MACHINE RECOGNITION OF INDEPENDENT AND CONTEXTUALLY CONSTRAINED CONTOUR-TRACED HANDPRINTED CHARACTERS

by

GODFRIED T. TOUSSAINT

B.Sc., University of Tulsa, 1968

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in the Department of
Electrical Engineering

We accept this thesis as conforming to the

required standard

Research Supervisor.....

Members of Committee.....

.....

Acting Head of Department.....

Members of the Department

of Electrical Engineering

THE UNIVERSITY OF BRITISH COLUMBIA

December, 1969

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree tha permission for extensive copying of this thesis for scholarly purposes may be granted by the Head of my Department or by his representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical Engineering

The University of British Columbia
Vancouver 8, Canada

Date January 7, 1970

ABSTRACT

A contour-tracing technique originally divised by Clemens and Mason was modified and used with several different classifiers of varying complexity to recognize upper case handprinted alphabetic characters. An analysis and comparison of the various classifiers, with the modifications introduced to handle variable length feature vectors, is presented.

On independent characters, one easily realized suboptimum parametric classifier yielded recognition accuracies which compare favourably with other published results. Additional simple tests on commonly confused characters improved results significantly as did use of contextual constraints. In addition, the above classifier uses much less storage capacity than a non-parametric optimum Bayes classifier and performs significantly better than the optimum classifier when training and testing data are limited.

The optimum decision on a string of m contextually constrained characters, each having a variable-length feature vector, is derived. A computationally efficient algorithm, based on this equation, was developed and tested with monogram, bigram and trigram contextual constraints of English text. A marked improvement in recognition accuracy was noted over the case when contextual constraints were not used, and a trade-off was observed not only between the order of contextual information used and the number of measurements taken, but also between the order of context and the value of a parameter ds which indicates the complexity of the classification algorithm.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

ACKNOWLEDGEMENT

Grateful acknowledgement is given to the National Research Council of Canada, The Defence Research Board of Canada, and the British Columbia Telephone Company for the financial support received under Grants NRC A-3308, and DRB 2801-30, and for the Graduate Scholarship, respectively.

I would like to thank Dr. Robert W. Donaldson, the supervisor of this project, for his generous counsel, and constant availability and encouragement.

I would like to thank Dr. Michael P. Beddoes for reading the manuscript and for his helpful suggestions. I am grateful to the graduate students of the Electrical Engineering Department of The University of British Columbia for providing the handprinted character samples and for participating in the human recognition tests. I am also grateful to Messrs. J. Cavers, M. Ito and S. Hussain for the many interesting discussions over N+1 cups of coffee.

I would also like to thank my wife, Ninozka, for her encouragement and for operating the projector during the tedious preparation of the data, Miss Beverly Harasymchuk for typing the manuscript, and Messrs. John Cossalter, Tom Fletcher, and Toomas Vilmansen, for proofreading the manuscript.

I. INTRODUCTION

1.1 Purpose of Research

The purpose of this research was to: (1) assess the suitability of a simple contour-analysis feature-extraction scheme for recognizing handprinted upper case characters when the feature vector is operated on by classifiers of varying degrees of complexity; (2) improve the results of the classifiers above by using contextual constraints and additional simple class-dependent tests to differentiate commonly confused characters; and (3) observe trade-offs, if any, between the number of measurements, complexity of the basic classifier and order of contextual information used when the available storage is limited.

1.2 Review of Previous Research

The great spatial variability of handprinted characters, even among samples from the same person, has lead many researchers to explore unique methods of feature extraction and classification. In many of these cases the preprocessing is of considerable complexity and the dimensionality of the feature vectors is large, typically greater than 100, making classification difficult [1-3]. Some of the researchers such as Kamentsky [4], Bakis et al. [5], and Greanias et al. [6] have considered only numeric character sets, and others such as Roberts [7], Highleyman [8] and Grimsdale et al. [9] used the same data set for training and testing which is now known to yield an overoptimistic prediction of performance.

In 1965 Clemens [10, 11] devised a relatively simple contour tracing algorithm to recognize machine-printed characters. The relative ease of implementation of the scheme and the low dimensionality of the feature vectors, typically about 20, attracted researchers of handprinted character recognition. In 1966 Chodrow et al. [12] reported poor results in applying Clemens' technique to alphabetic handprinted characters. In 1968 Munson [13] obtained equally poor

results using a slight modification of Clemens' technique. Since then, better results have been reported by Johnson et al. [14], Munson [13, 15] and Knoll [16] using more complicated techniques, but there seems little hope of arriving at very satisfactory results without the use of contextual information.

In the past, the use of contextual constraints to improve character recognition has been scant and limited generally to the dictionary look-up method and the Markov approach. In 1959 Bledsoe and Browning [1] used the character confidences of a word together with a vocabulary of English words of the length in question to make a decision on the word. Cornew [17] applied an extensive dictionary look-up method to spelling correction and Alter [18] used a sequential decoding method for determining which of many possible symbol sequences is in some sense most likely to have been intended given the sequence actually received. The disadvantages of these methods are that a dictionary of practical size takes up too much storage, and not enough information is used from the measurements. In fact these methods often operate on the output of a classifier which contains the decision, rather than the category likelihoods, for the input character. The Markov approach is based on the assumption that the true identity of a character is dependent on the true identities of neighbouring characters. Carlson [19] used the Markov approach to replace missing characters from names on the basis of the most probable trigrams. Edwards and Chambers [20] used conditional bigrams in a machine-printed character recognition scheme, which uses the previous decision and a present choice based on the two categories having highest likelihoods, actually resulting in a decrease in performance for high initial recognition accuracies.

In 1966 Abend [21, 22] theoretically solved the problem of optimum use of context using compound decision theory. Practically, however, this solution is virtually unrealizable and simplifying assumptions have to be made.

One way to simplify the problem is to assume Markov dependence among the characters to be recognized and then use sequential compound decision theory to make a decision on one character at a time [23]. Another way to simplify the problem is to have as the classifier output only a fixed number of choice categories having the highest confidence. This simplification was made by Duda and Hart [24] in a handprinted character recognition scheme using both syntax and semantics. However, the study was made only on the FORTRAN language and the confidences used were not equal to the category likelihoods. Similar studies need to be made on the much more difficult problem of handling natural language such as English. In addition, similar decision algorithms need to be derived for handling variable-length feature vectors which result from some feature extraction schemes such as the one used in this thesis.

1.3 Scope of the Thesis

In this thesis, the complete pattern recognition system shown in Fig. 1 is considered. The input to the system consists of upper case handprinted alphabetic characters, binary quantized on 50 x 50 arrays. The transducer, in this case, consists of an enlarger projector and keypunch operators. Two types of feature extraction are performed on the data, for comparison, by a method of contour tracing based on Clemens' technique. Various algorithms are used in the classifier stage either by themselves or making use of contextual information or tentative decision making followed by additional feature extraction before proceeding to a final decision.

The theoretical aspects of the techniques used to solve the pattern recognition problem are treated in Chapter II. Some aspects regarding the square contour trace not treated before in the literature, including the equations for its implementation, are discussed in section 2.3. Section 2.4 briefly reviews some of the simple classification techniques used in the past and

describes the necessary changes needed to handle variable length feature vectors. Subsection 2.4.5 describes a non-Euclidean distance classifier for which, to the author's knowledge, no character recognition experiments have been previously reported. Subsection 2.4.6 describes some of the important interrelationships, not observed in the literature, between the various classifiers, and presents an interesting theorem and proof concerning the equivalence between a linear decision boundary and the nonlinear decision boundary of the classifier of subsection 2.4.5. Subsection 2.4.8 describes a parallel-sequential mode of classification for which the scans and decision trees of section 4.2 were designed. It was necessary to obtain the equation, not available in the literature, for the optimum decision on a string of m characters having variable length feature vectors. This equation is derived in subsection 2.4.9 and an apparently very successful modification is made for decreasing the amount of computation.

The experiments are described in Chapter III. In the past, two methods have been used with relatively small data sets; the first does not use the data efficiently and the second needs a very large amount of computation. A new experimental procedure is described which is a compromise between the two methods above. The results are given in Chapter IV and the discussion and conclusions are treated in Chapter V.

II. THE PATTERN-RECOGNITION PROBLEM

2.1 Introduction

The general pattern recognition problem is expressed in the block diagram in Fig. 1. The real world data set is composed of a possibly infinite number of patterns which are to be classified into a finite number of categories in decision space. The transducer converts a pattern from the real world into a pattern in image space by digitization and quantization which contribute to the high dimensionality of the image space. The feature extractor maps the image space into the, usually much lower dimensional, measurement space. Feature extraction is effected by designing the mapping in such a way that there exist strong intra-category similarities and strong inter-category dissimilarities among feature vectors. In some cases the transducer and feature extractor are difficult to separate. The classifier acts on measurement space and makes either a tentative or final decision either solely on the basis of the feature vectors or with the help of contextual information.

Let the features from pattern class C_i be described by a vector $\vec{X} = (x_1, x_2, \ldots, x_d)$. If $P(C_i)$ and $P(\vec{X}|C_i)$ denote, respectively, the probability of C_i and the probability of $\vec{X}$ conditioned on C_i, then the probability of misclassification is minimized by choosing i to maximize any monotone increasing function of

$$R_i = P(\vec{X}|C_i)\ P(C_i) = P(x_1, x_2, .., x_d|C_i)\ P(C_i)\ . \qquad (1)$$

In most handprinted character recognition schemes the dimensionality d of $\vec{X}$ is large; typically $d > 100$. Even if the components of $\vec{X}$ are binary, learning the statistics of 2^d different probabilities $P(\vec{X}|C_i)$ for each i requires many training samples and much storage capacity. When the d components x_k of $\vec{X}$ are statistically independent, however,

$$P(\vec{X}|C_i)\ P(C_i) = \prod_{k=1}^{d} P(x_k|C_i)\ P(C_i) \qquad (2)$$

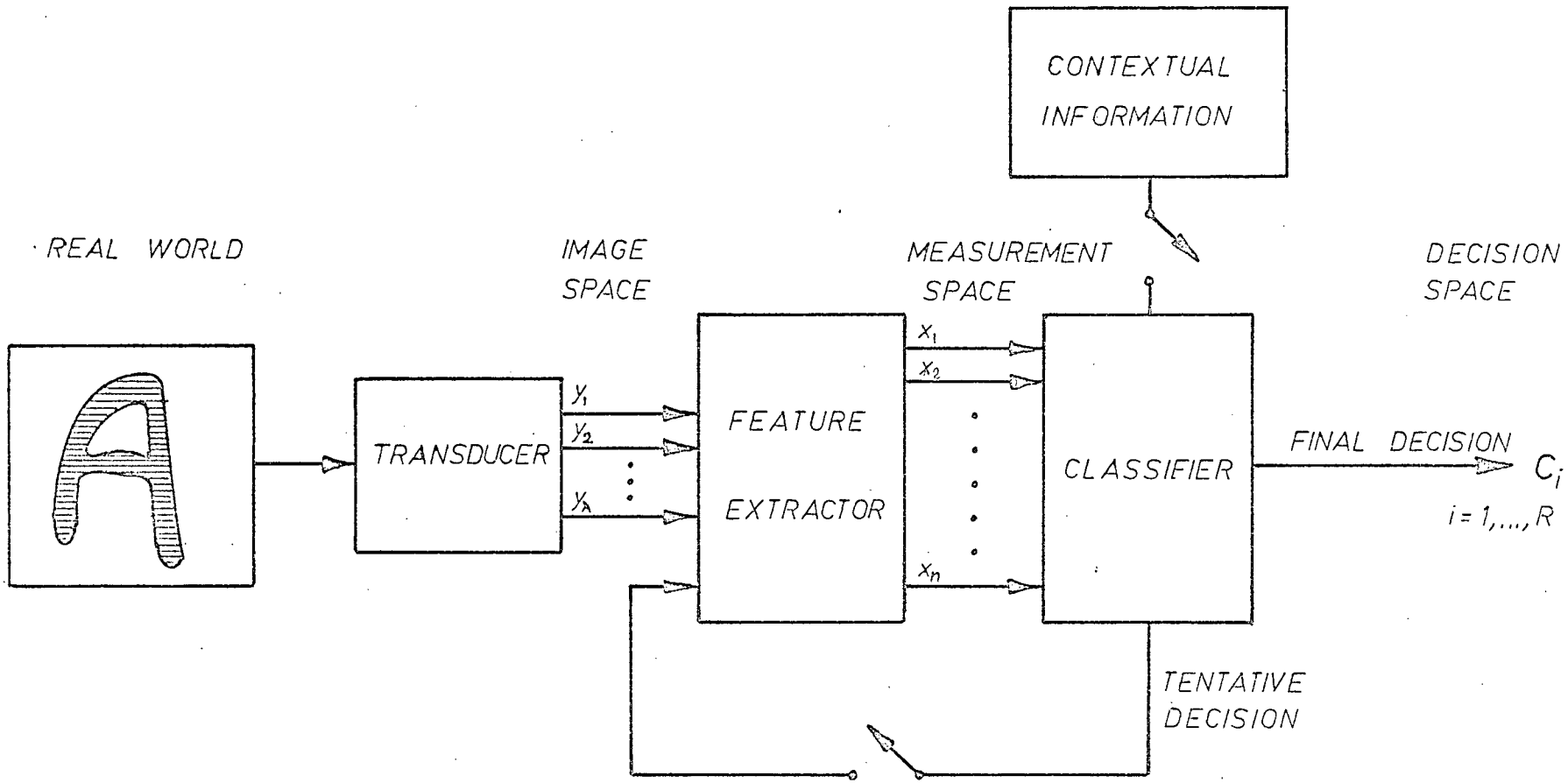


Fig. 1 The general pattern-recognition problem.

in which case it is necessary to learn and store only the d terms $P(x_k|C_i)$ for each i. Two possibilities now arise. (A) If the components x_k are not strongly interdependent, and if there are not enough samples to estimate $P(\vec{X}|C_i)$ sufficiently well when the true distribution is not known, it may be best to assume (2), or to use some other suboptimum classifier which can be trained on fewer samples than are needed to estimate $P(\vec{X}|C_i)$. (B) Even when $P(\vec{X}|C_i)$ is known a suboptimum classifier which requires less storage capacity than the optimum classifier may be desirable. Unused storage can then be occupied by contextual information and more sophisticated classification algorithms, both of which can effect considerable improvement in recognition accuracy [13, 18, 23, 24]. Thus, feature extraction schemes which retain those features essential for recognition while keeping d small are attractive, because not only do they simplify the classifier in terms of less storage and computation, but the relative independence of the x_k's encourages the use of (2), further simplifying the problem.

2.2 Transducers

Transducers can be roughly divided into two groups. In one kind only digitization and quantization are performed and the difference between the pattern in the real world and in image space is minimized. In the other kind actual feature extraction is done during transducing, thus mapping the real world directly into measurement space. Either scheme can be used with contour tracing.

2.3 Feature Extraction

2.3.1 Introduction

There exists today no general theory that will yield an optimal set of features for a given pattern recognition problem. Hence there is some justification in allowing designers to use features which they think might

extract the significant information from patterns. In the past some success has been achieved through intuitive application of ideas from biological prototypes but the field remains an art rather than a science [25].

It is well known that the contours of a pattern contain most of the information needed to recognize it [5, 6, 10-12, 25-32]. The feature extraction algorithms used in this thesis are based on two observations. First, reasonably legible Roman characters are recognizable solely on the basis of their external contour. Second, confusion between characters is not random, but highly structured. For example, in the experiments described below confusions arose mostly between the characters A and R, K and X, O and Q, and B and D. Accordingly, a character's outside contour was used to generate a binary vector $\vec{X}$ which was then classified either as a single character, or as one of a group of characters not easily distinguished solely on the basis of $\vec{X}$. In this latter case, simple class-dependent tests were designed to effect final recognition of the unknown character. The transformation of character contours into binary vectors is a modification of the one used by Clemens and Mason [10, 11, 25-27] and by Troxel [28, 29] for recognizing machine-printed characters. In addition to being easy to implement, the transformation yields a feature vector of small dimensionality.

2.3.2 Contour Tracing

All contour tracing algorithms are similar in that they effect, in some manner, a stepwise trace of the black-white boundary of a pattern. There are, however, several ways of implementing the stepwise trace each serving a slightly different purpose and requiring different local properties of the patterns to be traced [6, 10, 26, 29, 31-33]. Greanias et al. [6] was one of the first to use circular tracing. In his Ph.D. thesis Clemens [10] introduced hexagonal tracing, a simple operation requiring little logic.

Later Mason and Clemens [26] used the square trace which is used in this thesis. The algorithm consists of making a left turn upon entering a black square and a right turn upon entering a white square as illustrated in Fig. 2. Let R(x,y) be the image matrix composed of elements which can take on the values 'zero' or 'one', where the set of 'ones' is the quantized representation of some pattern $P(p_1,p_2,\ldots,p_n)$.

Definition

Two elements of R are adjacent if they differ by 1 in one of their coordinates.

Definition

Two elements of R are diagonally adjacent if they differ by 1 in both of their coordinates.

Definition

A pattern P is connected if, and only if, given any two elements p_i and p_j, there exists a sequence of elements $(p_1,p_2,\ldots,p_\ell)$ where $p_1=p_i$ and $p_\ell = p_j$ such that p_m and p_{m+1} are adjacent for $1 \leq m \leq \ell-1$.

Equations describing the square trace were derived in order to compare them with those for hexagonal tracing in [10]. Let (x_k, y_k) and (x_{k-1}, y_{k-1}) be the coordinates of the present and past location of the scanning spot. The next location on R is given by the following 3-state equations:

$$x_{k+1} = x_k + \{[y_k - y_{k-1}][1-2R(x_k,y_k)]\} \tag{3}$$

$$y_{k+1} = y_k + \{[x_k - x_{k-1}][2R(x_k,y_k)-1]\} \tag{4}$$

Although the square trace is geometrically simpler than the hexagonal trace, the equations (3) and (4) needed to implement it are slightly more complex than the 2-state equations in [10]. The hexagonal method is more suited to contour tracing patterns in the real world because points are interrogated only once, thus combatting the effects of jitter. Troxel [29] modified the square trace to combat jitter effects also. One advantage of the square trace is that it

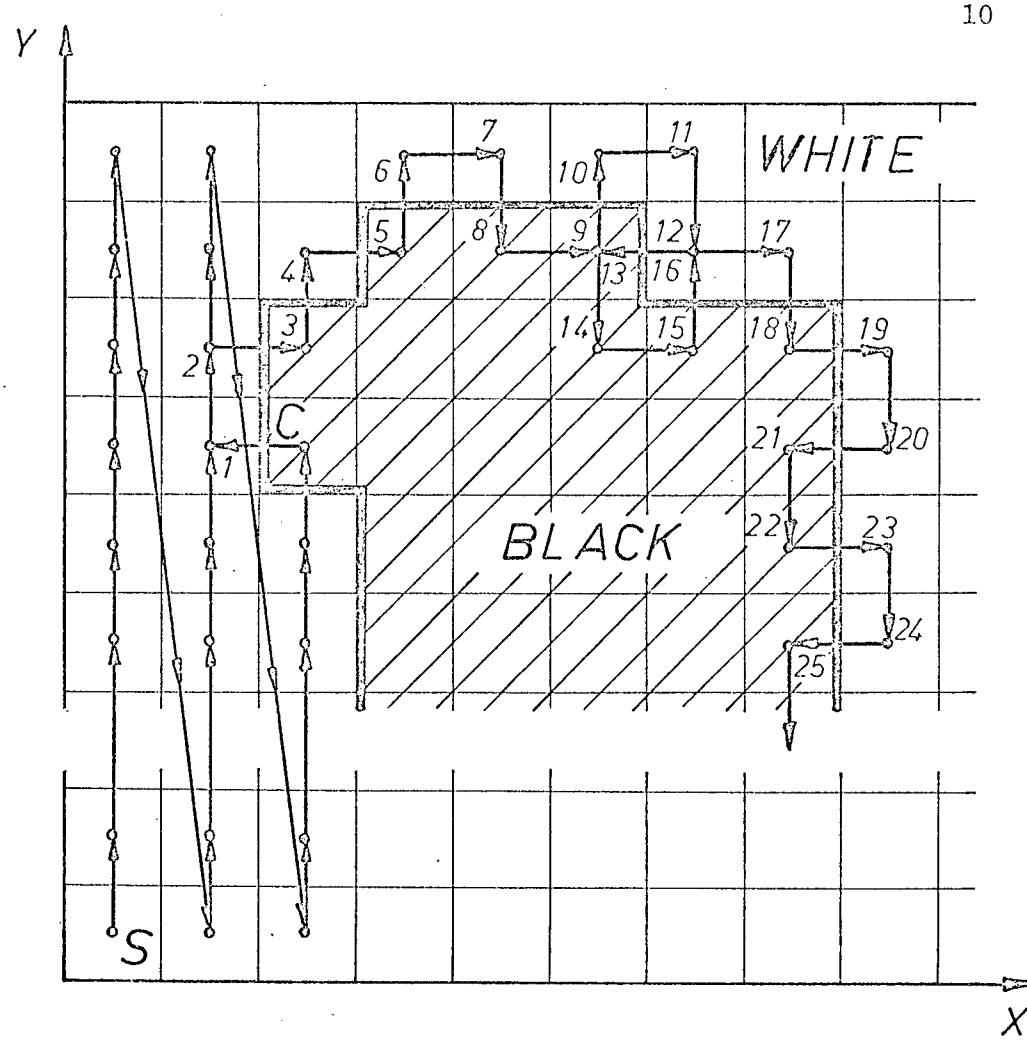


Fig. 2 Illustrating the contour trace. Search for the character begins at S. The CONTOUR mode begins at C.

can completely contour trace certain patterns containing sections which are only diagonally adjacent which cannot be traced completely by the hexagonal method. However, both schemes will contour trace any pattern that is connected in the sense defined above, and the choice of the algorithm in this thesis is strictly arbitrary. An analogous theorem to that given in [10] holds for the square trace.

Theorem

If the square tracing routine is started at the lowest black point of a non-white column, (x,y), its first move is to (x-1, y) and the trace will always return to (x-1, y) after tracing the outside contour of a connected pattern exactly once.

This theorem can be inductively proved by starting with a pattern consisting of only one square and constructing a general connected pattern by adding other adjacent black squares. A formal proof, however, will not be given here.

2.3.3 Smoothing

The same two-dimensional hysteresis smoothing technique used by Clemens [10, 25, 26] was used here. Let x_k and x_{k+1} be, respectively, the present and future x-coordinate of the raw character trace. Let x^s_k and x^s_{k+1} be, rexpectively, the present and future x-coordinate of the smoothed character trace. The algorithm can be described with three interrogations.

$$\begin{aligned} &\text{If } x^s_k < x_{k+1} - T_x/2, \text{ then } x^s_{k+1} = x_{k+1} - T_x/2, \\ &\text{If } x_{k+1} - T_x/2 \leq x^s_k \leq x_{k+1} + T_x/2, \text{ then } x^s_{k+1} = x_k, \\ &\text{If } x^s_k > x_{k+1} + T_x/2, \text{ then } x^s_{k+1} = x_{k+1} + T_x/2, \end{aligned} \tag{5}$$

where T_x is the threshold discussed in subsection 2.3.5. A similar operation is used for the y direction. The effect of this algorithm is to smooth out all extraneous extrema. The size of the extraneous extrema is determined by the relative size of T_y and T_x with respect to the height and width, respectively, of the character.

2.3.4 Gestalt Variables

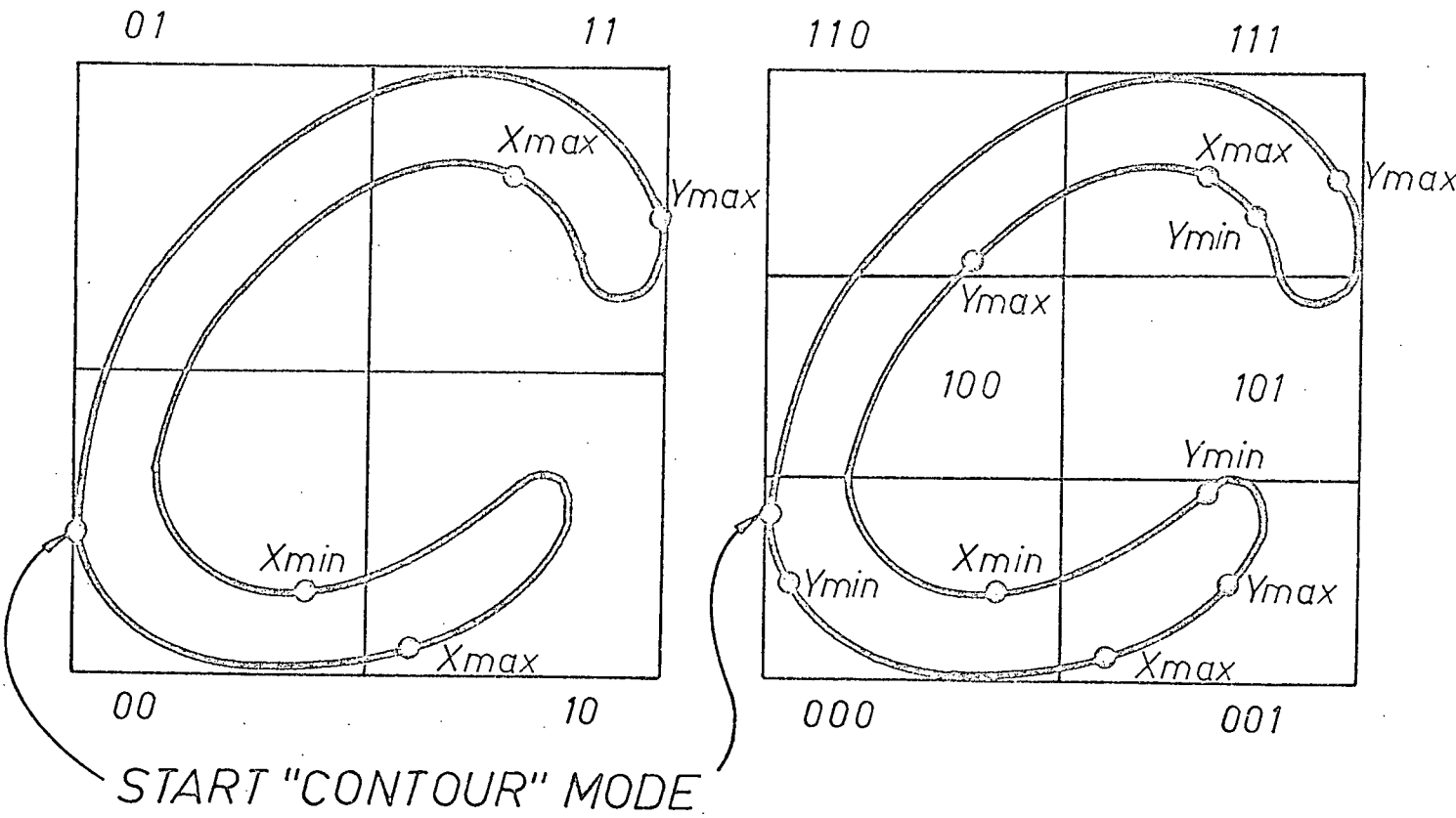
Variables which do not provide the kind of information needed to reconstruct fairly well the original pattern but which nevertheless abstract important properties of the shape as a whole are known as gestalt variables. Examples of gestalt variables are, the number of sides in a polygon, the height to width ratio (H/W) of a character, or the ratio of the perimeter to the square root of the area of a pattern ($P/\sqrt{A}$) [30]. Clemens used H/W very successfully on machine-printed characters which yielded four virtually non-overlapping distributions. It was anticipated that for handprinted characters H/W would not be satisfactory and so $P/\sqrt{A}$, which is a measure of the dispersion of a shape, was tried as well. Although $P/\sqrt{A}$ was slightly better than H/W as far as overlapping distributions are concerned, the amount of computation needed for it was too great to justify its improvement. In addition $P/\sqrt{A}$ only helped in separating categories which did not need help. Accordingly it was decided to abandon gestalt variables and look instead toward the white-to-black transition scans of section 4.2 to help with the commonly confused characters resulting from contour tracing alone.

2.3.5 Feature Vector Formation

During the SEARCH mode (Fig. 2) the scanning spot moves, point by point, from the bottom to the top of the left most column, and successively repeats this procedure on the column immediately to the right of the column previously scanned, until the first black point is found. Upon locating this point, the scanner enters the CONTOUR mode, in which the scanning spot moves according to the square trace described earlier and terminates when it completes its trace around the outside of a character and returns to its starting point.

After the character has been traced once the rectangle surrounding

the character is divided into either four or six equal-sized subrectangles whose size depends on the height and width of the letter (see Fig. 3). In previous research only the four-rectangle subdivision has been used but it was anticipated that less information from the middle of characters would be lost, due to quantization, with a six-rectangle subdivision. To further reduce quantization errors the subdivisions are assigned 3-bit labels in such a manner that the Hamming distance between any two labels is equal to the number of rectangles between the two labels, vertically and/or horizontally. A y threshold equal to one-half each rectangle's height and an x threshold equal to one-half each rectangle's width are defined and the character is contour traced for a second time. Whenever the x co-ordinate of the scanning spot reaches a local extremum and moves in the opposite direction to a point one threshold away from the resulting extremum, the resulting point is designated as either an x_{max} or x_{min}. Analogous comments apply to the y co-ordinate of the scanning spot. The starting point of the CONTOUR mode is regarded as an x_{min}. The CODE word for a character consists of a 1 followed by binary digits whose order coincides with the order in which extrema occur during contour tracing; 1 denotes x-extrema, while 0 denotes y-extrema. The ordering of the binary labels of the rectangles in accordance with the rectangles in which extrema fall in event sequence constitutes the COORD word. The feature vector consists of the concatenation of the CODE and COORD words. Fig. 3 shows the CODE and COORD words for "C". Further details relating to CODE and COORD word generation appear elsewhere [10, 11, 25-29].



CODE WORD	CODE WORD
10111	1001010010
COORD WORD	COORD WORD
00,11,11,00,10	000,111,111,111,110,000,001,001,001,000

Fig. 3 CODE and COORD words for "C".

2.4 Classification

2.4.1 Introduction

A great deal of theory has been developed during the past ten years in the field of classification theory [27]. Most classifiers fall into either the parallel or sequential type. In the parallel classifier a set of measurements is first taken on a pattern and then a decision is made. The bulk of the references and this thesis are concerned mainly with this type of classifier. When taking measurements is costly it is desirable to use sequential classification [34] in which after taking each measurement a decision is made either to take another measurement or to decide on the identity of the pattern based on the measurements taken up to that time.

Every classifier undergoes some kind of training phase, when the pattern distributions are not known a priori, in which either the distributions themselves are implicitly or explicitly estimated (nonparametric training) or in which some parameters of the distributions are estimated (parametric training). In past work Clemens [10], Chodrow [12], and Munson [13] all applied a table look-up classification scheme to the feature vectors. In this section the basic form of three parametric classifiers used, as well as the table look-up method, are described. For clarity, more meaningful comparison, and geometric interpretation, the classifiers are looked at from the discriminant function point of view [35]. For R categories there are R discriminant functions yielding R discriminants of which the largest is always chosen as representing the true category. Given R discriminant functions $g_1(\vec{X})$, $g_2(\vec{X})$, ..., $g_R(\vec{X})$, for any two of R categories i, j the equation of the decision surface between those two categories is given by $g_i(\vec{X}) - g_j(\vec{X}) = 0$. For simplicity it will be assumed in subsections 2.4.3 - 2.4.5 that all feature vectors have the same dimensionality and that all

categories have equal a priori probabilities. These special cases will be treated in 2.4.8 and 2.4.9.

2.4.2 The Table Look-up Method

The table look-up method (also exact match classifier) is a special case of the Fix and Hodges nonparametric method [36]. Given some fixed number N of training patterns in each of R categories a table is made up containing the NR feature vectors. Now, to classify an arbitrary pattern $\vec{X}$, the feature vectors in the R training subsets are pooled and a search is made of all feature vectors having zero-Hamming distance with the arbitrary pattern. Suppose that of all the zero-Hamming distance feature vectors found, n_1 belong to C_1, n_2 to C_2,...., n_R to C_R. We then set

$$\begin{aligned} g_1(\vec{X}) &= n_1 \\ g_2(\vec{X}) &= n_2 \\ &\vdots \\ g_R(\vec{X}) &= n_R \end{aligned} \tag{6}$$

and a decision is made by selecting the largest discriminant. If no zero-Hamming distance feature vectors are found then the pattern $\vec{X}$ is rejected. If the categories have some nonuniform a priori distribution then the table can be set up by using training subsets of sizes proportional to the a priori probabilities or the table can be constructed as above and the discriminants can be multiplied by the a priori probabilities. Then n_1, n_2, ..., n_R would become $n_1p(C_1)$, $n_2p(C_2)$,..., $n_Rp(C_R)$ where $p(C_i)$ are estimates of the a priori probabilities $P(C_i)$. Clearly this method is an attempt to estimate the values of $P(\vec{X}|C_i)P(C_i)$ for i=1, ..., R. For a low reject rate a large number of training samples are needed, especially with patterns of high variability such as handprinted characters. This requires a great deal of storage as well as a

rapid-access memory [35]. This method will be referred to as algorithm 'S'.

2.4.3 The Parametric Bayes Approach

Given a feature vector $\vec{X}$ it is desired to maximize the a posteriori probability $P(C_i|\vec{X})$ over i or an increasing monotonic function of it as in (1). From (1) and (2), assuming equiprobable characters all with d dimensional feature vectors and taking logarithms we get the discriminant function

$$g_i(\vec{X}) = \sum_{k=1}^{d} \ln P(x_k|C_i) \tag{7}$$

where the $P(x_k|C_i)$ for i=1, ..., R and k=1, ...,d are the parameters of the distributions $P(\vec{X}|C_i)$. When a discriminant function is linear, its decision boundary is a hyperplane and it can be very easily implemented in terms of threshold logic elements [35]. Any linear discriminant function can be expressed as a linear equation in terms of the components of $\vec{X}$ as

$$g_i(\vec{X}) = \sum_{k=1}^{d} \omega_{i,k} x_k + \omega_{i,d+1} \tag{8}$$

where the orientation of the hyperplane is determined by the weights $\omega_{i,k}$ for k=1, ..., d, and the position of the hyperplane is determined by the weight $\omega_{i,d+1}$. It can be shown [37] that for binary measurements (7) can be written in the form of (8) where

$$\omega_{i,d+1} = \sum_{k=1}^{d} \ln P(x_k=0|C_i) \tag{9}$$

and

$$\omega_{i,k} = \ln P(x_k=1|C_i) - \ln P(x_k=0|C_i).$$

2.4.4 Minimum Euclidean Distance from the Means

In the Euclidean distance classifier a metric, monotonic to the Euclidean distance such as the square of the Euclidean distance, is minimized over C_i, i=1, ..., R [35, 38]. This metric is taken between the given pattern $\vec{X}$ and the mean of each category $\vec{X}_{m,i}$. Let the mean $\vec{X}_{m,i} \triangleq (\bar{x}_{1,i}, \bar{x}_{2,i}, \ldots, \bar{x}_{d,i})$ where $\bar{x}_{k,i}$ is the mean of the k'th component of the training feature vectors for category C_i. A decision is then made by minimizing

$$\sum_{k=1}^{d} (x_k - \bar{x}_{k,i})^2 \tag{10}$$

which can be put in the linear discriminant function form of (8) where

$$\omega_{i,d+1} = -\frac{1}{2} \sum_{k=1}^{d} [P(x_k=1|C_i)]^2$$

and

$$\omega_{i,k} = P(x_k=1|C_i) \tag{11}$$

when the measurements are binary valued. Equation (11), however, will not be derived in this thesis.

2.4.5 A Non-Euclidean Distance Classifier

A metric difficult to find in classification theory but used in sequential decoding for telephone signals is the decoding distance described in [39]. This distance is defined as

$$\sum_{k=1}^{d} |x_k - \bar{x}_{k,i}|. \tag{12}$$

Although, in general, (12) implements a nonlinear decision boundary, it will be shown in the next subsection that, for binary measurements, it is equivalent to a linear discriminant function closely related to (7)

2.4.6 Some Important Interrelationships

For the three classifiers (7), (10) and (12) the prototype or mean point for each category lies somewhere inside a hypercube. For any two categories C_i, C_j there are two mean points which can have certain symmetries with respect to the sides and vertices of the hypercube depending on the distributions of the binary feature vectors. Certain interesting relationships arise between (7), (10) and (12) under certain types of symmetry. In addition, the following interesting theorem to classification theory is proved.

Theorem

The minimum decoding distance classifier, (12), which, in general, implements a nonlinear decision boundary is equivalent to a linear discriminant function when the measurements are binary and the categories have equal a priori probabilities.

Proof

For binary measurements

$$\bar{x}_{k,i} = P(x_k=1|C_i). \tag{13}$$

The minimum decoding distance classifier for equiprobable categories can be written as

$$\underset{i}{\text{Min}} \{ \sum_{k=1}^{d} |x_k - P(x_k=1|C_i)| \}. \tag{14}$$

When $x_k=1$, $|x_k - P(x_k=1|C_i)| = P(x_k=0|C_i)$.

When $x_k=0$, $|x_k - P(x_k=1|C_i)| = P(x_k=1|C_i)$.

$\therefore$ (14) can be written as

$$\underset{i}{\text{Min}} \{ \sum_{k\epsilon a} P(x_k=0|C_i) + \sum_{k\epsilon b} P(x_k=1|C_i) \} \tag{15}$$

where a is the set of k's for which $x_k=1$

and b is the set of k's for which $x_k=0$.

Since $\sum_k P(x_k=0|C_i)$ is inversely monotonic in i to $\sum_k - P(x_k=0|C_i)$ and to $\sum_k [1-P(x_k=0|C_i)]$ and since $\sum_k [1-P(x_k=0|C_i)] = \sum_k P(x_k=1|C_i)$, (15) can be written as

$$\max_i \{ \sum_{k\epsilon a} P(x_k=1|C_i) + \sum_{k\epsilon b} P(x_k=0|C_i) \}$$

which is equal to the discriminant function

$$g_i(\vec{X}) = \sum_{k=1}^{d} P(x_k|C_i). \tag{16}$$

(16) can be written as

$$g_i(\vec{X}) = \sum_{k=1}^{d} [x_k P(x_k=1|C_i) + (1-x_k)P(x_k=0|C_i)]. \tag{17}$$

Recombining terms yields

$$g_i(\vec{X}) = \sum_{k=1}^{d} [P(x_k=1|C_i) - P(x_k=0|C_i)]x_k + \sum_{k=1}^{d} P(x_k=0|C_i) \tag{18}$$

which is a linear discriminant function. Q.E.D. This algorithm will be called 'TL'. Note its similarity to (7). It can be generated by taking the anti-logarithm of every term in the summation of (7).

The relationship between (7), (10), (12) and TL can easily be observed when the symmetries are viewed in two-dimensional space. Let the following symmetries be defined for any i and j, $i \neq j$:

'A' symmetry
$$P(x_1=1|C_i) = P(x_1=0|C_j)$$
$$P(x_2=1|C_i) = P(x_2=1|C_j)$$

'B' symmetry
$$P(x_2=1|C_i) = P(x_2=0|C_j)$$
$$P(x_1=1|C_i) = P(x_1=1|C_j)$$

'C' symmetry
$$P(x_1=1|C_i) = P(x_2=0|C_j)$$
$$P(x_2=1|C_i) = P(x_1=0|C_j)$$

'D' symmetry
$$P(x_1=1|C_i) = P(x_1=0|C_j)$$
$$P(x_2=0|C_i) = P(x_2=1|C_j)$$

It can be shown that (10) and TL implement the same decision line when either A, B, C or D symmetries are observed, (7), (10) and TL implement the same decision line when either A, B or C symmetries are observed, and (7), (10), (12) and TL all implement the same decision line when A or B symmetries are observed. If none of the above symmetries are observed all four decision boundaries are different but some may still effect the same decision for binary feature vectors. For example, (12) and TL always implement the same decision for binary feature vectors, and (7), (10), (12) and TL do so under C symmetry. These ideas can be extended to treat hyperspace. Since in realistic problems these symmetries are likely to occur infrequently it is reasonable to expect different results for the three different algorithms. Fig. 4 shows an example of the decision boundaries, implemented by (7), (10), (12) and TL, for the two-dimensional case when the above symmetries are not observed.

2.4.7 Handling Feature Vectors in Multispace

In order to implement (7), (10) or (12), certain modifications have to be made in order to handle vectors in multispace, a problem which seldom arises in character recognition. In this thesis three approaches to the problem are taken. Let $\bar{x}_{k,i,j}$, $P(x_k|C_i,L_j)$ and L_j stand for the mean of the k'th component of the i'th category yielding feature vectors of the j'th length, the probability of the k'th component conditioned on category i of length j, and the j'th length, respectively. In addition let M be the maximum value of L_j.

In the first approach feature vectors of length $L_j < M$ were made to have lengths equal to M by setting $x_k=0$ for $L_j < k \leq M$. Index i was then chosen to maximize W_i and minimize Y_i, where

$$W_i = \sum_{k=1}^{M} \ln P(x_k|C_i) + \ln P(C_i) \qquad (19)$$

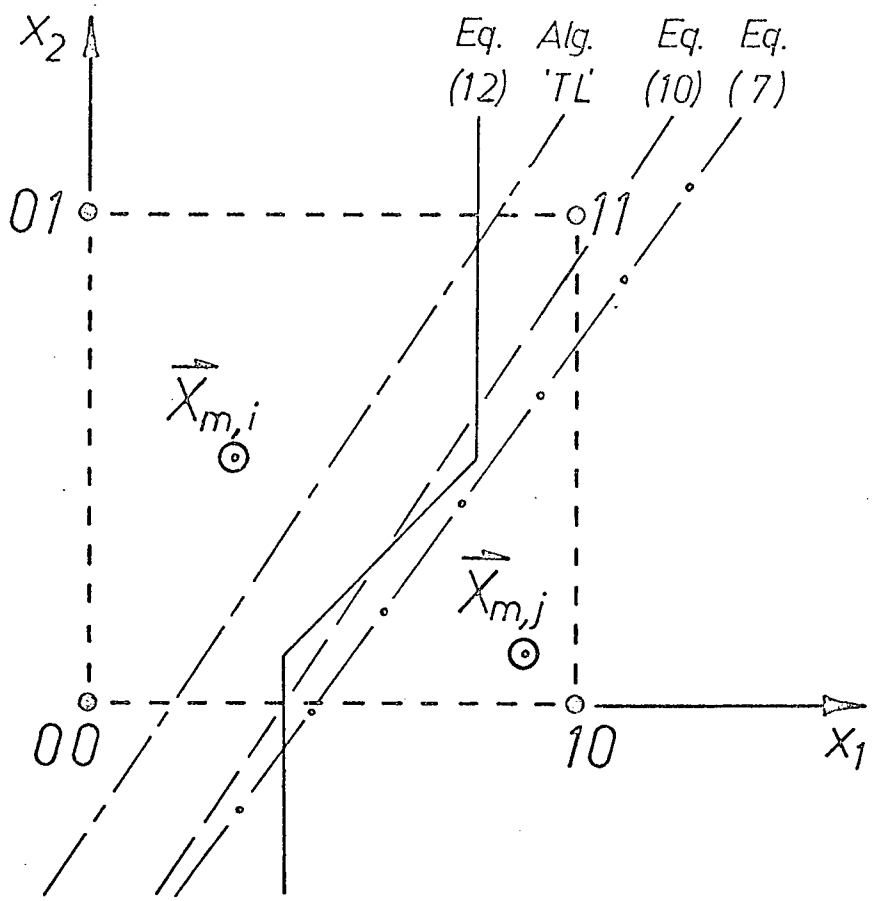


Fig. 4 Some decision boundaries between two pattern-class-means in two dimensional space.

$$Y_i = \sum_{k=1}^{M} |x_k - \bar{x}_{k,i}| - \ln P(C_i). \qquad (20)$$

Algorithms W and Y above are the same as (7) and (12), respectively, with a priori probability bias terms added. In W the bias term follows from a decision theoretic development, in Y the bias term is arbitrary but reasonable.

In the second approach, given an unknown character yielding a feature vector of a particular length L_j, only the discriminants for the categories C_i containing a prototype of the same L_j were calculated. Index i was then chosen to minimize U_i and V_i, where

$$U_i = \sum_{k=1}^{L_j} |x_k - \bar{x}_{k,i,j}| - \ln P(C_i) \qquad (21)$$

$$V_i = \sum_{k=1}^{L_j} (x_k - \bar{x}_{k,i,j})^2 - \ln P(C_i) \qquad (22)$$

Algorithms U and V above come from (12) and (10) respectively with bias terms added. Again, although the bias terms are arbitrary, they are reasonable and easy to implement because the $P(C_i)$ are known a priori.

The third approach consists of maximizing $P(\vec{X}C_i L_j)$, the joint probability that a feature vector $\vec{X}$ has L_j components and comes from a character belonging to category C_i, under statistical independence among components. As in (21) and (22) only the discriminants for the categories containing a prototype of the same L_j were calculated. Index i was then chosen to maximize T_i, where

$$T_i = \sum_{k=1}^{L_j} \ln P(x_k | C_i, L_j) + \ln P(L_j | C_i) + \ln P(C_i) \qquad (23)$$

Equation (23) follows directly from $P(\vec{X}C_i L_j)$ under statistical independence among components.

2.4.8 A Parallel-Sequential Classifier

In the parallel-sequential mode of classification some of the decisions formerly taken as final in algorithms T and U, were taken as tentative and the scans of Fig. 7(a), for algorithm U, were taken sequentially according to the decision trees of Fig. 7(b). The scans and decision trees of Fig. 7 are explained in section 4.2.

2.4.9 Compound Decisions on Strings of Dependent Characters

In the algorithms discussed so far a decision was made on individual characters solely on the basis of the measurements and the a priori probability of those characters. This mode of decision ignores the dependencies that characters may have in a given sequence. It is desirable to make use of these dependencies by making decisions on a string of characters rather than on

characters individually. In this subsection the equation for the optimum decision on a string of characters, given variable-length feature vectors, is derived using certain underlying assumptions, and a procedure for reducing the amount of computation is given.

We wish to maximize the a posteriori probability of a sequence of m characters given by

$$P(C_1^i,\ldots,C_m^i,L_1^j,\ldots,L_m^j \mid \vec{X}_1,\ldots,\vec{X}_m) \tag{24}$$

where C_n^i is the category of the n'th character in the sequence and can take on i values, L_n^j is the length of the feature vector of the n'th character and can take on j values, and $\vec{X}_n$ is the feature vector of the n'th character. Since we do not care what lengths L_n^j are associated with the decisions C_n^i, the optimum decision is arrived at by selecting the categories $C_1^i, C_2^i, \ldots, C_m^i$ which maximize (24). Using Bayes' rule (24) becomes

$$\frac{P(\vec{X}_1,\ldots,\vec{X}_m|C_1^i,\ldots,C_m^i,L_1^j,..,L_m^j)\ P(C_1^i,\ldots,C_m^i,L_1^j,\ldots,L_m^j)}{P(\vec{X}_1,\ldots,\vec{X}_m)} \cdot \tag{25}$$

Expanding the numerator of (25) and deleting the denominator because it does not depend on i yields

$$P(\vec{X}_1,\ldots,\vec{X}_m|C_1^i,\ldots,C_m^i,L_1^j,\ldots,L_m^j)P(L_1^j,\ldots,L_m^j|C_1^i,\ldots,C_m^i)P(C_1^i,\ldots,C_m^i)\,. \tag{26}$$

Assuming that,

1) $\vec{X}_n$ is independent of $\vec{X}_h$, C_ℓ and L_ℓ
   for $n=1,\ldots,m$; $h=1,\ldots,m$; $\ell=1,\ldots,m$; $n\neq h\neq \ell$,

2) L_n is independent of L_h and C_h
   for $n=1,\ldots,n$; $h=1,\ldots,m$; $n\neq h$,

(26) can be written as follows:

$$\prod_{n=1}^{m} P(\vec{X}_n|C_n^i,L_n^j) \prod_{n=1}^{m} P(L_n^j|C_n^i)P(C_1^i,\ldots,C_m^i). \tag{27}$$

Taking natural logarithms yields

$$\sum_{n=1}^{m} \ln P(\vec{X}_n | C_n^i, L_n^j) + \sum_{n=1}^{m} \ln P(L_n^j | C_n^i) + \ln P(C_1^i, \ldots, C_m^i) \quad . \tag{28}$$

If it is now assumed that the components of $\vec{X}_n = (x_1, x_2, \ldots, x_{L_n^j})$ are independent, then the optimum decision on the sequence of m characters is made by maximizing $G(\vec{X}_1, \ldots, \vec{X}_m)$ over R^m possible sequences, where

$$G(\vec{X}_1, \ldots, \vec{X}_m) = \sum_{n=1}^{m} \left[\sum_{k=1}^{L_n^j} \ln P(x_k | C_n^i, L_n^j) + \ln P(L_n^j | C_n^i) \right]$$

$$+ \ln P(C_1^i, \ldots, C_m^i). \tag{29}$$

One can vastly reduce the computation time by assuming that in an ordered list of bracketed, [], terms in (29) over C_n^i, L_n^j, the correct identity of an unknown character has a high probability of being in the top ds entries of the ordered list, where ds is called the 'depth of search' and G is maximized over $(ds)^m$ sequences. For bigrams m=2 and for trigrams m=3.

2.5 Training and Storage Requirements

Probabilities $P(\vec{X}|C_i)$, $P(x_k|C_i)$ and $P(x_k|C_i,L_j)$ were estimated or learned by determining the relative number of times a vector $\vec{X}$ or component x_k occurred, given the event C_i or the joint event C_i, L_j. Since we do not know the probability distributions of the probability values themselves we cannot make optimum estimates of those values and we are justified in using the maximum likelihood estimates described above.

Algorithm S must learn and store $P(\vec{X}|C_i)P(C_i)$ for virtually all $\vec{X}$ which occur in a test set. Algorithm T needs only $P(x_k|C_i,L_j)$, $P(L_j|C_i)$ and $P(C_i)$. In addition to $P(C_i)$, for binary measurements, U and V need $P(x_k|C_i,L_j)$ while W and Y need $P(x_k|C_i)$. Algorithm G needs $P(x_k|C_i,L_j)$, $P(L_j|C_i)$ and $P(C_1^i,\ldots,C_m^i)$. Note that, under the assumptions made, G learns the

same about the measurements as does T but needs to store R^m instead of R a priori probabilities and needs to search over the same number of discriminants.

III. EXPERIMENTS

3.1 Introduction

Although the whole system of Fig. 1 can be simulated in one program, too much computation would be duplicated in performing the many experiments. Accordingly, each section of the recognition problem was simulated separately. The contour tracer was simulated using an IBM-7044 computer into which punched cards containing all the data had been read, and out of which punched cards containing the feature vectors were obtained. Hence this operation was performed once for 4-PAD and once for 6-PAD only.* All the classifiers were simulated on an IBM-360/67 computer where the feature vectors were stored in a file. Since the training for algorithm G is the same as that for T, all the likelihoods for each feature vector in the T experiments were stored on tape together with bigram and trigram statistics. Hence the bigram and trigram programs operated only on the likelihoods of the feature vectors, thus saving a great deal of classification computation and reducing the bigram and trigram programs to combinational operations for the most part.

The purpose of the experiments in this thesis is five-fold; 1) to compare the six-part area division (6-PAD) feature extraction scheme with 4-PAD, 2) to compare the various classification algorithms in terms of performance and storage requirements, 3) to observe the effects of monogram, bigram and trigram contextual constraints in terms of performance, classifier complexity, and added storage requirements, 4) to obtain reasonable predictions of the probability of error on real world data, and 5) to compare performance on an individual with that on the population.

In early experiments researchers in the field predicted the performance of a recognition scheme by obtaining a sample of data, training the

---

* 4-PAD and 6-PAD refer to four and six-part area division, respectively.

algorithm on that sample and testing it on the same sample. This method is known as the R method (resubstitution) [42].

It is now well known [40] that this procedure yields a biased over-optimistic prediction of performance. Highleyman [41] showed that for very large fixed samples of data there exists an optimum partitioning of the set into a training set and a testing set in order to best predict the performance, with the result that the testing set should never be smaller than the training set. For small data sets, however, Highleyman's method breaks down and yields an overpessimistic estimate of performance [40]. This procedure is referred to as the H method (holdout). In 1968 Lachenbruch and Mickey [42, 43] showed that for small sample sizes a much better estimate of performance consists of training the classifier on all but one pattern in the set and then testing on the pattern left out during training, repeating this procedure until every pattern of the data has been used as a test pattern. Needless to say, unless the data size is very small, a great deal of training computation is needed in this method referred to as the U method.

In this thesis a variation of the U method is used to predict performance. The method and its advantages are described in section 3.3.

3.2 Description of the Data

A total of twenty upper case alphabets were obtained from seven different persons, all of whom were required to print on good quality graph paper in squares 0.40 in. high by 0.50 in. wide. A Paper Mate thin-felt-tip pen was used. Each person was asked to leave the outside contour of each character unbroken. All seven persons printed the alphabet once from A-Z, as shown in Fig. 5, and once from Z-A. Two persons printed three additional alphabets by copying three randomized lower case alphabets. An enlarger-projector of magnification ten was used to project each character onto an

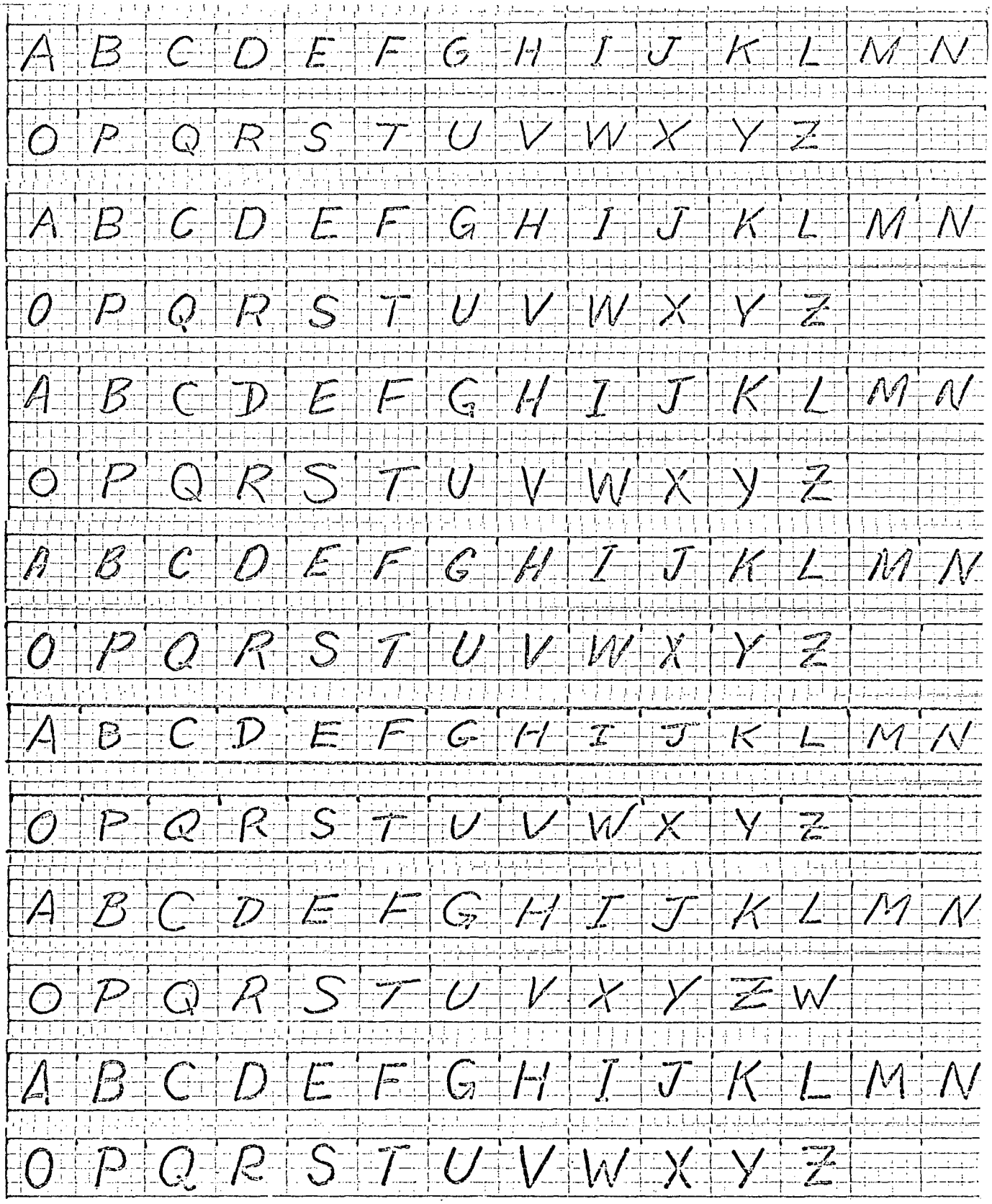


Fig. 5 Alphabets from seven persons. The sixth and seventh alphabets are from PERSONS 1 and 2, respectively.

area divided into 2500 0.1 in. squares. If more than 50% of any square's area was black the square was blackened and assigned a value 1. Otherwise the square was left white and assigned a value 0. Each spatially quantized character was then encoded onto IBM punch cards by experienced keypunch operators.

The legibility of the data was estimated by asking 10 graduate students to identify all 520 magnified spatially quantized characters. Each student responded verbally to characters viewed one-at-a-time in random order at close range for as long as desired. All viewing was done at one session which lasted approximately 12 minutes per student. The recognition accuracy averaged over the first two alphabets of all seven persons and over all 10 viewers was 99.7%. Averaging over all 10 viewers and over the five alphabets from PERSONS 1 and 2 yielded 99.8% and 100%, respectively. Machine recognition errors in excess of a few per cent would therefore result from weaknesses in either feature extraction or classification, or both.

3.3 Experiments on Independent Characters

The experiments on the test data over the population (POP) were performed using the first two alphabets from five persons for training, and two alphabets from each of the two remaining persons for testing. The results were averaged over seven trials. In the i'th trial, data from PERSONS i and i+1, i=1,2,...,6, were used for testing. In the seventh trial data from PERSONS 7 and 1 were used for testing. For tests on data from one individual, four alphabets were used for training and the remaining one for testing. The results were averaged over five trials in which a different alphabet served as the test data for each trial. This procedure for estimating the performance is a compromise between the H method and the U method [40]. Although the U method would be a better estimate and would probably predict a better performance, too much training computation would be required with the data set above.

The method used here, in addition to being more economical on data than the H method, involves far less training computation than the U method. It is, thus, a reasonable method to use and if anything probably yields a slightly negatively biased estimate of performance. For experiments on the training data in the POP tests the first two alphabets from each person were used. In tests on alphabets from PERSONS 1 and 2 all five alphabets were used.

The experimental procedures described above yielded maximum likelihood estimates of $P(\varepsilon|C_i)$, i=1,2,...,26, i.e., the probability of error conditioned on pattern class C_i, for algorithms S, T, U, V, W and Y. To calculate the total probability of error, $P(\varepsilon)$, for each algorithm, the estimates of $P(\varepsilon|C_i)$ were substituted into

$$P(\varepsilon) = \sum_{i=1}^{26} P(\varepsilon|C_i)P(C_i) \qquad (30)$$

For the experiments on equiprobable characters (EQP), $P(C_i)$ was made equal to 1/26 for all i in (30) and in the classification algorithms. For the experiments on English a priori probability characters (ENG), 5-decimal digit estimates of the values of $P(C_i)$ were obtained from Pierce [44].

3.4 Experiments on Dependent Characters

Algorithm G, eq. (29), was used with bigram (m=2) and trigram (m=3) contextual constraints. Maximum likelihood estimates for the bigram and trigram a priori probabilities were obtained by dividing their frequency of occurrence, given in Pratt [45], by their total number observed. There were 304 and 2,510 bigram and trigram entries, respectively. All other possible combinations of two and three characters were considered illegal. Accordingly, when the depth of search (ds) was so small that all the prospective strings of m characters were illegal, a decision was made on individual characters without using context. However, the likelihood of observing legal bigrams and trigrams in the search made by G rapidly increases, varying with the square and cube

of ds, respectively.

The experiments on the test data were performed using the first two alphabets from five persons for training and forming samples of bigrams and trigrams out of the two alphabets from each of the two remaining persons for testing. The results were averaged over seven trials as described in section 3.3. In each trial $2(2)^m$ samples of bigrams or trigrams were formed for each entry, where 2^m samples are formed from the two character-samples of each testing person. For experiments on the training data the first two alphabets from each of the seven persons were used for training and $7(2)^m$ bigram or trigram samples per bigram or trigram entry were formed, as described above, for testing.

The experimental procedures described above yielded estimates of $P(\varepsilon|Ci_1,Ci_2,\ldots,Ci_m)$ for $i_1=1,\ldots,26$; $i_2=1,\ldots,26$; $\ldots$; $i_m=1,\ldots,26$, where any combination of $i_1,i_2,\ldots i_m$ was a legal string of m characters. The total probability of error was then calculated using

$$P(\varepsilon) = \sum_{i_1=1}^{26} \sum_{i_2=1}^{26} \cdots \sum_{i_m=1}^{26} P(\varepsilon|Ci_1,Ci_2,\ldots,Ci_m)P(Ci_1,Ci_2,\ldots,Ci_m) \qquad (31)$$

over all the legal strings of m characters.

The probability of error was calculated for various values of the depth of search, for bigrams up to ds=16 and for trigrams up to ds=5, using POP training data with 4-PAD feature extraction. Having found an experimental optimum value of ds, $P(\varepsilon)$ was calculated for the 4-PAD TS case and for the 6-PAD TR and TS cases.

IV. RESULTS

4.1 Introduction

Although the results of main interest are those coming from the test set, results on the training set are included for three reasons. (1) The smaller the difference between the performance of a classifier on the training set and on the testing set is, the smaller the amount of data needed altogether becomes. In other words, if both methods yield comparable results one can be reasonably confident that enough data has been collected. This may be helpful to know when data is difficult to collect. (2) Given a small data set, if two classifiers perform equally well on the testing set it may do well to decide on the classifier yielding better performance on the training set. Since if a large data set becomes available one would expect the test set results to lie between the previous test set and training set results, the classifier previously yielding better performance on the training set may subsequently yield better performance on the testing set. (3) If the small data set available is an unbiased estimate of the future characters to come then the performance on the training data of the small set is a reasonable estimate for the performance on the future characters. For other comments see also [46].

4.2 Independent Characters

Results obtained using algorithms S, T, U, V, W and Y of subsection 2.4.7 appear in Fig. 6. EQP means equiprobable characters; ENG means that the $P(C_i)$ assumed the probabilities in English text. TS and TR denote test set and training set, respectively, while 6 and 4 apply to the area division. The first, second, and third squares apply, respectively, to data from the population, PERSON 1 and PERSON 2. The numbers which are neither in brackets nor parenthesis indicate misclassification (error plus reject) probabilities when rejects do occur; if no rejects occur the numbers indicate error probabilities. Numbers

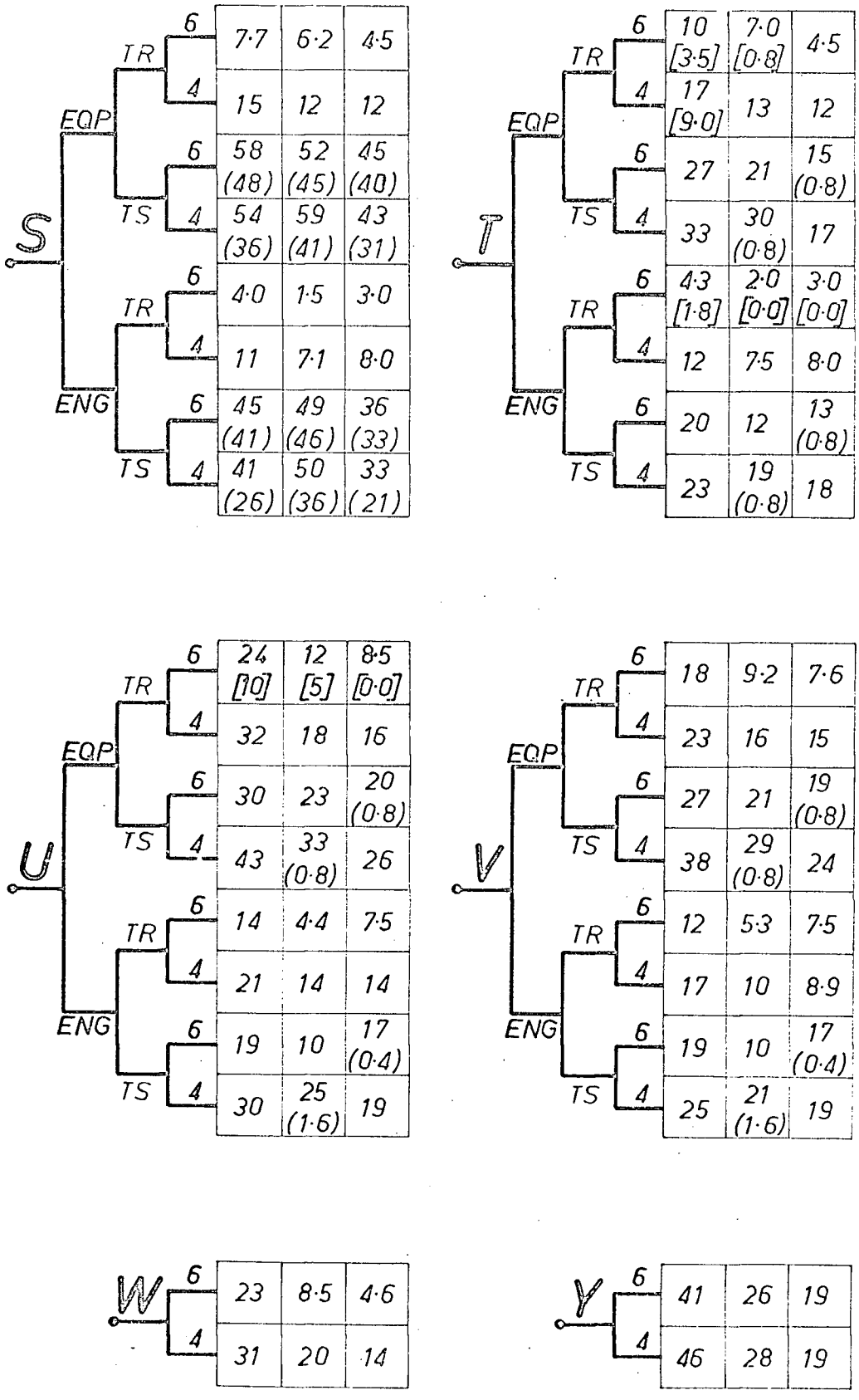


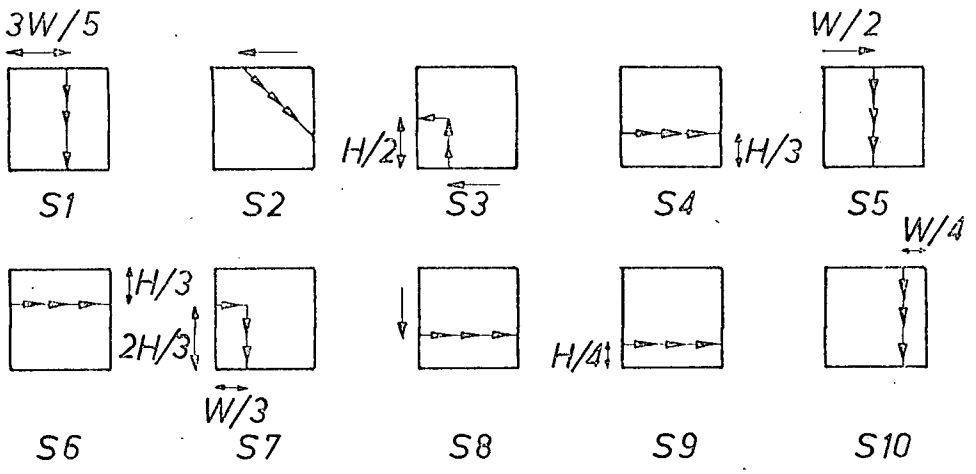
Fig. 6 Misclassification and reject probabilities in per cent.

in parentheses denote the reject probability and numbers in brackets denote the error probability when the scans of Fig. 7 were used to resolve confusions. For algorithms W and Y, results are for equiprobable characters on the training set.

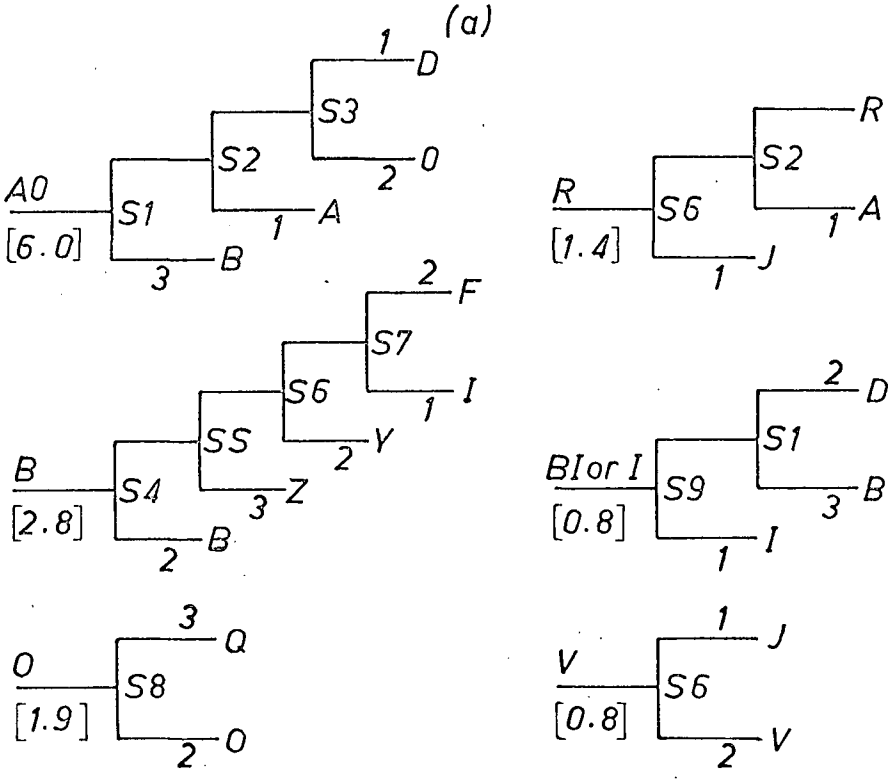
Over POP's 14 alphabets the average vector length was 17 and 25 for 4-PAD and 6-PAD, respectively.

The procedure used to obtain finite estimates of $P(\vec{X}|C_i)$ and $P(x_k|C_i,L_j)$ makes rejection of any character unnecessary. For example, when $P(\vec{X}|C_i)P(C_i)$ = zero for all i, the optimum decision is to select any one of the 26 characters as being correct. The number in parenthesis in Fig. 4 merely indicates the probability that the optimum choice is any one of the 26 characters; in many recognition schemes, however, the unknown characters would be rejected in such cases. Thus, rejects are indicated for algorithm S whenever a feature vector different from any encountered during training occurs during testing, and for T, U and V whenever the length of the test vector is different from any encountered during training. No rejects occur, as expected, when testing is done on the training data. For all algorithms, when the discriminants of two or more categories were equal, the unknown character was classified on the basis of the first discriminant in alphabetic order.

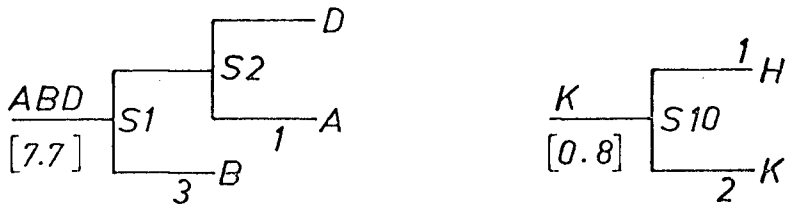
Some scans used to differentiate easily confused characters appear in Fig. 7. Fig. 7-(a) shows the scans for resolving character confusions that were used on the training data with algorithm U, 4-PAD, and equiprobable characters. Single headed arrows external to the boxes indicate that the scan begins at the leftmost black point in the top and bottom rows, respectively, of S2 and S3, and at the lowest black point in the leftmost column of S8. Character width and height is W and H, respectively. Double headed arrows accompany the height or width fraction at which the scans occur. Fig. 7-(b) shows



(a)



(b)



(c)

Fig. 7 Scans and decision trees for resolving character confusions.

the decision trees for the population. The number on a tree branch indicates the number of white-to-black (WB) transitions resulting from the scan shown at the previous node. Thus, if U_i in equation (21) is minimum for both C_i = A and C_i = 0, then S1 is applied; if three WB transitions occur, final classification is B. Otherwise S2 is applied followed, if necessary, by S3. The decrease in error probability for any given tree appears in square brackets. Fig 7-(c) shows the decision trees used for PERSON 2. Use of these scans and decision trees on the training data with algorithm U reduced error probability for POP from 24% to 10%. The ABD decision tree reduced the error for PERSON 2 from 8.5% to 0.8%. A procedure similar to the one in Fig. 7 yielded the other numbers in square brackets in Fig. 6, although fewer scans, trees, and tree branches were needed in these other cases. Applying decision trees and scans to the ENG-6-PAD training data with algorithm T yielded recognition rates of 98.2%, 100% and 100% on the data from the popula= tion, PERSON 1 and PERSON 2, respectively.

4.3 Dependent Characters

Results obtained using algorithm C with m=2 and m=3 appear in Figs. 8 and 9. Fig. 8 shows the per cent error as a function of ds for the 4-PAD testing data. For the bigram case the per cent error was calculated for ds up to a value of 16. Note that there is no need to increase ds any further since,for the data in these experiments, 16 was the largest number of categories that could have prototype feature vectors of the same length as that of the unknown vector in question. In other words, given any test feature vector, the largest number of possible category likelihoods for that vector was 16. For the trigram case calculation of the error rate stopped at ds=5 because the amount of computer time needed for ds > 5 would have been too large. A FORTRAN program written twice for the sake of efficiency

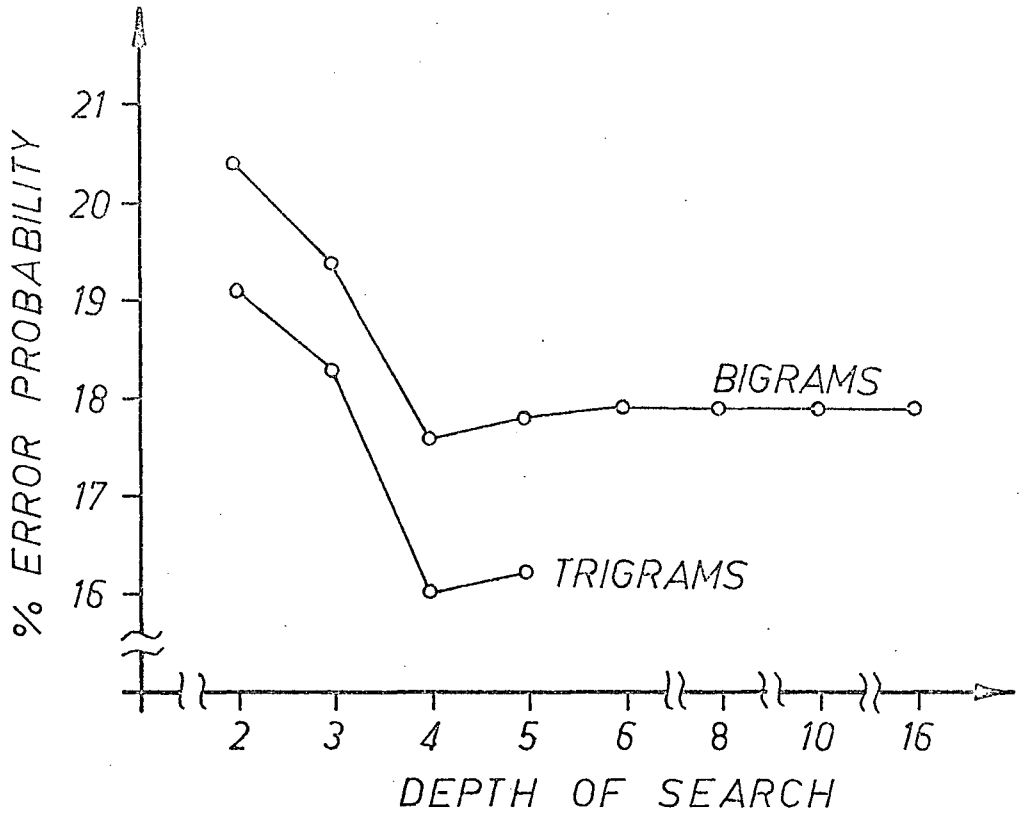


Fig. 8 Per cent error probability as a function of ds for the 4-PAD testing data.

took almost 2 hours of CPU time on the IBM-360/67 for the trigram case with ds=5.

Both error rates on Fig. 8 decrease at an accelerating rate reaching a minimum at ds=4, after which they increase at a decelerating rate. The bigram $P(\varepsilon)$ rapidly becomes uniform at a value of 17.9%. Since the data distributions are the same for the bigram and trigram cases and since the same underlying assumptions of algorithm G are made for bigrams and trigrams one would expect that the trigram error curve for ds > 5 would behave in a similar fashion to the bigram error curve.

Fig. 9 shows the per cent $P(\varepsilon)$, for the 4-PAD and 6-PAD cases on the training and testing data sets, as a function of the order of contextual information used. Zero order represents equiprobable characters, first order

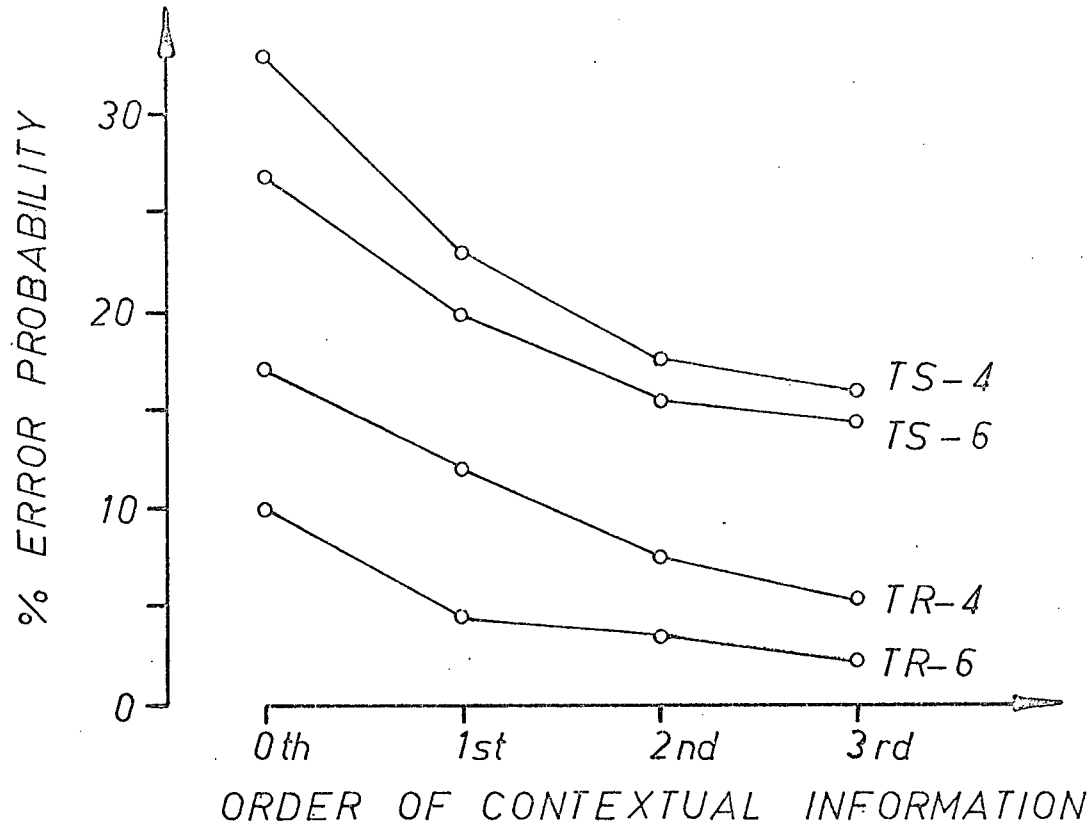


Fig. 9 Percent error probability as a function of the order of contextual information used.

represents monogram statistics, etc.. Note that the zero and first order results come from algorithm T which requires the same measurement training as G but cannot make use of character dependencies. The bigram and trigram values of $P(\varepsilon)$ are those for ds=4. With trigrams and 6-PAD algorithm G yields 86% and 98% correct recognition on the testing and training data respectively. A marked improvement is noted in all four cases in going from zero to third order contextual information. The error rate for TS-4-PAD, for example, reduces from 33% to 16%, i.e., an error reduction of more than 50%.

V. DISCUSSION AND CONCLUSIONS

5.1 Evaluation of Results

Although algorithm S yielded the lowest $P(\varepsilon)$ on the training set with independent characters, the difference between S and T is small. This suggests that the assumption of statistical independence among feature vector components is reasonable. Algorithms T, U and V all performed much better than S when the training and test data were disjoint, which indicates that in such circumstances suboptimum classifiers having relatively few parameters may be significantly better than optimum classifiers requiring extensive training. This conclusion is similar to the one reached by Hughes [47] which was that when a pattern recognition problem is selected at random from all possible problems, the optimum number of theoretically possible feature vectors decreases with the amount of training data. For equiprobable characters T performs better than U and V. This is reasonable because both U and V do not have $P(L_j|C_i)$ available as does T. In addition it can be shown (theorem in subsection 2.4.6), that for equiprobable characters and binary measurements, U can be generated by omitting the $P(L_j|C_i)$ term from T and taking the antilogarithm of the remaining terms. If, in addition, the components of the feature vectors are statistically independent then U must be inferior to T.

When the $P(C_i)$ of English text are used the difference between T, U and V on the testing data with 6-PAD becomes minimal and, in fact, U and V perform equally well and both perform slightly better than T. This clearly indicates that when storage capacity is limited a trade-off exists between contextual constraints and measurement statistics. Suboptimum classifiers which use less measurement statistics and more contextual data may well be superior to optimum classifiers which make little or no use of existing contextual constraints.

Comparison of the results for T vs. W and U vs. Y shows that neglecting the information contained in the length of the feature vectors, by making them equal in length through the addition of zeros, makes recognition more difficult. It should be kept in mind that although the feature vectors in W and Y are made longer by adding zeros, W and Y in fact use up much less storage than T and U because they store probabilities for C_i rather than C_i,L_j categories.

For T and U recognition accuracy improves as the number of scans and decision tress increases. Although the experiments show that the scans can indeed improve recognition, to fully appreciate their capability, more experiments on large testing sets would have to be performed.

Considerable improvement results using 6-PAD rather than 4-PAD with all algorithms except S. It was expected that S would perform worse with 6-PAD on the testing data because S has no generalization capability and a larger number of different feature vectors have to be learned with 6-PAD. Clemens [10] used the binary vector resulting from 4-PAD, along with two binary digits describing one of the four H/W ratios observed, to recognize 260 upper case alphabetic characters from 10 different type fonts (algorithm S was used). Even on the training data 15% error resulted although, by using x and y thresholds equal to 3/16 the letter height, the error was reduced to 3%. Important extrema on many of the upper case characters occur in squares 100 and 101 in Fig. 3, and these extrema are more accurately located by a six-part than a four-part COORD word. In addition, characters that are commonly confused under 4-PAD, such as P and D, due to the fact that the x_{max} occurs in square 10 for both P and D, are not confused any more with 6-PAD because the x_{max} lies in rectangle 101 for P and 001 for D in Fig. 3. In applying Clemens' technique to the recognition of upper case handprinted characters Munson [13] obtained 42% misclassification of which 19% was reject when 2340 characters from seven writers were used for training and an (apparently)

comparable number of samples from different individuals was used for testing. These results are for x and y thresholds of 1/10 a character's width and height, respectively. Perhaps Munson's results would improve considerably if, in addition to using a six-part area, either a larger number of training samples were used or the experiment were repeated using algorithm T rather than S.

On completely unconstrained characters the feature extraction scheme in this thesis may encounter some difficulty with broken characters such as "R" in Fig. 10. Two methods of attacking this problem are suggested in the next section. Of course the problem of broken characters can also be solved by simply requiring that people avoid printing them, a constraint which apparently in these experiments, did not cause the printer real difficulty although some was reported in [12].

All algorithms for all cases performed better on individuals than on the population and in some cases the improvement was pronounced. For some cases algorithms T and U yielded recognition accuracies of 100% on the training data of PERSONS 1 and 2 when the scans were used.

As was expected bigram and trigram information with algorithm G decreased $P(\varepsilon)$ for all cases. Looking at the TS 4-PAD and 6-PAD error curves of Fig. 9 one can see that they tend to follow a decreasing exponential path. This suggests that a decreasing amount of new contextual information becomes available by increasing further the order of the contextual information used. In fact the curves suggest that $P(\varepsilon)$ would not be decreased significantly by going beyond 3rd order contextual information. The curves also show that as the order of context increases the difference in performance between 6-PAD and 4-PAD decreases although it still remains important, noting that 6-PAD with bigrams results in a lower $P(\varepsilon)$ than 4-PAD with trigrams. In addition it is much easier to implement 6-PAD with bigrams because the storage saving in going from trigrams to bigrams is much greater than the extra storage

needed by the slightly longer feature vectors. These results illustrate the importance of balancing appropriately the information from the measurements and from context as suggested by Raviv [23]. In addition, Fig. 8 shows that it is also important to balance contextual information from the order of the context used with that obtained from the depth of search. Although for every value of ds $P(\varepsilon)$ is lower for trigrams than for bigrams, for bigrams with ds=4 it is lower than for trigrams with ds=3. In addition, much less storage is needed for bigrams, and the classification computation is reduced since,for the former case, 4^2=16 a posteriori probabilities are searched or 8 per character, while,for the latter case, 3^3=27 or 9 probabilities per character are searched.

It is interesting to note that the experimental optimum value of ds with respect to $P(\varepsilon)$ does not correspond with that predicted by eq. (29). This suggests that the assumptions made in subsection 2.4.9 in the derivation of (29) are not entirely valid. However, the assumptions are nevertheless necessary in order to make the solution feasible. As was noted in subsection 2.4.9 the curves of Fig. 8 are expected to depend also on the distribution of the correct labels (characters) with respect to ds.

It is reasonable to expect $P(\varepsilon)$ to decrease at an accelerating rate for low values of ds because for very low values of ds less contextual information is made use of. For example, out of the 17,576 theoretically possible trigrams only 2,510 are English entries in these experiments; when ds has a value of 2 only 2^3 or 8 out of the possible 17,576 are looked at. One can see that there is a significant probability that the 8 trigrams looked at are part of the 15,066 illegal trigrams after which a decision is made individually without the use of context. This was observed to happen quite frequently for ds=2 and less frequently for ds=3. In spite of the frequent disregard for context when ds=2, $P(\varepsilon)$ for bigrams was still slightly lower than that resulting from the use of monograms.

It would seem from this data that the bulk of the correct labels occurs for ds $\leq$ 4. This would help to explain the slight increase in $P(\varepsilon)$ for ds > 4. If the greater part of the correct labels occurs for ds < 4, increasing ds further, would only increase the chance of finding an erroneous bigram or trigram with a higher a posteriori probability than the correct one, and thus increase $P(\varepsilon)$.

5.2 Suggestions for Further Research

Since the requirement that people avoid printing broken characters may not be a desirable constraint, in view of the fact that some printers [12] found difficulty in overcoming this constraint, two ways of handling unconstrained data are suggested. One corrective measure is to close small breaks by blackening all white squares adjacent to a black square and repeating this procedure a certain number of times. However, if the break is more than a few units wide the character can be distorted considerably by repeating the above procedure enough times to close the break. A second alternative without this disadvantage is to move a bar of length b along the outside contour, keeping one end of the bar against the contour and maintaining an angle of 90° between the bar and the line tangent to the contour at the bar-contour point of contact. Whenever the other end of the bar encounters black during the contour trace, the line defined by that portion of the bar between the two black points is made part of the character, as in Fig. 10. The bar must be long enough to close most breaks but short enough that intentional breaks that differentiate pattern classes remain.

More experiments with and without the above methods should be performed on large unconstrained data sets.

One source of feature vector variability within pattern classes in the present scheme is the mode of searching for the character. In many characters

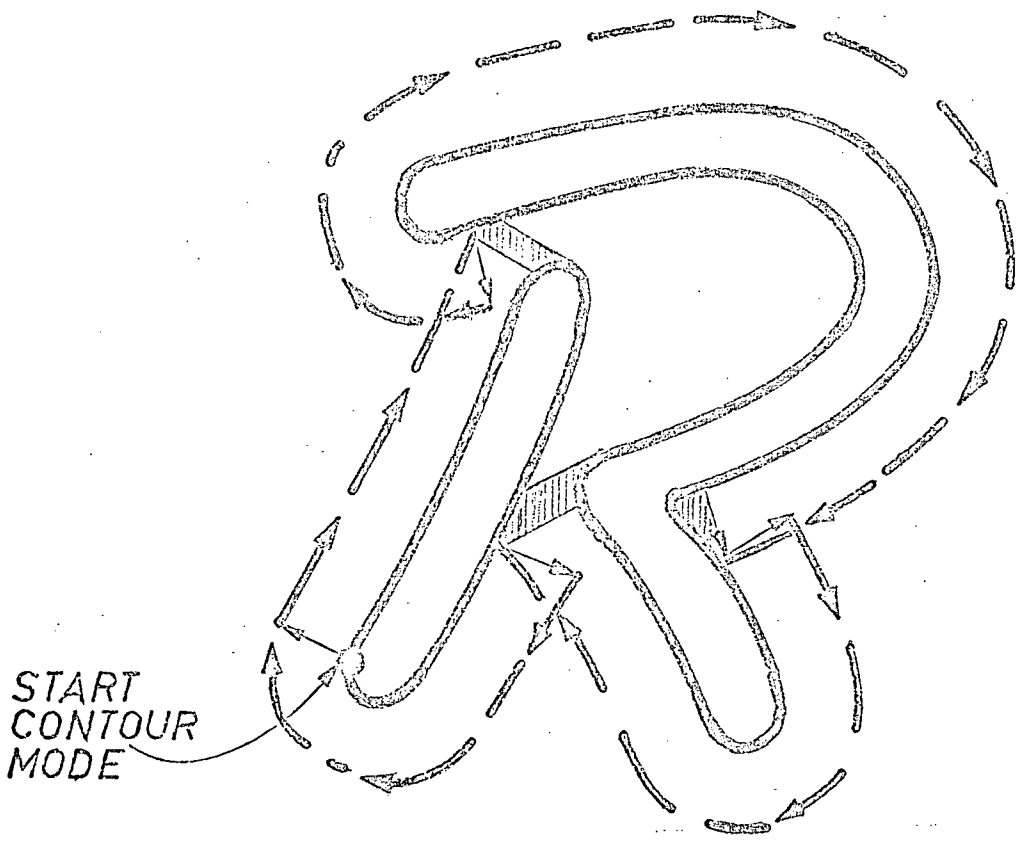


Fig. 10 Illustrating a technique for closing small breaks in broken characters.

the leftmost black part of the character may occur at the top or the bottom of the character in various samples of the same character, especially if printers slant in different ways. This causes the vertical search mode to encounter the character sometimes at the upper-leftmost part and sometimes at the lower-leftmost part, resulting in different feature vectors. In most of these cases, a diagonal search mode at 135 degrees would encounter the lower-leftmost part of a character only.

As can be seen in Fig. 6, with English a priori probabilities U performed better on TS-6-PAD than T, even though U is a simpler classifier and uses less storage. Similar methods to algorithm G for bigrams and trigrams should

be tested using U to compare with G's results.

5.3 Summary of Conclusions

Although final conclusions should wait for experiments on several large data sets, it appears that the parametric classifiers T, U and V operating on binary vectors obtained from contours of characters quantized as in Fig. 3 yield recognition accuracies much better than those obtainable using the optimum nonparametric classifier S, and that if 6-PAD is used reasonably good recognition accuracies can be achieved, particularly if English a priori probabilities of the characters are incorporated. The results also suggest that recognition schemes which use additional simple class dependent feature extraction to differentiate between commonly confused characters may perform significantly better than those which use only one feature extraction method common to all pattern classes. In addition, it appears that algorithm G using bigram and trigram statistics, with a depth of search value equal to 4, can significantly improve the recognition accuracy. These conclusions support those of Bakis et al. [5] that curve-following features extract the significant information from handprinting, and those of Raviv [23] and Duda and Hart [24] that using the measurements of neighbouring characters together with contextual constraints greatly improves the recognition accuracy.

REFERENCES

1. W.W. Bledsoe and J. Browning, "Pattern recognition and reading by machine," Pattern Recognition, L. Uhr., Ed., pp. 301-316, Wiley, New York, 1966.

2. L. Uhr and C. Vossler, "A pattern-recognition program that generates, evaluates and adjusts its own operators," Computers and Thought, Feigenbaum and Feldman, Eds., pp. 251-268, McGraw-Hill, New York, 1963.

3. W.H. Highleyman, "Linear decision functions with application to pattern recognition," Proc. IRE, Vol. 50, No. 6, pp. 1501-1514, June, 1962.

4. L.A. Kamentsky, "The simulation of three machines which read rows of hand-written arabic numbers," IRE Trans. on Electronic Computers, Vol. EC-10, No. 3, pp. 489-501, September, 1961.

5. R. Bakis, N.M. Herbst, and G. Nagy, "An experimental study of machine recognition of handprinted numerals," IEEE Trans. on Systems Science and Cybernetics, Vol. SSC-4, pp. 119-132, July, 1968.

6. E.C. Greanias, et al., "The recognition of handwritten numerals by contour analysis," IBM Journal, Vol. 7, No. 1, pp. 14-21, January, 1963.

7. L.G. Roberts, "Pattern recognition with an adaptive network," IRE 1960 International Convention record, pp. 66-70.

8. W.H. Highleyman, "An analog method for character recognition," IRE Trans. on Electronic Computers, Vol. EC-10, pp. 502-512, September, 1961.

9. R.L. Grimsdale, et al., "A system for the automatic recognition of patterns," Proc. IEE , Vol. 106B, No. 26, pp. 210-221, March, 1959.

10. J.K. Clemens, "Optical Character Recognition for Reading Machine Applications," Ph.D. thesis, Department of Electrical Engineering, M.I.T., Cambridge, Mass., August, 1965.

11. J.K. Clemens, "Optical character recognition for reading machine applications," M.I.T. Electronics Reserach Lab., Cambridge Mass., Quart. Progress Rept. 79, pp. 219-227, October, 1965.

12. M.M. Chodrow, W.A. Biovona and G.M. Walsh, "A Study of Hand-Printed haracter Recognition Techniques," Rome Air Development Center Technical Report No. RADC-TR-65-444, February, 1966.

13. J.H. Munson, "The recognition of handprinted text," Pattern Recognition, L. Kanal, Ed., Thompson Book Co., Washington, D.C., 1968.

14. D.D. Johnson, C.F. Haugh, and K.P. Li, "The application of a few hyperplane decision techniques to handprinted character recognition," Proc. NEC, Vol. 22, pp. 869-874, 1966.

15. J.H. Munson, "Experiments in the recognition of hand-printed text: Part I - Character recognition," Fall Joint Computer Conference AFIPS Proc., Vol. 33, pt. 2, pp. 1125-1138, Thompson, Washington, D.C., 1968.

16. A.L. Knoll, "Experiments with 'characteristic loci' for recognition of handprinted characters," IEEE Trans. on Electronic Computers, Vol. EC-18, pp. 366-372, April, 1969.

17. R.W. Cornew, "A statistical method of error correction," Information and Control, No. 2, February, 1968.

18. R. Alter, "Utilization of contextual constraints in automatic speech recognition," IEEE Trans. on Audio and Electroacoustics, March, 1968.

19. G. Carlson, "Techniques for replacing characters that are garbled on input," AFIPS Conference Proceedings, Vol. 28, 1966 Spring Joint Computer Conf., pp. 189-192.

20. A.W. Edwards and R.L. Chambers, "Can a priori probabilities help in character recognition?" J. Assoc. Computing Machinery, Vol. 11, pp. 465-470, October, 1964.

21. K. Abend, "Compound decision procedures for pattern recognition," Proc. NEC, Vol. 22, pp. 777-780, 1966.

22. K. Abend, "Compound decision procedures for unknown distributions and for dependent states of nature," Pattern Recognition, L. Kanal, Ed., Thompson Book Co., Washington, D.C., 1968.

23. J. Raviv, "Decision making in Markov chains applied to the problem of pattern recognition," IEEE Trans. on Information Theory, Vol. IT-13, pp. 536-551, October 1967.

24. R.O. Duda and P.E. Hart, "Experiments in the recognition of handprinted text: Part II - context analysis," 1968 Fall Joint Computer Conf. AFIPS Proc. Vol. 33, pt. 2, Washington, D.C.: Thompson, 1968, pp. 1139-1149.

25. M.D. Levine, "Feature extraction: A survey," Proc. of IEEE, Vol. 57, No. 8, pp. 1391-1407, August, 1969.

26. S.J. Mason and J.K. Clemens, "Character recognition in an experimental reading machine for the blind," Recognizing Patterns, P.A. Kolers and M. Eden, ed., The M.I.T. Press, Cambridge, Mass., 1968, pp. 156-167.

27. G. Nagy, "State of the art in pattern recognition," Proc. IEEE, Vol. 56, pp. 836-862, May, 1968.

28. D.E. Troxel, F.F. Lee, and S.J. Mason, "Reading Machine for the Blind," Quarterly Progress Report No. 89, Research Laboratory of Electronics, M.I.T., April 15, 1968, pp. 245-248.

29. D.E. Troxel, "Page reader for a reading machine for the blind," Quarterly Progress Report No. 94, Research Laboratory of Electronics, M.I.T., July 15, 1969, pp. 233-246.

30. F. Attneave and M.D. Arnoult, "The quantitative study of shape and pattern perception," Pattern Recognition, L. Uhr, Ed., pp. 123-141, Wiley, New York, 1966.

31. J.A. Bradshaw, "Letter recognition using captive scan," IEEE Trans. on Electronic Computers, EC-12, p. 26, February, 1963.

32. A.G. Semenovskiy, "Recognition of handwritten characters by means of a servo scan," Character Readers and Pattern Recognition, V.A. Kovalevsky Ed., Spartan Books, New York, 1968, pp. 213-222.

33. G.M. Austin, "Design and Construction of an Opaque Optical Contour Tracer for Character Recognition Research," M.A.Sc. Thesis, University of British Columbia, December, 1967.

34. K.S. Fu, "Sequential Methods in Pattern Recognition and Machine Learning," Academic Press, New York and London, 1968.

35. N.J. Nilsson, "Learning Machines: Foundations of Trainable Pattern-classifying Systems," McGraw-Hill, 1965.

36. E. Fix, and J.L. Hodges, Jr., "Discriminatory Analysis, Nonparametric Discrimination: Consistency Properties," Project 21-49-004, USAF School of Aviation Medicine, Randolph Field, Texas, February, 1951.

37. C.K. Chow, "Statistical independence and threshold function," IEEE Trans. Electronic Computers, Vol. EC-14, pp. 66-68, February, 1965.

38. G.S. Sebestyen, "Decision-making processes in pattern recognition," The Macmillan Company, New York, 1962

39. J.M. Wozencraft and I.M. Jacobs, "Principles of Communication Engineering," John Wiley & Sons, Inc., New York, 1965, pp. 459-460.

40. L. Kanal and B. Chandrasekaran, "On dimensionality and sample size is statistical pattern classification", National Electronics Conference, Chicago, 1968, pp. 2-7.

41. W.H. Highleyman, "The design and analysis of pattern recognition experiments," The Bell System Technical Journal, March, 1962, pp. 723-744.

42. P.A. Lachenbruch and M.R. Mickey, "Estimation of Error Rates in Discriminant Analysis," Technometrics, Vol. 10, No. 1, February, 1968, pp. 1-11.

43. W.G. Cochran, "Commentary on "Estimation of Error Rates in Discriminant Analysis," Technometrics, Vol. 10, No. 1, February, 1968, pp. 204-205.

44. J.R. Pierce, "Symbols, Signals and Noise: The nature and process of communication," Harper & Row, New York, 1961, p. 283.

45. F. Pratt, "Secret and Urgent, the story of codes and ciphers," Blue Ribbon Books, Garden City, New York, 1942.

46. G.T. Toussaint and R.W. Donaldson, "Algorithms for recognizing contour-traced handprinted characters," IEEE Transactions on Computers, in press.

47. G.F. Hughes, "On the mean accuracy of statistical pattern recognizers," IEEE Trans. on Information Theory, Vol. IT-14, No. 1, January 1968, pp. 55-63.