12 also indicates that better recognition is achieved if training is done with a larger training set.

- The data for runs 3 and 4 are identical to the data used in [LAM86] and [LS88]. This method gave slightly less substitution errors but more rejects than those reported in [LAM86].

Table 10.14: Comparison of substitution errors and rejects with [LAM86]

	TRAINING (4000 samples)			TESTING (2000 samples)		
	This Study	[LAM86] Crit.1	Crit.2	This Study	[LAM86] Crit.1	Crit.2
Sub.	31	86	58	43	63	45
Rej.	94	19	119	98	17	92

Notes: In [LAM86], a set of 3 distance thresholds were used to decide the recognition result as the nearest neighbor to the unknown sample. Criterion 2 defines a lower threshold and therefore is more stringent than Criterion 1.

Table 10.1: Confusion matrix of Run 1
Training using Set A

OUT ->	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	SUCC.	TOTAL
IN														
0			1						3		2	4	194	200
1													200	200
2					1						1	1	198	200
3											2		198	200
4			1								4	1	195	200
5											3		197	200
6											7		193	200
7											4		196	200
8					1						3	1	196	200
9					4						5	4	191	200
TOTAL			2		6				3		31	11	1958	2000

NUMERAL	TOTAL	REJECT	%	SUBST.	%	SUCCESS	%	RELIABILITY
0	200	2	1.00	4	2.00	194	97.00	97.98
1	200	0	0.00	0	0.00	200	100.00	100.00
2	200	1	0.50	1	0.50	198	99.00	99.50
3	200	2	1.00	0	0.00	198	99.00	100.00
4	200	4	2.00	1	0.50	195	97.50	99.49
5	200	3	1.50	0	0.00	197	98.50	100.00
6	200	7	3.50	0	0.00	193	96.50	100.00
7	200	4	2.00	0	0.00	196	98.00	100.00
8	200	3	1.50	1	0.50	196	98.00	99.49
9	200	5	2.50	4	2.00	191	95.50	97.95
TOTAL	2000	31	1.55	11	0.55	1958	97.90	99.44

Table 10.2: Confusion matrix of Run 2
Testing Set B using training knowledge from Set A

OUT -> IN	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	SUCC.	TOTAL
0				1					3		9	4	187	200
1											2		198	200
2				1	2		3		3		20	9	171	200
3			2			1		2			27	5	168	200
4			1					1			9	2	189	200
5				2			2	1			13	5	182	200
6	1				1	2			1		32	5	163	200
7		1		2	1				1	1	10	6	184	200
8	2		1	1		1	1	1		2	7	9	184	200
9					6	1			1		13	8	179	200
TOTAL	3	1	4	7	10	5	6	5	9	3	142	53	1805	2000

NUMERAL	TOTAL	REJECT	%	SUBST.	%	SUCCESS	%	RELIABILITY
0	200	9	4.50	4	2.00	187	93.50	97.91
1	200	2	1.00	0	0.00	198	99.00	100.00
2	200	20	10.00	9	4.50	171	85.50	95.00
3	200	27	13.50	5	2.50	168	84.00	97.11
4	200	9	4.50	2	1.00	189	94.50	98.95
5	200	13	6.50	5	2.50	182	91.00	97.33
6	200	32	16.00	5	2.50	163	81.50	97.02
7	200	10	5.00	6	3.00	184	92.00	96.84
8	200	7	3.50	9	4.50	184	92.00	95.34
9	200	13	6.50	8	4.00	179	89.50	95.72
TOTAL	2000	142	7.10	53	2.65	1805	90.25	97.15

Table 10.3: Confusion matrix of Run 3
Training using Sets A and B

OUT -> IN	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	SUCC.	TOTAL
0			1						4	2	4	7	389	400
1							1				2	1	397	400
2					1				1		12	2	386	400
3								2			11	2	387	400
4			1								7	1	392	400
5							2				7	2	391	400
6					2						10	2	388	400
7				3							10	3	387	400
8					1		1			1	17	3	380	400
9					8						14	8	378	400
TOTAL			2	3	12		4	2	5	3	94	31	3875	4000

NUMERAL	TOTAL	REJECT	%	SUBST.	%	SUCCESS	%	RELIABILITY
0	400	4	1.00	7	1.75	389	97.25	98.23
1	400	2	0.50	1	0.25	397	99.25	99.75
2	400	12	3.00	2	0.50	386	96.50	99.48
3	400	11	2.75	2	0.50	387	96.75	99.49
4	400	7	1.75	1	0.25	392	98.00	99.75
5	400	7	1.75	2	0.50	391	97.75	99.49
6	400	10	2.50	2	0.50	388	97.00	99.49
7	400	10	2.50	3	0.75	387	96.75	99.23
8	400	17	4.25	3	0.75	380	95.00	99.22
9	400	14	3.50	8	2.00	378	94.50	97.93
TOTAL	4000	94	2.35	31	0.77	3875	96.87	99.21

Table 10.4: Confusion matrix of Run 4
Testing Set C using knowledge from training sets A and B

OUT -> IN	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	SUCC.	TOTAL
0			1	1					1	2	10	5	185	200
1						1					8	1	191	200
2				1	1		1		2	1	17	6	177	200
3			3		1	1			1		5	6	189	200
4				1		1		1		1	13	4	183	200
5				2			1				12	3	185	200
6	1		1			1			1		9	4	187	200
7			2	2	1						5	5	190	200
8			2							1	11	3	186	200
9	1		1			1		3			8	6	186	200
TOTAL	2		10	7	3	5	2	4	5	5	98	43	1859	2000

NUMERAL	TOTAL	REJECT	%	SUBST.	%	SUCCESS	%	RELIABILITY
0	200	10	5.00	5	2.50	185	92.50	97.37
1	200	8	4.00	1	0.50	191	95.50	99.48
2	200	17	8.50	6	3.00	177	88.50	96.72
3	200	5	2.50	6	3.00	189	94.50	96.92
4	200	13	6.50	4	2.00	183	91.50	97.86
5	200	12	6.00	3	1.50	185	92.50	98.40
6	200	9	4.50	4	2.00	187	93.50	97.91
7	200	5	2.50	5	2.50	190	95.00	97.44
8	200	11	5.50	3	1.50	186	93.00	98.41
9	200	8	4.00	6	3.00	186	93.00	96.87
TOTAL	2000	98	4.90	43	2.15	1859	92.95	97.74

Table 10.5: Confusion matrix of Run 5
Testing Set C using knowledge from training set A only

OUT -> IN	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	SUCC.	TOTAL
0			1	1		1			3		14	6	180	200
1											3		197	200
2				3			5		2	1	18	11	171	200
3			2		1	1			1		18	5	177	200
4								1		4	17	5	178	200
5				2			1				15	3	182	200
6			1			1			1		15	3	182	200
7			2	1	1						11	4	185	200
8				1						1	9	2	189	200
9	1					1		1	1		12	4	184	200
TOTAL	1		6	8	2	4	6	2	8	6	132	43	1825	2000

NUMERAL	TOTAL	REJECT	%	SUBST.	%	SUCCESS	%	RELIABILITY
0	200	14	7.00	6	3.00	180	90.00	96.77
1	200	3	1.50	0	0.00	197	98.50	100.00
2	200	18	9.00	11	5.50	171	85.50	93.96
3	200	18	9.00	5	2.50	177	88.50	97.25
4	200	17	8.50	5	2.50	178	89.00	97.27
5	200	15	7.50	3	1.50	182	91.00	98.38
6	200	15	7.50	3	1.50	182	91.00	98.38
7	200	11	5.50	4	2.00	185	92.50	97.88
8	200	9	4.50	2	1.00	189	94.50	98.95
9	200	12	6.00	4	2.00	184	92.00	97.87
TOTAL	2000	132	6.60	43	2.15	1825	91.25	97.70

Table 10.6: Confusion matrix of Run 6
Testing Sets A and B using training knowledge from main experiment

OUT -> IN	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	SUCC.	TOTAL
0			1	2	1		1			1	5	6	189	200
1						1					1	1	198	200
2	1			1			1		1		7	4	189	200
3			2		1	1			1		3	5	192	200
4			4	2				2	1	1	7	10	183	200
5				4			1	1			9	6	185	200
6			2			1			1		7	4	189	200
7			3	2	1						2	6	192	200
8	1									1	8	2	190	200
9			1	1				2			10	4	186	200
TOTAL	2		13	12	3	3	3	5	4	3	59	48	1893	2000

NUMERAL	TOTAL	REJECT	%	SUBST.	%	SUCCESS	%	RELIABILITY
0	200	5	2.50	6	3.00	189	94.50	96.92
1	200	1	0.50	1	0.50	198	99.00	99.50
2	200	7	3.50	4	2.00	189	94.50	97.93
3	200	3	1.50	5	2.50	192	96.00	97.46
4	200	7	3.50	10	5.00	183	91.50	94.82
5	200	9	4.50	6	3.00	185	92.50	96.86
6	200	7	3.50	4	2.00	189	94.50	97.93
7	200	2	1.00	6	3.00	192	96.00	96.97
8	200	8	4.00	2	1.00	190	95.00	98.96
9	200	10	5.00	4	2.00	186	93.00	97.89
TOTAL	2000	59	2.95	48	2.40	1893	94.65	97.53

Table 10.7: Confusion matrix of Run 7
Testing Set C using training knowledge from main experiment

OUT -> IN	0	1	2	3	4	5	6	7	8	9	REJECT	SUBST.	SUCC.	TOTAL
0			1				1		4		7	6	387	400
1							1				1	1	398	400
2				5	1				1	1	12	8	380	400
3		1						2			11	3	386	400
4			1					2			4	3	393	400
5				2			2				23	4	373	400
6	1		1		2				2		8	6	386	400
7		1	1	4					2	2	6	10	384	400
8	2		2		1		1				10	6	384	400
9	1		3		9			2			20	15	365	400
TOTAL	4	2	9	11	13		5	6	9	3	102	62	3836	4000

NUMERAL	TOTAL	REJECT	%	SUBST.	%	SUCCESS	%	RELIABILITY
0	400	7	1.75	6	1.50	387	96.75	98.47
1	400	1	0.25	1	0.25	398	99.50	99.75
2	400	12	3.00	8	2.00	380	95.00	97.94
3	400	11	2.75	3	0.75	386	96.50	99.23
4	400	4	1.00	3	0.75	393	98.25	99.24
5	400	23	5.75	4	1.00	373	93.25	98.94
6	400	8	2.00	6	1.50	386	96.50	98.47
7	400	6	1.50	10	2.50	384	96.00	97.46
8	400	10	2.50	6	1.50	384	96.00	98.46
9	400	20	5.00	15	3.75	365	91.25	96.05
TOTAL	4000	102	2.55	62	1.55	3836	95.90	98.41

Table 10.8 : Recognition results of structural method
Run 4: Testing Set C using knowledge from training sets A and B

TUNING FACTOR	NO.OF PAT. CLASSES	RECOG.%	SUBST.%	REJECT%	RELIABILITY%
0.0	529	91.00	1.90	7.10	97.95
0.1	508	90.40	1.85	7.75	97.99
0.2	405	87.55	1.45	11.00	98.37
0.3	343	85.75	1.15	13.10	98.68
0.4	276	82.35	0.85	16.80	98.98
0.5	227	80.00	0.80	19.20	99.01
0.6	205	78.30	0.75	20.95	99.05
0.7	183	76.25	0.70	23.05	99.09
0.8	151	71.55	0.60	27.85	99.17
0.9	145	70.85	0.50	28.65	99.30
1.0	118	68.05	0.50	31.45	99.27
1.1	109	66.65	0.45	32.90	99.33
1.2	102	65.65	0.40	33.95	99.39
1.3	97	64.75	0.40	34.85	99.39
1.4	91	63.90	0.40	35.70	99.38
1.6	82	62.85	0.35	36.80	99.45
1.8	73	60.50	0.35	39.15	99.42

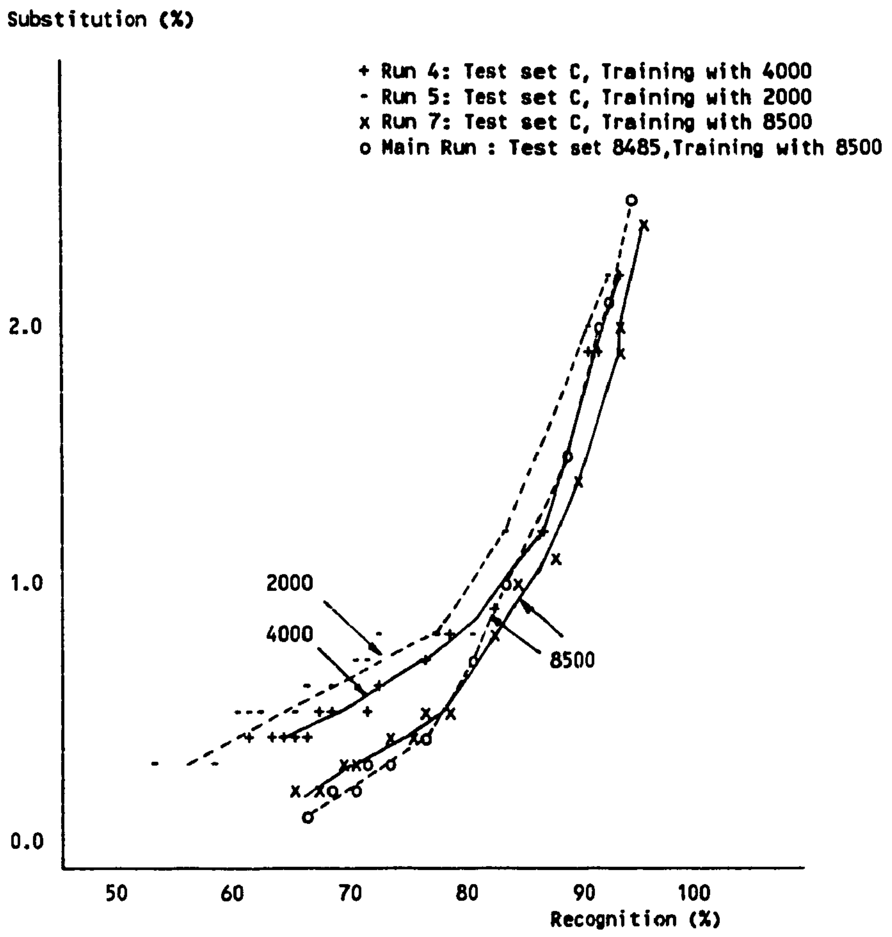
Table 10.9 : Recognition results of structural method
Run 5: Testing Set C using knowledge from training set A only

TUNING FACTOR	NO.OF PAT. CLASSES	RECOG.%	SUBST.%	REJECT%	RELIABILITY%
0.0	362	89.85	2.00	8.15	97.82
0.1	360	89.85	2.00	8.15	97.82
0.2	314	87.80	1.45	10.75	98.38
0.3	235	83.00	1.20	15.80	98.57
0.4	209	79.60	0.80	19.60	99.00
0.5	184	77.40	0.75	21.85	99.04
0.6	143	72.45	0.75	26.80	98.98
0.7	131	70.95	0.65	28.40	99.09
0.8	121	69.60	0.65	29.75	99.07
0.9	106	67.85	0.60	31.55	99.12
1.0	98	66.35	0.60	33.05	99.10
1.1	90	64.95	0.45	34.60	99.31
1.2	82	62.35	0.45	37.20	99.28
1.3	75	60.85	0.45	38.70	99.27
1.4	71	59.80	0.45	39.75	99.25
1.6	60	57.90	0.30	41.80	99.48
1.8	50	53.45	0.25	46.30	99.53

Table 10.10 : Recognition results of structural method
Run 7: Testing Set C using training knowledge from main experiment

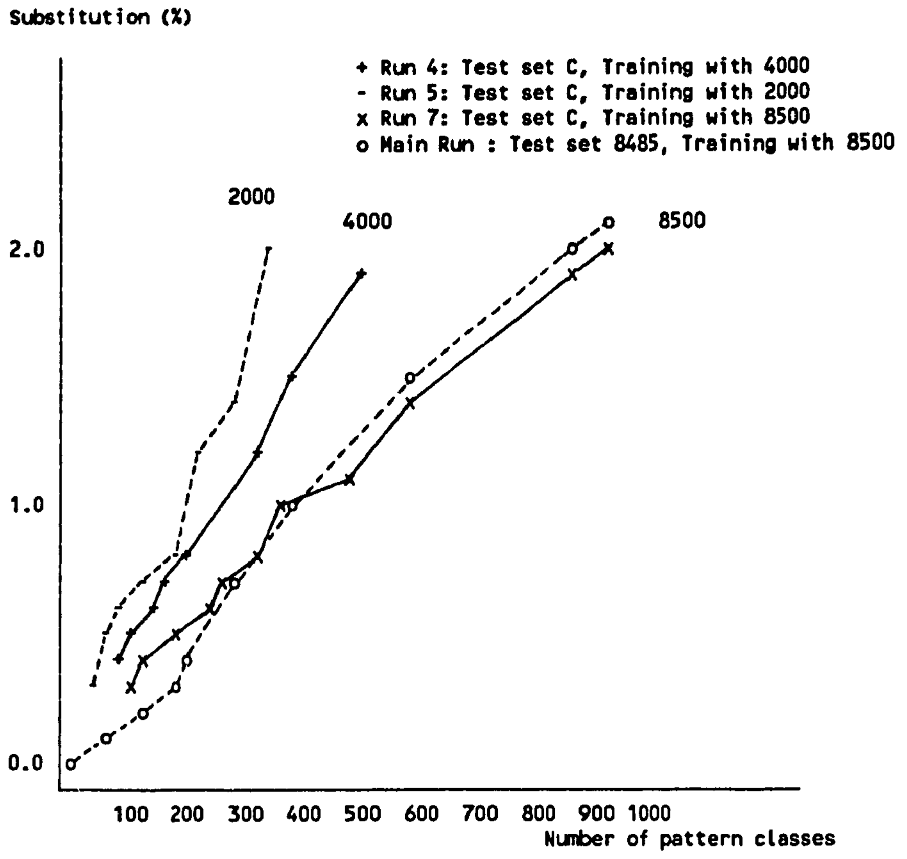
TUNING FACTOR	NO.OF PAT. CLASSES	RECOG.%	SUBST.%	REJECT%	RELIABILITY%
0.0	947	93.05	2.00	4.95	97.90
0.1	881	92.90	1.90	5.20	98.00
0.2	605	89.40	1.35	9.25	98.51
0.3	497	86.85	1.05	12.10	98.81
0.4	390	84.20	0.95	14.85	98.88
0.5	348	82.20	0.80	17.00	99.04
0.6	287	79.55	0.70	19.75	99.13
0.7	265	78.35	0.50	21.15	99.37
0.8	219	75.95	0.45	23.60	99.41
0.9	207	74.75	0.40	24.85	99.47
1.0	185	72.75	0.40	26.85	99.45
1.1	167	71.20	0.30	28.50	99.58
1.2	157	69.95	0.30	29.75	99.57
1.3	146	69.30	0.30	30.40	99.57
1.4	140	68.50	0.25	31.25	99.64
1.6	126	66.55	0.20	33.25	99.70
1.8	114	64.85	0.15	35.00	99.77

Fig. 10.11: Comparison of performance curves of different runs



As the size of the training set increases, the performance curve shifts to the right i.e. better performance.

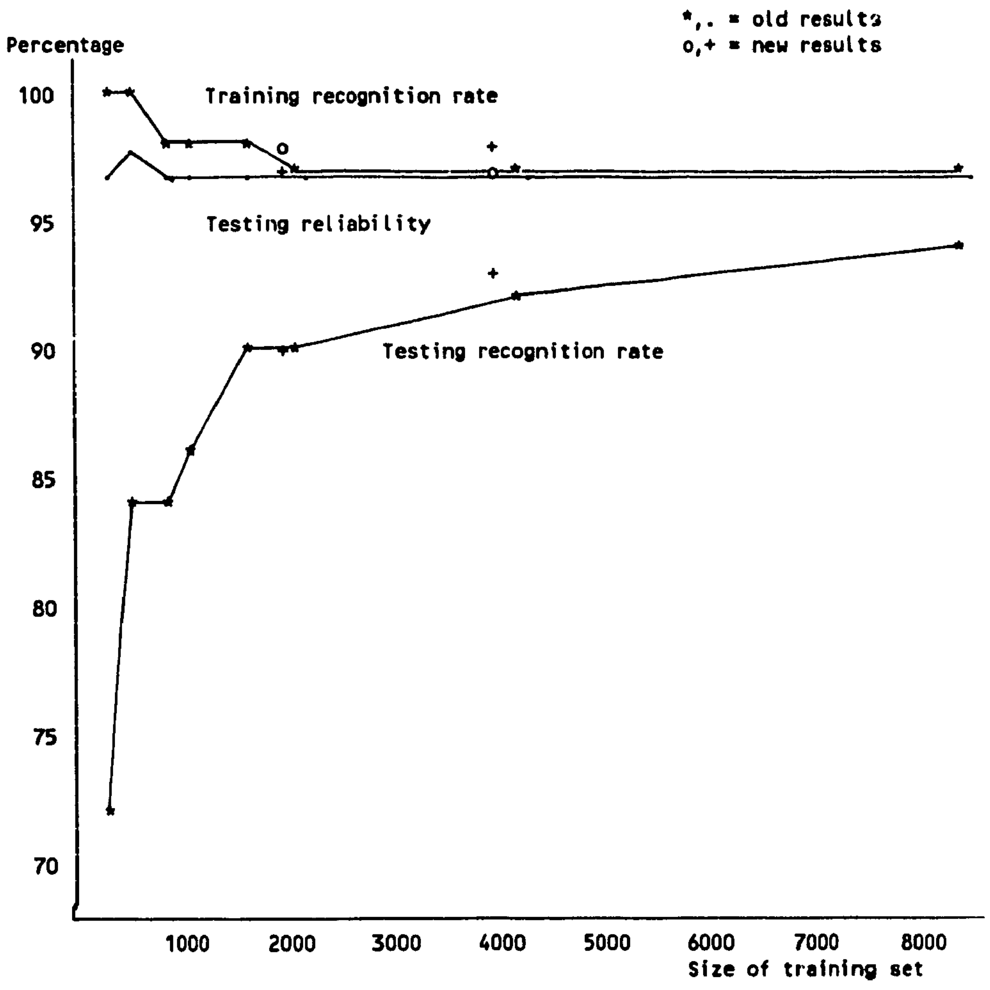
Fig. 10.12: Relationship between the number of pattern classes and substitution error rate



Substitution error rate increases when the knowledge base accepts more pattern classes with smaller frequency of occurrence from the training set.

For the same number of accepted pattern classes, the knowledge base from a training set with larger size produces less misrecognition than that from a smaller set.

Fig. 10.13: Effect of the size of training set on the recognition rates



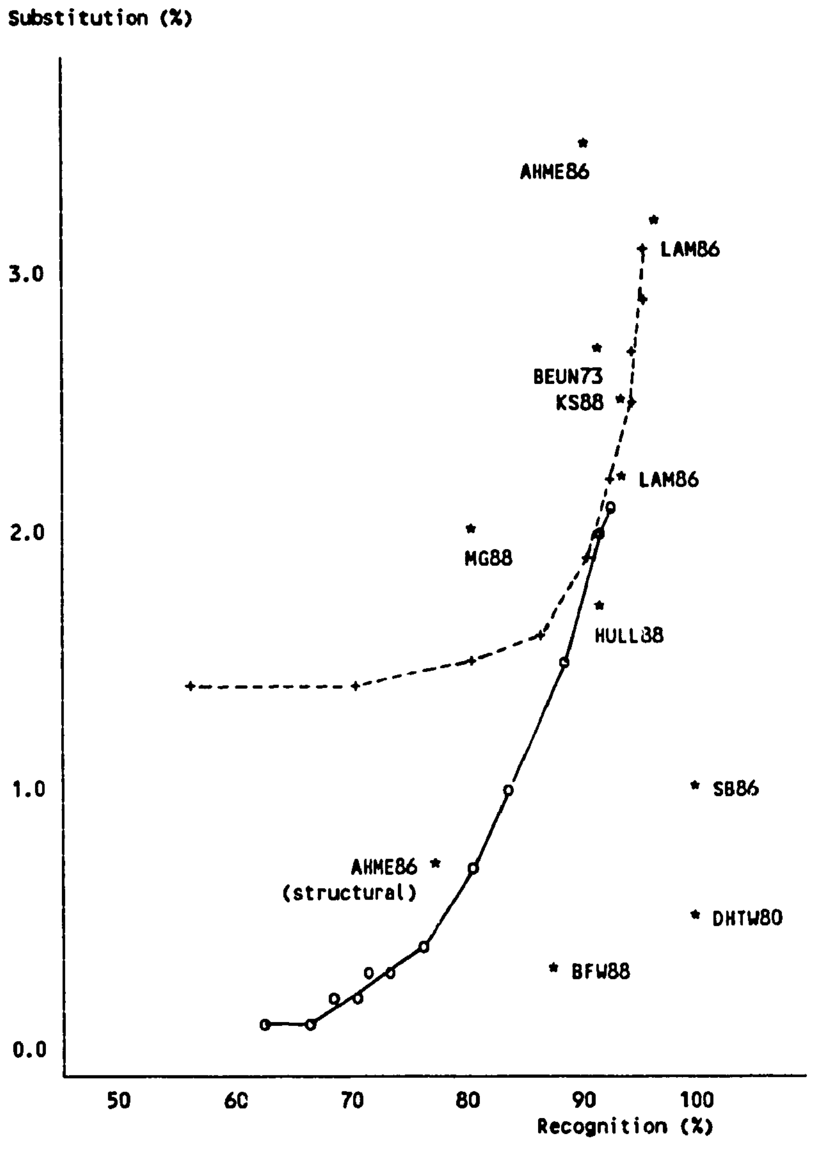
11. COMPARISON WITH OTHER STUDIES

11.1 COMPARISON

It has been noted by researchers in this field that it is not easy to compare the results of one piece of work with those of others [AHME86]. The main reason is that the quality of the data used by different studies cannot be measured accurately. Another reason is that the degrees of complexity in shape of different numerals are not the same, for example numerals 0, 1, 6 and 7 have simpler shapes and usually have less substitution errors than others (see Fig. 9.2). By using a higher proportion of easy-to-recognize numerals than others in the test set, the overall recognition rate can become 'much better'. Instead of using the percentage of substitution rate over the entire test set, the normalized value, being the average of substitution rates of all numerals will give a more representative measurement. Unfortunately, most works used test sets that do not have equal distribution of numerals, some authors did not provide details about the distribution or did not list the substitution rates for individual numerals.

A summary of results from selected studies on recognition of unconstrained handwritten characters is given in Table 1.1. Their recognition rates are plotted together with the representative results of this study in the form of a performance graph in Fig. 11.1. The plotted rates from this study are the overall, unnormalized rates because a majority of other studies also used unnormalized rates. Because of the limitations cited above on data quality and data distribution, the recognition rates in this graph cannot be compared in a meaningful way. Therefore, the purposes of a comparison with other studies are:

Fig. 11.1: Performance graph comparison of selected works



Notes: * Results of other studies
+ Results of this study using inference method
o Results of this study using structural method

- to identify any similar trend in the studies in order to find some general strategy in character recognition.
- to assess the potential of different approaches and to see whether it is possible to estimate a realistic goal that can be achieved.

Fig. 11.1 indicates that the recognition rates from this study are nearly the same as most of other works. There are several studies that cited better recognition rates than this study, they are discussed briefly below:

- **Shridhar and Badreldin** [SB86] used 2 methods, one method employed Fourier descriptors as features and a distance classifier as the recognition method, the other method used geometrical measurements and profile distances as features and a structural recognition method. Both methods achieved a recognition rate of about 99%. However, the number of writers was only 20, the size of the data base of isolated characters used for training was 400 and for testing 450. In the same work, the authors also used a data base of 10,000 continuous samples of 3 digits (total 30,000 digits) but this data base was used only for the authors' research on recognition of connected characters.

- **Duerr et al** [DHTW80] achieved a remarkable recognition rate of 99.50% using a 4-stage classifier:

  1/ A statistical classifier using Karhunen-Loeve expansion,

  2/ A structural classifier and hypothesis generator which analysed the topological composition of the characters,

  3/ A reduction stage which reduced the number of hypotheses made from the first 2 stages using contour information, 4/ A mixed stage which performed a heuristic search for maximum similarity between the character and syntactically-generated prototypes.

The data base had 10,000 characters written by 200 persons and was divided equally into a training set and a test set. Samples of data given in the paper showed that the data base contained unconstrained characters, some of which were also of poor quality. 95% of the numerals were recognized by the first 2 stages, 1% were correctly recognized by the third stage, and 4% were passed to the 4th stage, of which 3% were recognized by this stage, the remaining 1% were recognized using the result from the first stage.

The size and quality of the data base and the reliability of the method make this method a very interesting one. The reliability was improved by using a large number of features: topological, contour, mathematical transformation and more than one recognition method.

- **Brown et al** [BFW88] achieved a recognition rate of 97% with a substitution rate of only 0.3% and rejection rate of 2.7%. The authors used an additional class of 'thrash' which consisted of rejected characters, this class covered 10.4% of the data base, so in effect the rejection rate was 13.1% and the true recognition rate was only 86.6%. Even at the adjusted recognition rate of 86.6%, the accuracy of this method is very good. The most interesting points in this paper are several concepts and techniques in thinning, including the use of expert sub-modules to achieve the best skeleton. However, the high recognition rate was also partly attributed to the good quality of the characters, because they were written by the people in the mapping and charting profession on maps and chart documents, and can be expected to have a better quality than that of characters written on envelopes by the general public. The size of the data base was 7000, the number of writers was not given, but the authors mention that 6612 samples came from 5 mapping documents, and 388 were collected from different writers. The authors also did not mention about testing procedure, most importantly, whether the result was obtained from the training set or the test set.

For studies that cited recognition rates of 98-99%, this author has some reservations that they might not use totally unconstrained handwritten characters, or that the recognition rates could not be repeated with other test sets. The reasons are:

- Most studies on totally unconstrained characters achieved a recognition rate of only about 95-96%.
- Many papers on unconstrained character recognition quoted a recognition rate of 96-98% for human recognition, and we do not expect that at this point in time the computer will achieve a recognition rate better than human beings. In Beun's work [Beun73], an experiment was conducted using 35 persons to recognize 840 unconstrained handwritten numerals. The results from this experiment were: recognition rate 98%, substitution rate 1.3%, and rejection rate 0.7%. Another estimate of human recognition error rate of 1.2 % comes from this study (Section 9.4).

11.2 SOME GENERAL OBSERVATIONS

1/ Most methods that deal with unconstrained characters did not achieve a recognition rate much higher than 95% or a substitution rate less than 2.0%. The methods that achieved a high recognition rate usually applied a combination of 2 or more methods, for example:

[LAM86]	:	structural and relaxation matching,
[AHME86]	:	shape-based table and statistical method,
[HULL88]	:	voting from 3 methods : structural, structural combined with statistical, and template matching.
[DHTW80]	:	4-stage classifier using structural, statistical and heuristic matching.

2/ Usually, with the same data and within the same research work, the structural method has a higher reliability rate than other methods, although the recognition rate may be lower.

Table 11.2 : Comparison of reliability rates

	Structural	Other	Note
Ahme86	99.06	88.55	statistical
Hull88	96.4	94.8	mixed structural & statistical
		88.6 - 96.8	template matching
Lam86	97.75	92.92	relaxation matching
This study	99.98 max.	98.2 max.	inference (statistical)

A detailed comparison between the statistical and the structural method used in this study can be found in Section 5.6.

3/ Regardless of the method used (structural, statistical or template matching), the most important requirement is that the method must be able to distinguish a number of defined pattern classes. If the pattern classes are distinct and can be easily distinguished from one another, all the 3 recognition methods will yield an accurate recognition. As for an estimation of the number of pattern classes, some data can be found from [LAM86] or this study. In Lam's work [LAM86], in addition to the structural decision trees, an additional 242 masks (or pattern classes) were used. In this study, a total of 1195 pattern classes were used, of which there were less than 718 common pattern classes that are expected to be found in another distinct set (see Table 9.1, column (7)).

The pattern classes are defined by the features, and if the features are optimally selected, there will be a smaller number of pattern classes which can cover a majority of all possible input characters.

Therefore, **feature extraction is really the most important stage of the whole recognition process.** Feature extraction is dependent on preprocessing, and an **integration of preprocessing and feature extraction** also promises to be beneficial. A large proportion of samples that were misrecognized or rejected in this study came from noisy patterns (presence of extra strokes, extra holes, touching strokes). If the preprocessing is driven by knowledge from the feature extraction stage and the knowledge is applied to 'retouch' and enhance the pattern in an appropriate way, then the product will be a pattern which has a simpler shape and possessing the most important and distinctive features. This is the approach used by Brown et al [BFW88] which can be enhanced and improved further in future research.

As for the recognition methodology, if there is a good feature extraction, the choice of the recognition method is not an important issue. It has been demonstrated in this study that using the same feature extraction process, different recognition methods can be used to achieve the desired recognition rate or reliability rate. In fact, as indicated in Chapter 5, the boundary between statistical methods and structural methods is not absolutely clear. The criterion to identify a structural feature is also based on a statistical threshold.

4/ This researcher believes that the cooperation between human knowledge and computer's capability of processing and organizing data is important in obtaining the most useful features. Human knowledge serves as a guidance in feature selection and the computer serves as a organizer of data so that the fuzzy boundary between different pattern classes can be optimally defined.

5/ For practical implementation, reliability is more important than the recognition rate, as long as the recognition rate is acceptable. Computer recognition is only acceptable if it is at least as reliable as human reader. With less reliability, human reader is required to check the computer's recognition results, and hence there is no benefit to be gained by using computer for the recognition task. Because reliable data coming from a careful and large scale investigation are not available, let us assume that the human recognition rate in Beun's work (98% success, 1.3% error) is more or less reliable, then a realistic goal of a computer recognition method is to have 1.3% substitution error rate, not 98% recognition rate. From the performance graph in Fig. 11.1, the structural method in this study can deliver the substitution rate of 1.3% with the corresponding recognition rate of about 85%, which is an acceptable rate. In fact, a recognition rate as low as 60% is also acceptable because it already represents a saving of 60% of manual labor. The other important requirement is that the implementation must be less costly than using a human reader and has at least the same speed as a human reader.

From this argument, we can see that the goal of using computer for the recognition of real-life handwritten characters is realistic and feasible. However, there is a change in the meaning of the task, it is no longer the recognition of unconstrained characters, but really the recognition of the portion of input characters that are well-formed and easy to recognize.

12. CONCLUSIONS

12.1 SUMMARY

In this study, a training method was implemented, which aimed at minimizing human subjectiveness in the selection of the training set and features. Training was done concurrent to the recognition process, the recognition results were used as feedback to the training process to help the human expert to select further features. In that sense, the analysis of training data and the development of a knowledge base of features went hand in hand and cooperated with each other, and the knowledge base was built in a heuristic and incremental manner. The selection of primary features was systematic and done by the computer, but the selection of secondary features was dependent on the training data and judgement of the human expert.

The knowledge obtained in training was organized in 2 ways: it contained entries of hypothesis sets, each hypothesis set belonged to one feature, or it contained entries of pattern classes, each pattern class corresponded to a different feature vector.

An inference method using set operations was used with the knowledge base of hypothesis sets. It achieved the maximum recognition rate of 94.9% with a reliability of 97%. A structural method was used with the knowledge base of pattern classes, which gave a high reliability rate of 99.8% but the recognition rate was only 65%, which is much lower than that achieved by human. To have the same substitution error rate as a human recognizer which is assumed to be about 1.3%, the structural method would give a recognition rate of about 85%.

In contrast with the majority of studies in literature, this method analyzed the input data in more detail and obtained a number of quantitative findings on the problem of recognition of unconstrained handwritten numerals. Coming from experimentation with a data base of nearly 17,000 samples, the findings are more suggestive than conclusive:

- The varieties of shapes of numerals generated by unconstrained handwriting is infinite, but there appears to be a limit on the classes of shapes that occur more frequently, this limit is estimated to be a few hundreds up to less than a thousand.

- The substitution and rejection errors in computer recognition is caused by the fact that it is never possible to establish threshold values that segregate the samples into distinct and well-defined classes. If high reliability is desired, then for each feature, 2 extreme thresholds are used, one to accept a sample to a pattern class, the other to reject a sample from a pattern class. Samples having measurements in the grey area between the 2 thresholds are distinguished by other features and may be rejected as unrecognizable depending on the combined result of feature extraction. With this philosophy, it is possible to design methods with high reliability, but the recognition rate is more limited.

- It is possible to include all samples, including the non-representative and poorly-written samples into training. The odd samples are minority facts in the constructed knowledge base and will be disregarded using minimum threshold values. The advantage of this approach is that the designed system is more objective and more immune to poor data. The experimental results in this study showed that recognition results in general do not become any worse when all poor samples are included in training.

- Structural methods tend to give more reliability and have less computing cost than a statistical method using the same feature extraction, but its recognition rate may be lower. This observation comes from the recognition results of the 2 methods in this study, and also from the recognition results reported in other research papers (see Note 2, Section 11.2).

12.2 ASSESSMENT OF METHOD

The method proposed in this study is a hybrid implementation of many methods that have been reported in the literature including the structural method, fuzzy set concept, artificial intelligence concepts of expert system and machine learning. The salient features of the method are:

- It behaves like an expert system by storing all knowledge obtained from training and using this knowledge to infer the recognition of an unknown sample.

- It is heuristic and objective in nature. The rules established by the human expert are used by the algorithm to learn and build the knowledge base. The human expert does not subjectively exclude or include a certain sample to the training set, but he indirectly controls the quality of the knowledge base by adding only features that are judged general, consistent and valid.

- The inference method allows fuzzy classification, if there are many recognition alternatives, the likelihood of each alternative is evaluated and the alternative with the greatest likelihood is selected as the recognition result.

- The knowledge base is organized in a flexible way which enables feature extraction and inference to proceed partially in parallel. The knowledge base of pattern classes can be organized as a decision tree with one or many levels.

- The method can be tuned to achieve different combinations of recognition rate and reliability rate.

- The achieved recognition rates are comparable to those obtained from existing methods in this field, while the computing cost is less than most other methods.

The neural network model and the expert system model have influenced the initial design of this method. At the end, the method turns out to be different from the neural network and expert systems model in the following ways:

- The recent research works using the neural network approach consider only machine intelligence, and take no advantage of human intelligence. Because of the lack of human intelligence, the recognition rate of the neural network is low and the computation time is high. The computer, left to itself, has a difficult task of connecting the array of pixels it receives with the recognition result that it has to produce. By introducing a very small direction from the human expert, like using the sequence of special points as discussed in Section 4.1, the computer was able to reach the same recognition rate as the neural network method with much less processing effort.

- The inference method is a simple expert system which imitates human process in recognition. It does not have a complicated relationship between rules like most expert systems, and uses only forward chaining, because the problem is too simple to apply backward chaining. A distinct feature in this method is the fact that the computer is used as a tool to find the solution. The solution is not purely a human task, but a co-operating effort between computer and human expert.

A number of points that can be considered as weaknesses in this method are:

- There is a lack of elegance in the training stage where the secondary features are proposed, tested and refined; but it is felt that this is the only way to obtain a knowledge base that is detailed and sophisticated enough to handle real-life unconstrained characters.

- The construction of the knowledge base is the work of a single human expert. With contribution from many experts in character recognition, better features could be selected, more discriminative knowledge base could be built which would result in a better recognition rate.

- Like most other recognition methods, the recognition performance of this method is dependent on the training samples and the preprocessing methods used. As more varieties of training samples are used, the knowledge base will be more sophisticated, however new features must be included to achieve a higher level of sophistication. Because many features depend on the skeleton and the original image, when the preprocessing method is changed, the threshold values of the features must be readjusted to co-operate better with the new preprocessing method in getting the maximum recognition rate.

- There is complicated dependency between features. A newly introduced feature will create unknown side effects with previous features. Therefore, building a consistent knowledge base is tedious and time-consuming; the determination of the dependency between features and the order of applying rules is not systematic and not objective.

12.3 DIRECTIONS FOR IMPROVEMENT

12.3.1 Data base

In Section 5.1, we observed that there was a limitation on the training samples, some confusions were found in the test set which were not found in the training set. Also, in Section 8.4 and Chapter 10 (Figures 10.11, 10.12), experimental results showed that using the same number of features, a knowledge base built from a larger training set would have a higher degree of confidence and give a better recognition rate. Therefore an increase in the size of the data base will help in improving the recognition rate.

The quality of the data base can also be improved. In this data base, the resolution is 166 dots/inch. More recent data bases were digitized by better equipment and most of them had the resolution of 300 dots/inch. Naturally the amount of noise is less and little smoothing effort is required when higher resolution is used.

12.3.2 Preprocessing

More sophisticated thinning methods have been discussed in the literature that can provide the following effects:

- to keep the skeleton as the center line of the stroke (in this method the skeleton is sometimes biased more to one side, and straight segments are not well preserved).
- to use information on thickness to preserve loop shapes, and prevent them from being thinned into a single stroke.
- to use knowledge about the numeral to have selective processing, for example removing spurious tails and joining discontinuities in strokes in selected areas

of the image, or having special rules to convert common noise to simpler patterns.

12.3.3 Feature selection

Most methods of recognition and also this method used features that were selected by trial and error, or by subjective decision. The final result was a combination of features that gave the best experimental recognition rate. No attempts were made:

a) to explain why the selected features gave best recognition results,

b) and more importantly, to see whether the selected features are the optimum set in terms of recognition rate and computing cost.

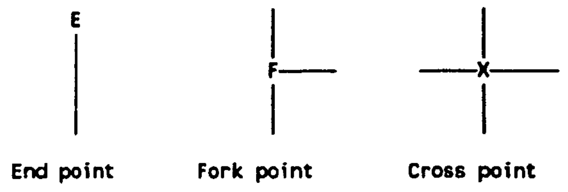
Further works can aim for establishing a framework to measure the 'goodness' of features, to establish the relationship between features (whether the features form a tree structure or a network, what features are at higher level of the tree, what features should be at the leaf node), and to optimize the selection of features and their structure. These works would give better insight to character recognition, produce better recognition rates, or establish an attainable goal on the accuracy and reliability of recognition.

APPENDIX 1

List of Primary and Secondary Features

Notations

- A feature string consists of characters representing the special points in the skeleton that are found in a scan from top to bottom, left to right. There are 3 types of special points, as shown:



In the description column:

- An '*' denotes a primary feature.
- The number with signs in the parenthesis on the right, if present, denotes the purpose of the feature, for example (+9,-5) means the purpose of the feature is to emphasize more on numeral 9 and de-emphasize numeral 5.

Feature No.	Feature String	Description
2	Blank	*
4	EE	*
5	EE	curve at top left
6	EE	small max. width to distinguish 1
13	EE	profile of 3/8 at right
14	EE	large width at bottom
22	EE	inflection point as in a 5
26	EE	small distance between 2 end points
31	EE	numeral at right of 2 end points
38	EE	curve at bottom left
40	EE	a straight stroke between 2 end points
48	EE	first end point very close to top border
49	EE	second end point very close to bottom border
52	EE	curve at bottom left
65	EE	inflection point as in a 2

Feature No.	Feature String	Description
73	EE	stroke to the left of first end point
74	EE	stroke to the right of first end point
83	EE	reject 3 profile
84	EE	reject 3 profile
103	EE	distinguish 7 from stroke widths
131	EE	curve at top right
133	EE	curve at bottom right
136	EE	curve at top left (restricted)
140	EE	distinguish 5 from top and bottom curves
151	EE	large width at bottom
160	EE	2nd end point at top (+0)
171	EE	straight segment at right (+7)
182	EE	a straight stroke at bottom
183	EE	distinguish 2 from top and bottom curves
184	EE	2 end points are far away
206	EE	distinguish 3 from 1,7 using right profile
214	EE	straight segment at bottom right (+9,-5)
229	EE	high jump at right (+9,-5)
230	EE	high jump at right (+9,-5)
233	EE	small bend (+1,-7)
255	EE	right jump at bottom (+6)
257	EE	bend to left (+6,-2)
258	EE	numeral strictly at left of end points
265	EE	distinguish 0-6 from width,bottom end (+0)
270	EE	2nd end point turns into center (+6,-0)
275	EE	small horz. dist. between 2 end points
284	EE	narrow curve at top (+9,-7)
287	EE	profiles of 7
294	EE	eroded tail at bottom (+4)
303	EE	distinguish 2-7 from width
309	EE	reject 3
317	EE	small top (-7)
318	EE	narrow curve at bottom left (-2)
321	EE	thin width at bottom (-6)
350	EE	thick top (+9)
351	EE	thick bottom (+6)
352	EE	profile of 0 at left

Feature No.	Feature String	Description
353	EE	profile of 0 at right
357	EE	distinguish 7 from 1
368	EE	profile of 3 at left
372	EE	reject 0 from profile
384	EE	last end point at right (-3)
385	EE	tail of 2 at bottom right
388	EE	bend at top (+7,-3)
390	EE	small bend of 3 at top (+3,-5)
391	EE	small bend (+1,-7)
406	EE	large curve at bottom right
409	EE	small segment at top (+5,-7)
410	EE	small segment at bottom (+5,-9)
411	EE	no bend in middle (+7,-3)
416	EEEEFF	*
27	EEEF	*
147	EEEF	profile of 6 at left
200	EEEF	low fork point (+2,+6)
156	EEEFFE	*
186	EEEFFE	second end point is low (+2)
240	EEEFFE	left jump at top (+2)
244	EEEFFE	curve at top right (+2)
290	EEEFFE	large horz. dist. between point 2 and cross point
439	EEEFFEFE	*
441	EEEFFF	*
85	EEEFFFEE	*
440	EEEFFFFF	*
325	EEEXE	*
326	EEEXE	2nd end point far from cross point

Feature No.	Feature String	Description
12	EEFE	*
17	EEFE	profile of 4 at left
18	EEFE	2 end points at top
20	EEFE	straight stroke at left side (+4)
24	EEFE	a limb of 3 after fork
33	EEFE	fork point near top
43	EEFE	curve at top left
54	EEFE	small dist. between 2 first end points
55	EEFE	fork point at right of all end points
60	EEFE	fork point at left of all end points
61	EEFE	low fork point
63	EEFE	bottom right > top right
64	EEFE	intersection between end and fork point > 1
66	EEFE	curve at bottom left
75	EEFE	last end point inside numeral
76	EEFE	2nd end point at right of line (-3)
88	EEFE	reject 4 from horz. dist.
100	EEFE	distinguish 3 and 5
102	EEFE	compare 2 top strokes
104	EEFE	straight stroke at right (+1,+4,+7)
132	EEFE	curve at top right
134	EEFE	curve at bottom right
150	EEFE	small horz. dist. between 2 end points
153	EEFE	straight profile at right
168	EEFE	tip of 9 at top
169	EEFE	profile of 5 at right
174	EEFE	profile of 2 at left
189	EEFE	a left jump in a 9
194	EEFE	profile of 3 before fork
195	EEFE	profile of 3 at right
210	EEFE	profile of 6 at left
222	EEFE	thick width at 2nd end point (+6)
227	EEFE	large width between fork and end point (+8,-9)
254	EEFE	limb of 2 before fork
273	EEFE	distinguish 7 from profiles
286	EEFE	right stroke above fork not straight (-4)
306	EEFE	distinguish 3 from 5

Feature No.	Feature String	Description
315	EEFE	distinguish 8 from symmetry
335	EEFE	2 end points on top nearly horizontal
336	EEFE	curve at bottom left
337	EEFE	stroke to the left of first end point
338	EEFE	stroke to the right of first end point
341	EEFE	short straight stroke at left (+7)
359	EEFE	2nd end point close to fork (+1)
383	EEFE	reject 3
392	EEFE	low middle stroke (+2)
414	EEFEEF	*
112	EEFEEFFE	*
19	EEFEFE	*
120	EEFEFE	distinguish 4 from profiles
187	EEFEFE	second end point is low (+2)
207	EEFEFE	curve at top right
220	EEFEFE	distinguish 4
361	EEFEFE	small horz. distance in last 2 end points (+4)
378	EEFEFE	curve at bottom right
394	EEFEFE	large vert. dist. between 4th-6th end points (+2)
436	EEFEFEFE	*
429	EEFEFF	*
81	EEFEFFEE	*
424	EEFEFFFE	*
29	EEFF	*
35	EEFF	large distance between 2 end points (+2,+6)
50	EEFF	profile of 3 at right
59	EEFF	2nd end point at right of 3rd fork (-3)
94	EEFF	horizontal stroke at fork
117	EEFF	high fork point

Feature No.	Feature String	Description
127	EEFF	2 neighboring fork points
142	EEFF	right stroke above fork >> left stroke
178	EEFF	stroke at 2nd end point heads left (-6,-0)
180	EEFF	left profile of an 8
196	EEFF	intersection at middle of fork points = 1
223	EEFF	small width between fork points (+8)
289	EEFF	limb of 2 above fork
295	EEFF	low fork point (+2)
312	EEFF	large dist. between fork
16	EEFFEE	*
90	EEFFEE	straight stroke at bottom (+4)
91	EEFFEE	intersect = 3 at 2 fork points (+4)
111	EEFFEE	profile of 3 at left or right
137	EEFFEE	5th end point row near and left of fork
204	EEFFEE	2 fork points far from each other
236	EEFFEE	low fork point (+4,-7)
238	EEFFEE	left jump at top (+2)
242	EEFFEE	curve at top (+8)
246	EEFFEE	horizontal strokes at bottom (+2)
313	EEFFEE	2 bottom strokes heading left (+8)
340	EEFFEE	2 horz. strokes at right (+6)
349	EEFFEE	slope of middle line large (+8)
23	EEFFEEFE	*
177	EEFFEEFE	profile of 3 at left or right
427	EEFFEF	*
437	EEFFEFEE	*
125	EEFFFE	*
190	EEFFFE	curve at top right
239	EEFFFE	left jump at top (+2)
243	EEFFFE	curve at top left (9)
247	EEFFFE	neighboring forks, last stroke like in a 4
407	EEFFFE	3 fork points at right (+3)

Feature No.	Feature String	Description
408	EEFFFE	straight stroke at right (+4)
438	EEFFFEEE	*
428	EEFFFF	*
432	EEFXE	*
435	EEFXEEE	*
11	EEX	*
218	EEX	small tails (+0)
324	EEX	2 end points are far (+2)
374	EEX	distinguish 6 and 8
375	EEX	a small stroke from fork (+6)
403	EEX	curve at top right
78	EEXEE	*
138	EEXEE	4th end point row near and left of fork
154	EEXEE	intersect = 2 above fork (+4)
158	EEXEE	straight stroke at right (4)
237	EEXEE	left jump at top (+2)
241	EEXEE	curve at top right (+2)
1	EF	*
25	EF	fork point close to end point
58	EF	profile of 3 at right
121	EF	fork point near top
123	EF	profile of 5/8 before fork
145	EF	tail is inside loop
181	EF	fork at right
191	EF	low fork point
193	EF	curve at top right
213	EF	curve at top left
219	EF	a limb of 6 before junction point
225	EF	bend of 3 before fork
231	EF	thick width at top (+5,-8)

Feature No.	Feature String	Description
249	EF	thick stroke between end and fork points (+8)
250	EF	profile of 8 at left
253	EF	bend of 5 and fork at right (+5)
280	EF	limb of 2 before fork
282	EF	profile of 5 at right
299	EF	distinguish 5 from 8
305	EF	changed direction at right of fork (+8)
343	EF	profile of 7
345	EF	fork point far from end (-0)
362	EF	loop comes up higher than fork (+2)
363	EF	long tail at fork
381	EF	fork point can be ignored (+7)
382	EF	distinguish 8 from 6
7	EFEE	*
34	EFEE	fork point near top
36	EFEE	fork at right
46	EFEE	fork near bottom
47	EFEE	curve at top right
53	EFEE	fork point at right of all end points
67	EFEE	curve at bottom left
69	EFEE	curve between fork and last end point
92	EFEE	last end point at left border
93	EFEE	straight segment at bottom (+9)
95	EFEE	3 top points EFE form straight line to left (+5,+7)
97	EFEE	thick top (+9)
98	EFEE	first end point at left of fork
113	EFEE	limb of 2 before fork
135	EFEE	curve at bottom right
141	EFEE	2nd end point at top (+7)
157	EFEE	profile of 3 from fork
201	EFEE	2 first end points are close
232	EFEE	straight right profile (+7,-5)
261	EFEE	stroke of 3,5 after fork
263	EFEE	2nd end point at top (+2)
266	EFEE	middle fork, low end (+8)
281	EFEE	profile of 3 before fork

Feature No.	Feature String	Description
307	EFEE	stroke to the left of first end point
308	EFEE	stroke to the right of first end point
320	EFEE	straight stroke at right (+1,+4,+7)
331	EFEE	limb of 6 before fork
333	EFEE	2 end points are nearly horizontal
342	EFEE	fork point to left (+1,-7)
344	EFEE	2nd end point at right of line (-3)
356	EFEE	reject 3 that has 3rd E at right of fork
389	EFEE	straight profile at left (+1)
393	EFEE	low middle stroke (+2)
396	EFEE	limb of 3 after fork
431	EFEEEF	*
108	EFEEFE	*
110	EFEEFE	2 fork points are far (-4)
208	EFEEFE	curve at top right
323	EFEEFE	2nd end point at right of line (-3)
334	EFEEFE	2 fork points are high
360	EFEEFE	small tail at top (+2)
369	EFEEFE	2 fork points are very high (+7)
377	EFEEFE	jump at bottom right
379	EFEEFE	small vert. dist. between 2 first end points
387	EFEEFE	curve at top left
143	EFEEFF	*
346	EFEEFF	jump at top and at bottom left (+3)
347	EFEEFF	profile of 5 at right
348	EFEEFF	last 2 fork points at middle
420	EFEEFFEE	*
8	EFEF	*
161	EFEF	3rd end point at right of fork (+2)
162	EFEF	stroke to the left of first end point
163	EFEF	stroke to the right of first end point
166	EFEF	profile of 8 at left

Feature No.	Feature String	Description
245	EFEF	profile of 7 at right
268	EFEF	profile of 3 at right
298	EFEF	low fork point
304	EFEF	curve at top right
370	EFEF	a limb of 6 before junction point
397	EFEF	2 neighboring fork points (+2)
148	EFEFEE	*
185	EFEFEE	fork at left of 2 first end points
203	EFEFEE	curve at bottom right
297	EFEFEE	last 2 end points at left of fork (+3)
426	EFEFEF	*
425	EFEFEFEE	*
404	EFEFFE	*
405	EFEFFE	2nd fork at right and not very high (+3)
433	EFEFFF	*
419	EFEFFFEE	*
443	EFEXE	*
9	EFFE	*
56	EFFE	high fork point
96	EFFE	loop at left of fork point
124	EFFE	curve at top right
128	EFFE	horz. intersection between EF = 1 (+2)
139	EFFE	high fork points
144	EFFE	2 neighboring fork points
167	EFFE	profile of 6 at left
172	EFFE	straight stroke of 4 at left
205	EFFE	first end point right of fork
216	EFFE	distinguish a 6 limb
228	EFFE	large width between FE (+8,-9)

Feature No.	Feature String	Description
271	EFFE	limb of 3 after fork
279	EFFE	curve at top left
285	EFFE	large distance between EF
339	EFFE	large width at fork
386	EFFE	curve at bottom right
109	EFFEEE	*
423	EFFEEFEE	*
212	EFFEFE	*
332	EFFEFE	small distance between forks (+2)
450	EFFEFF	*
30	EFFF	*
71	EFFF	profiles of 0
89	EFFF	3 forks at bottom
115	EFFF	profile of 6 at left
116	EFFF	profile of 8 at right
118	EFFF	profile of 8 at left
234	EFFF	profile of 5 at right
256	EFFF	right jump at bottom (+6)
259	EFFF	straight stroke at top left (+6)
310	EFFF	2 neighboring fork points
41	EFFFEE	*
311	EFFFEE	points 5 and 6 are close (+8)
422	EFFFEF	*
417	EFFFEFFE	*
122	EFFFFE	*
87	EFFFFF	*

Feature No.	Feature String	Description
442	EFFFXFFFE	*
452	EFFXE	*
159	EFX	*
149	EXE	*
217	EXE	distinguish a 6 limb
107	EXEEE	*
444	EXEFE	*
418	EXFFE	*
3	FE	*
15	FE	profile of 3/8 at right
44	FE	fork point far from end point
57	FE	fork at right
119	FE	curve at bottom left
129	FE	straight line between F and E
152	FE	end point close to fork (+8)
173	FE	high end point (-9)
188	FE	left jump (+9)
199	FE	low end point
226	FE	large width between FE (+8,-9)
252	FE	straight stroke at right
267	FE	profile of 9 at right
274	FE	fork at top
283	FE	distinguish 8 from symmetry
291	FE	straight stroke of 4 at left
302	FE	profile of 3 from fork
314	FE	fork at left
327	FE	fork point very far from end point
358	FE	small max. width to distinguish 1
395	FE	large width at bottom (+2,-9)
402	FE	curve at bottom right

Feature No.	Feature String	Description
86	FEEE	*
398	FEEE	fork point near top (+0)
399	FEEE	fork point near bottom (+2)
400	FEEE	intersect at 2nd F = 3 (+9)
401	FEEE	2 end points at left and close to fork (+9)
413	FEEFEE	*
449	FEEFFF	*
21	FEFE	*
130	FEFE	low 2nd fork point
164	FEFE	2nd end point close to fork
235	FEFE	fork point at bottom
329	FEFE	profile of 7 at right
434	FEFEEE	*
39	FEFF	*
445	FEX	*
37	FF	*
42	FF	profiles of 0
77	FF	forks not in middle of length (-8)
215	FF	intersect. between FF = 1
224	FF	fork not in middle of width (-8)
322	FF	intersect. between FF = 1
373	FF	fork points are close (+8,-0)
376	FF	loop comes up higher than fork
80	FFEE	*
82	FFEE	2nd fork far from first fork (+2)
198	FFEE	profile of 3 at right
278	FFEE	fork point not in middle (-8)
371	FFEE	large vert. distance between end points

Feature No.	Feature String	Description
447	FFEEFF	*
354	FFEF	*
355	FFEF	large horz. dist. between points 1 and 4 (+8)
101	FFEFFE	*
412	FFEFFF	*
10	FFFE	*
45	FFFE	last 2 fork points are high (+9)
126	FFFE	2nd fork far from first fork (+2)
430	FFFEFE	*
79	FFFF	*
451	FFFFEE	*
421	FFFFFE	*
446	FFFFFF	*
448	FFXFF	*
415	FXE	*
70	X	*
99	XEE	*
202	XEE	2nd end point at left of tork

Appendix 2

PROGRAM EXECUTION TRACE OF INFERENCE METHOD ON ONE SAMPLE

```
{ The knowledge base of hypothesis sets is loaded to program memory, this
  knowledge base has 407 rules and is build from 8500 training samples}

LOADED   407 8500

{ The unknown sample is read and measured }

              1         2         3         4         5
    012345678901234567890123456789012345678901234567890
  0
  1                                --------------
  2                                --------------
  3                               ----------------
  4                               ---E*****--------
  5                               ---------**------
  6                               -----------*-----
  7                               -----   ----*----
  8                                       ----*---
  9                                     ----**---
 10                                   -----*----
 11                                 ------*----
 12                                ------*----
 13                          -----------*---
 14                       ----------***F----
 15                      ----------*----*----
 16                     -----******------*----
 17                     ---**-------------*---
 18                    ---E---------  - ---*--
 19                     --------         --*--
 20                      ---             --*--
 21                                     ---*--
 22                                    ---*---
 23                                   ---*---
 24                                  ---*---
 25                                 ---*---
 26                                ----*---
 27                              ----**---
 28                            -----*----
 29    -                     ------*----
 30   ----               --------**----
 31   --E-           ---------***----
 32   --*-     - -----------**-----
 33  ---*---------------****------
 34   ---*-------*******------
 35   ----*******-----------
 36    -----------------
 37             ---
```

```
NUMERAL NO.4003    HEIGHT 37    WIDTH 46    DIAGONAL 59

AVERAGE THICKNESS   6.91

{ There are 2 jumps on the left profile, occurring at rows 21 and 29, the
  magnitudes of the jumps are respectively 15 pixels to the right and 24
  to the left }

  Number of jumps: Left  2  Right  0

             LEFT                          RIGHT
       Position  Jump Distance  Position   Jump Distance
    1     21         15             0          0
    2     29        -24             0          0

{ the skeleton 's position is from rows 4 to 35, columns 4 to 42 }

    Top   Bottom   Left   Right
     4      35       4     42

{ the strings representing the profiles are displayed, LP, RP are left and right
  profiles of the original image, LSP, RSP are left and right profiles of the
  skeleton }

  LP 23 RP 14 LSP 14 RSP 14

{ the primary feature string is displayed }

  EFEE

=====================================     START MATCHING  =====================================

{ the features are matched with the knowledge base }

{ By the primary string, the sample has feature 7 which is the primary feature
  EFEE. For this inference, the hypothesis set of the primary feature is the
  universal set }

  RULE   7   :  0:  3    1:  4    2:178    3:466    4:  4    5: 42    6:  5    7: 61    8:  1    9:  3
  HYPOT. SET :  0:  3    1:  4    2:178    3:466    4:  4    5: 42    6:  5    7: 61    8:  1    9:  3

-----------------------------------------------------------------------------------------------

{ MATCHING OF FEATURE 34 }
{ the hypothesis set of feature 34 is listed }
  RULE 34    :  0:  3    1:  4    2:  1    3:290    4:  1    5: 36    6:  1    7: 60    9:  2

{ Feature 34 is found to be present }
  RULE 34    :  FORK POINT NEAR TOP  * TRUE *
```

```
{ Hence the hypothesis set is the hypothesis set of feature 34 intersected with
  the previous hypothesis set. Because the previous set is the universal set,
  the new hypothesis set is the same as the hypothesis set of feature 34 }

  HYPOT. SET :  O:  3    1:  4    2:  1    3:290    4:  1    5: 36    6:  1    7: 60    9:  2

--------------------------------------------------------------------------------------------------

{ MATCHING OF FEATURE 36 }
  RULE  36   :  1:  2    2:  1    3: 82    7: 43    9:  1

{ The sample does not have this feature }
  RULE 36    :  FORK POINT AT RIGHT (2)

{ It then must also belong to another hypothesis set which is the universal set
  subtracted by the hypothesis set of feature 36 }

  SET    7   :  O:  3    1:  4    2:178    3:466    4:  4    5: 42    6:  5    7: 61    8:  1    9:  3
  subtracted by
  SET   36   :           1:  2    2:  1    3: 82                               7: 43             9:  1
  gives the result
  RESULT     :  O:  3    1:  2    2:177    3:384    4:  4    5: 42    6:  5    7: 18    8:  1    9:  2

{ The last hypothesis set is from Rule 34. The 2 sets are intersected to give
  a new hypothesis set }

  RESULT     :  O:  3    1:  2    2:177    3:384    4:  4    5: 42    6:  5    7: 18    8:  1    9:  2
  intersected with
  HYPOT. SET :  O:  3    1:  4    2:  1    3:290    4:  1    5: 36    6:  1    7: 60    9:  2
  gives the result :

  HYPOT. SET :  O:  3    1:  2    2:  1    3:290    4:  1    5: 36    6:  1    7: 18    9:  2

--------------------------------------------------------------------------------------------------

{ The inference process continues
  - If a feature is found: the hypothesis set of the feature is intersected
    with the previous hypothesis set.
  - If a feature is not found: the hypothesis set of the feature is subtracted
    from the universal set, then the result is intersected with the previous
    hypothesis set }

  RULE  46   :  2:144    3:  3    4:  1    6:  2    7:  1
  RULE  46   :  FORK POINT NEAR BOTTOM
  HYPOT. SET :  O:  3    1:  2    2:  1    3:290    4:  1    5: 36    6:  1    7: 18    9:  2

  RULE  47   :  2:154    3:397    7:  2
  RULE  47   :  CURVE AT TOP RIGHT     * TRUE *
  HYPOT. SET :  2:  1    3:290    7:  2
```

```
RULE  53    :  2:  2     3:375
RULE  53    :  FORK POINT ON THE RIGHT OF ALL END POINTS    * TRUE *
HYPOT. SET  :  2:  1     3:290

RULE  67    :  O:  3     2: 39     3:  1     6:  3
RULE  67    :  CURVE AT BOTTOM LEFT
HYPOT. SET  :  2:  1     3:290

RULE  69    :  3:  1     5: 22     6:  2     7: 16     8:  1     9:  1
RULE  69    :  CURVE AT TOP LEFT
HYPOT. SET  :  2:  1     3:290

RULE  92    :  1:  3     2: 83     3:368     4:  1     5: 37     6:  2     7: 42     9:  2
RULE  92    :  LAST END POINT ON LEFT BORDER  * TRUE *
HYPOT. SET  :  2:  1     3:290

RULE  93    :  5:  1     7: 55     9:  1
RULE  93    :  STRAIGHT SEGMENT AT BOTTOM (7,9)
HYPOT. SET  :  2:  1     3:290

RULE  95    :  1:  2     3:  1     5: 18     6:  1     7: 50     9:  1
RULE  95    :  EFE FORMS STRAIGHT LINE TO LEFT (+5,+7)
HYPOT. SET  :  2:  1     3:290

RULE  97    :  7:  1     9:  1
RULE  97    :  THICK TOP (9)
HYPOT. SET  :  2:  1     3:290

RULE  98    :  O:  2     2:145     3:437     4:  1     7:  5     9:  1
RULE  98    :  FIRST END POINT LEFT OF FORK  * TRUE *
HYPOT. SET  :  2:  1     3:290

RULE 113    :  2:163     3:174     4:  2     5:  4
RULE 113    :  LIMB OF 2 BEFORE FORK
HYPOT. SET  :  2:  1     3:290

RULE 135    :  2: 47     3:413     5: 35     6:  1
RULE 135    :  CURVE AT BOTTOM RIGHT   * TRUE *
HYPOT. SET  :  2:  1     3:290

RULE 141    :  3:  1     5:  1     7: 31
RULE 141    :  2ND E ON TOP (7)
HYPOT. SET  :  2:  1     3:290

RULE 157    :  O:  2     3:  9
RULE 157    :  PROFILE OF 3 FROM FORK
HYPOT. SET  :  2:  1     3:290

RULE 201    :  9:  1
RULE 201    :  2 FIRST END POINTS CLOSE (+9)
HYPOT. SET  :  2:  1     3:290
```

```
RULE 232    :  1:  1    4:  1    7: 49    9:  1
RULE 232    :  STRAIGHT RIGHT PROFILE (+7,-5)
HYPOT. SET  :  2:  1    3:290

RULE 261    :  3:236    5: 28    6:  1
RULE 261    :  STROKE OF 3,5 AFTER FORK
HYPOT. SET  :  2:  1    3:230

RULE 263    :  0:  3    1:  1    3:189    4:  1    5: 11    7: 59    9:  2
RULE 263    :  2ND END POINTS AT TOP  * TRUE *
HYPOT. SET  :  3:189
-----------------------------------------------------------------------------------

{ at this point, the hypothesis set has only 1 hypothesis, however the
  inference still continues to make the recognition reliable. In many cases,
  another hypothesis which is in contradiction with the established is
  encountered. The intersection of the 2 hypothesis is a null set. In those
  cases, the character is rejected as unrecognizable }

RULE 266    :  2: 14    8:  1
RULE 266    :  MIDDLE FORK, LOW END (+8)
HYPOT. SET  :  3:189

RULE 281    :  3:  3
RULE 281    :  PROFILE OF 3 BEFORE FORK
HYPOT. SET  :  3:189

RULE 307    :  0:  1    1:  2    3:  3    4:  3    5: 36    6:  5    7: 54    9:  1
RULE  73    :  STROKE TO THE LEFT OF END POINT (+5)
HYPOT. SET  :  3:189

RULE 308    :  2: 41    3: 85    7:  1
RULE 308    :  STROKE TO THE RIGHT OF END POINT (+3)
HYPOT. SET  :  3:189

RULE 320    :  1:  2    4:  3    7: 51    9:  1
RULE 320    :  STRAIGHT RIGHT STROKE IN 1,4,7
HYPOT. SET  :  3:189

RULE 331    :  2: 12    4:  1    6:  3    7:  1
RULE 331    :  LIMB OF 6 BEFORE FORK
HYPOT. SET  :  3:189

RULE 333    :  4:  1    5: 31    6:  1    7: 50    9:  1
RULE 333    :  EE ON TOP NEARLY HORIZONTAL (+5)
HYPOT. SET  :  3:189

RULE 342    :  1:  2    4:  1
RULE 342    :  FORK POINT TO LEFT (+1,-7)
HYPOT. SET  :  3:189
```

```
 RULE 344    :  2: 41    7:  2
 { this feature is not matched because a precondition exists }
 HYPOT. SET ·  3:189

 RULE 356    :  2: 22    7:  2
 RULE 356    :  REJECT 3 THAT HAS 3RD E AT RIGHT OF FORK
 HYPOT. SET :  3:189

 RULE 389    :  1:  2
 RULE 389    :  STRAIGHT LEFT PROFILE (+1)
 HYPOT. SET :  3:189

 RULE 393    :  2:  2
 RULE 392    :  LOW MIDDLE STROKE (+2)
 HYPOT. SET :  3:189

 RULE 396    :  3:436    5: 39    6:  1    7:  5    8:  1
 RULE 271    :  LIMB OF 3 AFTER FORK    * TRUE *
 HYPOT. SET :  3:189
------------------------------------------------------------------------------------------

 FINAL HYPOTHESIS SET :  3:189

=== 4003  RECOGNITION :    3  CONFIDENCE  100 %   NUMERAL  3  ===
```

REFERENCES

[AHME86] P. Ahmed,"Computer Recognition of Totally Unconstrained Handwritten Zip Codes," Ph. D Thesis in Computer Science, Concordia University, Montreal, Canada, 1986.

[AS82] P. Ahmed and C. Y. Suen, "Segmentation of Unconstrained Handwritten Numeric Postal Zip Codes," Proc. Int. Conf. Pattern Recognition and Artificial Intelligence, 1987, pp. 545-547.

[AS87] P. Ahmed and C. Y. Suen, "Computer Recognition of Totally Unconstrained Handwritten Zip Codes," Int. Journ. of Pattern Recognition and Artificial Intelligence, Vol. 1, 1987, pp. 1-15.

[BADR85] A. M. K. Badreldin, "Structural Recognition of Handwritten Numeral Strings," Ph. D. Thesis, University of Windsor, Ontario, Canada, 1985.

[BAIR86] H. S. Baird, "Feature Identification for Hybrid Structural/Statistical Pattern Classification," Proceedings IEEE Conf. on Computer Vision and Pattern Recognition, Florida, 1986, pp. 150-155.

[BEUN73] M. Beun, "A Flexible Method for Automatic Reading of Handwritten Numerals," Philips Technical Reviews, Vol 33, 1973, pp. 89-101, 130-137.

[BFW88] R. M. Brown, T. H. Fay and C. L. Walker, "Handprinted Symbol Recognition System," Pattern Recognition, Vol. 21, No. 2, 1988, pp. 91-118.

[BK88] G. Baptista and K. M. Kulkarni, "A High Accuracy Algorithm for Recognition of Handwritten Numerals," Pattern Recognition, Vol. 21, No. 4, 1988, pp. 287-291.

[CJ83] R. G. Casey and C. R. Jih, "A Processor-Based OCR System," IBM J. Res. Devel. , Vol. 27, No. 4, July 1983, pp. 386-399.

[DHTW80] B. Duerr, W. Haettich, H. Trof and G. Winkler, "A Combination of Statistical and Syntactical Pattern Recognition Applied to Classification of Unconstrained Handwritten Numerals," Pattern Recognition, Vol. 12, 1980, pp. 189-199.

[FU80] K. S. Fu, "Digital Pattern Recognition," Ed. K. S. Fu, 2nd Ed. , Springer-Verlag 1984.

[FH83] J. F. Flemming and R. F. Hemmings, "A Method of Recognition for Handwritten Block Capitals," Pattern Recognition Letters 1, 1983, pp. 457-464.

[GALL85] S. I. Gallant, "Automatic Generation of Expert Systems from Examples," Proc. of The 2nd Conf. on AI Applications, 1985, pp. 313-319.

[GLUC67] H. A. Glucksman, "Classification of Mixed-font Alphabetics by Characteristic Loci," Digest of 1st. Ann. IEEE Comp. Conf. , Sept 1967, pp. 138-141.

[GUDE76] A.Gudesen, "Quantitative Analysis of Preprocessing Techniques for the Recognition of Handprinted Characters," Pattern Recognition, Vol. 8, 1976, pp. 219-227.

[GV88] A. Gardin du Boisdulier and P. Vanhulle, "Topological Modelization of Handwritten Digits Based on a Geometrical Primitive Decomposition," Proc. US Postal Service Advanced Technology Conf. 3, 1988, pp. 797-812.

[HC86] J. S. Huang and K. Chuang, "Heuristic Approach to Handwritten Numeral Recognition," Pattern Recognition, Vol. 19, No. 1, 1986, pp. 15-19.

[HOSK72] K. H. Hosking, "A Contour Method for The Recognition of Handprinted Characters," Machine Perception of Patterns and Pictures, The Institute of Physics, London, 1972, pp. 19-27.

[HULL88] J. J. Hull, S. N. Srihari, E. Cohen, C. L. Kuan, P. Cullen and P. Palumbo, "A Blackboard-Based Approach to Handwritten Zip Code Recognition," Proc. US Postal Service Advanced Technology Conf. 3, 1988, pp. 1018-1032.

[HUNT72] D. J. Hunt, "A Feature Extraction Method for The Recognition of Handprinted Characters," Machine Perception of Patterns and Pictures, The Institute of Physics, London, 1972, pp. 28-33.

[KC88] F. A. Kamangar and M. J. Cykana, "Recognition of Handwritten Numerals using a Multilayer Neural Network," Proc. US Postal Service Advanced Technology Conf. 3, 1988, pp. 768-781.

[KS88] C. L. Kuan and S. N. Srihari, "A Stroke-Based Approach to Handwritten Numeral Recognition," Proc. US Postal Service Advanced Technology Conf. 3, 1988, pp. 1033-1041.

[KPS79] C. C. Kwan, L. Pang and C. Y. Suen, "A Comparative Study of Some Recognition Algorithms in Character Recognition," Proc. Int. Conf. Cybernet. Soc. 1979, pp. 530-535.

[LAM86] L. Lam, "Structural Classification and Relaxation Matching of Totally Unconstrained Handwritten Numerals," Master Thesis in Computer Science, Concordia University, Montreal, Canada, 1986.

[LS88] L. Lam and C. Y. Suen, "Structural Classification and Relaxation Matching of Totally Unconstrained Handwritten Numerals," Pattern Recognition, Vol. 21, No. 1, 1988, pp. 19-31.

[MAI86a] T. Mai, "Term Project in COMP 673," Concordia University, April 1986.

[MAI86b] T. Mai, "Term Project in COMP 772," Concordia University, December 1986.

[MANT87] J. Mantas, "Methodologies in Pattern Recognition and Image Analysis - A Brief Survey," Pattern Recognition, Vol. 20, 1987, pp. 1-6.

[MCLA68] J. A. McLaughlin and J. Raviv, "Nth-Order Autocorrelations in Pattern Recognition," Inform. Control, Vol. 12, 1968, pp. 121-142.

[MG88] B. T. Mitchell and A. M. Gillies, "Advanced Research in Recognizing Handwritten ZIP Codes," Proc. US Postal Service Advanced Technology Conf. 3, 1988, pp. 813-827.

[MONG82] Y. Mong, "An Experimental Study of the Syntactic Analysis and Recognition of Hand-written Numerals," Master Thesis, Concordia University, 1982.

[MYY84] S. Mori, K. Yamamoto and M. Yasuda, "Research on Machine Recognition of Handprinted Characters," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No. 4, July 1984, pp. 386-405.

[NS84] N. J. Nacache and R. Shingal, "An Investigation into the Skeletonization Approach of Hilditch," Pattern Recognition, Vol. 17, 1984, pp. 279-284.

[PWY87] Pan Bao-Chang, Wu Si-Chang and Yan Guang-Yi, "A Method of Recognizing Handprinted Characters," Proc. of the 3rd Int. Symposium on Handwriting and Computer Applications, 1987, pp. 75-77.

[PLHS88] T. F. Pawlicki, D. Lee, J. J. Hull and S. N. Srihari, "Neural Network Models and their Application to Handwritten Digit Recognition," Proc. US Postal Service Advanced Technology Conf. 3, 1988, pp. 751-767.

[SAL88] C. Y. Suen, P. Ahmed and L. Lam, "Recognition of Totally Unconstrained Handwritten Zip Codes," Proc. US Postal Service Advanced Technology Conf. 3, 1988, pp. 885-866.

[SB86] M. Shridhar and A. Badreldin, "Recognition of Isolated and Simply Connected Handwritten Numerals," Pattern Recognition, Vol. 19, No. 1,1986, pp. 1-12.

[SCHU82] J. Schurmann, "Reading Machines," Proc. 6th Int. Conf. on Pattern Recognition, 1982, pp. 1031-1044.

[SUEN82a] Ching Y. Suen, "Distinctive Features in the Automatic Recognition of Handprinted Characters," Signal Processing 4, 1982, pp. 193-207.

[SUEN82b] Ching Y. Suen, "The Role of Multi-Directional Loci and Clustering in Reliable Recognition of Characters," Proc. 6th Int. Conf. on Pattern Recognition, 1982, pp. 1020-1022.

[SUEN86] C. Y. Suen, "Character Recognition by Computer and Applications," Handbook of Pattern Recognition and Image Processing. Academic Press 1986.

[ULLM82] J. R. Ullmann, "Advances in Character Recognition," Ed. K. S. Fu, Application of Pattern Recognition, CRC Press, 1982, pp. 197-236.

[VANB76] J. F. van Bilzem, A. A. Spanjersberg and J. van Staveren, "Method and Device for the Recognition of Characters, Preferably of Figures," US Patent 3999161, Dec. 1976.

[WATA88] S. Watanabe, Y. Nakamura, M. Suda, K. Oh-I, K. Yura, H. Sasaki, N. Takagi and A. Tanaka, "Kanji Character Recognition and its Application to Address Reading," Proc. US Postal Service Advanced Technology Conf. 3, 1988, pp. 826-841.

[WS84] Q. R. Wang and C. Y. Suen, "Analysis and Design of a Decision Tree based on Entropy Reduction and its Application to Large Character Set Recognition," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-6, No. 4, July 1984, pp. 406-417.