Department of Computer Sciences
University of Wisconsin
Madison, Wisconsin

FEATURE EXTRACTION ALGORITHMS

by

S. Kamal Abdali

Technical Report #87

March 1970

ABSTRACT

This paper describes methods for extracting pattern-synthesizing features. A set of patterns is expressed as a Boolean matrix, allowing the problem of feature extraction to be viewed as one of factoring this matrix. Feature extraction methods based on matrix factorization and pattern intersection are presented. Attribute inclusion is defined to be the implication of the presence of one attribute by that of another, and an algorithm for obtaining features correlated by inclusion is discussed.

1. INTRODUCTION

The description of patterns in terms of the raw measurements obtained by a transducer does not, in general, yield to simple recognition schemes, because the number of measurements is large, and/or the classes are not linearly separable. Encoding the patterns in terms of constituent features offers a more economical description, and also tends to reduce the non-linearities in the separation of pattern classes. As a result, most character recognition systems employ feature detection to some extent. Since high dimensionality and non-linearities account for most of the difficulties in pattern recognition, feature extraction can be seen to be a central problem of this field. But, as pointed out by Munson[12], feature extraction techniques are poorly developed in comparison with classification methods. Most work done on this problem is of an *ad hoc* nature, and a reasonable theory is yet to be developed.

The word *feature* seems to have been used in the literature in at least three related, but by no means identical, connotations: specialized measurements, visual features, and pattern-synthesizing features. In the first meaning, features are considered to be some particular sets of outstanding measurements derived from the initial pattern measurements--outstanding in the sense that they more strongly characterize and discriminate classes than the raw measurements obtained by the receptor. Although such features may turn out to be similar to the visual features described below, this is not obvious from the way they are selected. The selection methods are based on statistical and information-theoretical measures of the discriminating ability of these features.

Thus, each measurement in a pattern is allotted a "goodness" coefficient by Lewis[8], so that a best set can be chosen. The goodness of a measurement takes into account the probabilities with which it occurs: its overall a priori probability, and its joint and conditional probabilities with all classes. An information measure for the relative merit of an n-tuple of measurements (for apriori equally probable classes) has been given by Kamentsky and Liu[7]. This is based on the criterion that if the conditional probabilities of different classes with respect to an n-tuple widely differ among themselves, then it is a better feature than the one for which they are almost equal. Another similar measure has been derived for the case of linear decision and uncorrelated measurements by Bakis et al.[2]. Tests for best feature sets based on maximizing average class divergence are given by Chen[5]. Correlation of the pattern measurements with suitable filtering functions is used to derive features by Meyer and Giuliano[10]. A mathematical theory of features, based on functional transformation of patterns, has been proposed by Iijima[9]. A quasi-normal pattern, derived from the pattern image by means of certain transformations, is expanded as a system of hyper-orthogonal functions. The terms of this expansion are regarded as features.

Features in the second meaning are used to denote visual, i.e. topological and geometrical characteristics of patterns; for example: closed curves, forks, corners, straight segments, bays, etc. Such features are usually intuitively chosen, and elaborate techniques are used to detect them in patterns. For

example, character edge slope computation is used by Sublette[15], and multiple correlation with certain functions are used to detect these by Clowes[6]. Muchnik has given an algorithm to extract significant visual features automatically by explorations for the local minima of certain functions of pattern measurements[11].

In the third sense, features are considered to be subpatterns whose superposition results in a given pattern. Feature selection is then equivalent to determining a minimal set of features which are sufficient to synthesize all patterns.

Further conditions have been imposed by Block et al.[3]: a feature must be the intersection of all patterns containing it, and a pattern, the union of all features that it contains. Two algorithms are proposed for feature extraction under those conditions. The algorithm convergence as well as good results are dependent on certain assumptions regarding feature sizes.

"Imperfect" features with possible errors in pattern reconstruction (taken into account as a figure of merit) have been considered by Nagy[13]. From the pattern vectors in the measurement space, measurement vectors in the pattern space are constructed. The distances are small between those measurements (now treated as points in pattern space), which tend to be present or absent together in many patterns. Since features are precisely comprised of such measurements, they are sought as (possibly non-disjoint) clusters of measurement points. Ternary features with present, absent or indifferent measurements are also considered.

Pattern-synthezing features do not seem to have been applied to character recognition. This paper attempts to study automatic extraction of such features, for they might perform better as decision parameters than the ones selected arbitrarily. Moreover, these may be especially useful in the recognition of non-visual and multidimensional patterns, where intuition is of little help.

We represent a set of patterns by a Boolean matrix, and find that the feature extraction problem is equivalent to that of factorizing this matrix. A duality is seen to exist between patterns and attributes, so that features can be found by an analysis of attributes as well as of patterns. We discuss several feature extraction methods. Next, we define attribute inclusion as the relation that the presence of one attribute implies that of another, and seek features which group together attributes correlated by inclusion. This additional constraint on features leads to a feature extraction algorithm which, though not optimal, is quite simple and rapid.

2. PRELIMINARY CONSIDERATIONS

Patterns, Attributes and Features

The patterns considered here are represented by a set of measurements derived by a transducer. Each of these measurements is considered to be the value of a certain attribute. We restrict ourselves to 2-valued attributes, with values 1 and 0 (corresponding to black and white units of a visual pattern). A pattern can thus be represented by a pattern vector of dimensionality equal to the number of its attributes, and having attribute values for its components. For example the pattern P_1 of Fig. 1 is represented by the 25-dimensional attribute space vector

$$(1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,0,0,0,0,0,0,0,0,0,0).$$

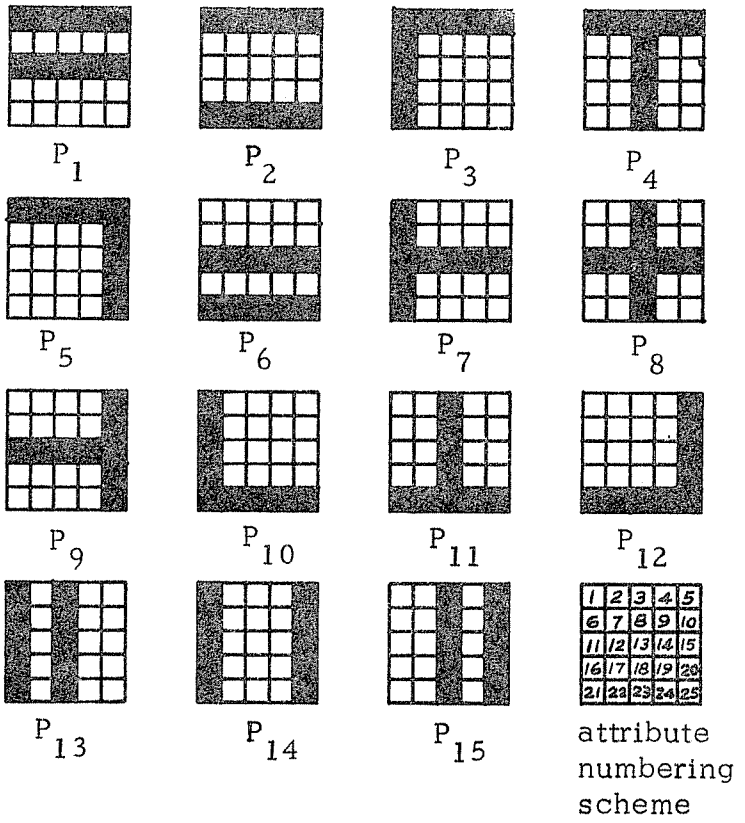
The transducer collects whatever information is considered a priori relevant to characterize a pattern. This information usually contains substantial redundancies. The attributes are interrelated and may not be equally significant in characterizing patterns. It may be possible to construct simpler patterns, called features, in the same attribute space, and to express the original patterns as superposition of features. Fig. 1 illustrates this point. The feature F_1 can be represented by the feature vector

$$(1,1,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0).$$

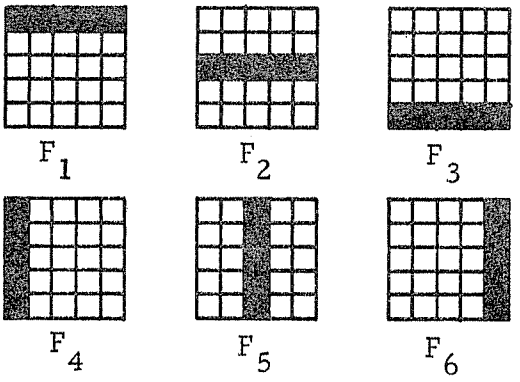
It is clear that each pattern is formed by the superposition of two of the six shown features. Thus, the patterns P_1 and P_2 can be represented, respectively by the reduced pattern vectors

$$(1,1,0,0,0,0) \text{ and } (1,0,1,0,0,0)$$

in the 6-dimensional feature space.



(a) Patterns



(b) Features

Fig. 1 Patterns and features.

Note: The above figures are from[3].

A pattern vector contains the value of all attributes in a given pattern. We are often interested, however, in knowing the values of a particular attribute in all patterns (of a given set). This information is given by the attribute vector. For example, the attribute A_1, corresponding to the unit 1 of the patterns in Fig. 1 is represented by the vector

$$(1,1,1,1,1,0,1,0,0,1,0,0,1,1,0)$$

in the 15-dimensional pattern space.

Definitions and Notations

In order to formalize the above concepts, we need some definitions and notational conventions. The vectors and matrices considered here are all comprised of Boolean elements, i.e. those admitting values 1 and 0 only. We denote by a+b and ab the Boolean inclusive sum and product of a and b. We define $a < b$ when $a=0$ and $b=1$, and $a \leq b$ whenever $a=b$ or $a<b$. The above operations and relations are defined for vectors on element-by element basis. Elements, vectors and matrices are denoted by lower case, upper case and script capital letters, respectively. An element of a vector or matrix is denoted by the corresponding lower case letter, suffixed to show its position; e.g., a_n and b_{mn} are the n^{th} and $(m,n)^{th}$ components of vector A and matrix $\mathcal{B}$, respectively. The n^{th} column of matrix $\mathcal{B}$ is denoted B_n. A set of vectors $\{A_1, A_2, \ldots, A_N\}$ is also considered a matrix, with A_n for its n^{th} column, and vice versa; this set and the matrix are both denoted by the same symbol $\mathcal{A}$.

Let $\mathcal{A},\mathcal{B}$ be MxN and NxK matrices. Then the transpose $\mathcal{A}^t$ is the NxM matrix $\mathcal{C}$ such that $c_{nm} = a_{mn}$. The product $\mathcal{AB}$ is the MxK matrix $\mathcal{D}$ such that $d_{mk} = \sum_{n=1}^{N} a_{mn} b_{nk}$. Clearly, if $\mathcal{D} = \mathcal{AB}$, then $\mathcal{D}^t = \mathcal{B}^t\mathcal{A}^t$. For a vector A, the modulus $\|A\|$ is the number of components of A with value 1. The null vector O and identity vector I are vectors of any dimension with 0 and 1, respectively, for components. The unit vector U_m is of any dimension $\geq m$ with 1 for m^{th} component and 0 for all others.

Patterns are represented by pattern vectors, and a set of these vectors is a pattern set. A set of vectors $\mathcal{F} = \{F_1, F_2, \ldots, F_K\}$ is a feature set for a set of vectors $\mathcal{P} = \{P_1, P_2, \ldots, P_M\}$, if for every $m \leq M$, there exists a set $k(m) \subseteq \{1,2,\ldots,K\}$ such that $P_m = \sum_{k \in k(m)} F_k$. Members of $\mathcal{F}$ are called feature vectors of $\mathcal{P}$.

For a set of N-dimensional pattern vectors $\mathcal{P} = \{P_1, P_2, \ldots, P_M\}$, the attribute set is the M-dimensional vector set $\mathcal{A} = \{A_1, A_2, \ldots, A_N\}$ such that $U_m = A_n$ if and only if $U_n = P_m$, for $m \leq M$ and $n \leq N$. Members of $\mathcal{A}$ are attribute vectors of $\mathcal{P}$.

According to an above stated convention, sets $\mathcal{P}$, $\mathcal{F}$ and $\mathcal{A}$ are also treated as matrices, with their respective members as columns, and are called, respectively, pattern, feature and attribute matrix.

We have not defined features uniquely. A pattern set $\mathcal{P}$ is a feature set for itself. Another feature set is $\mathcal{U}$ which consists of unit vectors for each

attribute. Other trivial feature sets can be obtained by adding arbitrary features to $\mathcal{P}$ or $\mathcal{U}$. It is not of much interest to find arbitrarily large feature sets. We are mainly concerned with finding small feature sets for an economical description of patterns, and the number of features should certainly not exceed the smaller of the numbers of patterns and attributes; otherwise, $\mathcal{P}$ or $\mathcal{U}$ would be good enough.

Two properties, independence and nonredundance are desirable in a feature set. Independence ensures that each pattern can be reconstructed by a unique combination of features. (That is, the set k(m) mentioned in the definition of features is unique for each m.) Nonredundance means that each feature is eventually utilized. Formally, a set $\mathfrak{F} = \{F_1, F_2, \ldots, F_K\}$ of vectors is independent if for any subsets s and t of $\{1,2,\ldots,K\}$, $\sum_{k \in s} F_k = \sum_{k \in t} F_k$ implies s = t. A feature set $\mathfrak{F}$ of a given pattern set $\mathcal{P}$ is nonredundant if no proper subset of $\mathfrak{F}$ is a feature set for $\mathcal{P}$.

Matrix Formulation of Feature Extraction Problem

A basic property of features is given by the following

THEOREM: *$\mathcal{P}, \mathfrak{F}$ being sets of vectors, a necessary and sufficient condition for $\mathfrak{F}$ to be a feature set for $\mathcal{P}$ is that there exist a set of vectors $\mathcal{R}$ such that $\mathcal{P} = \mathfrak{F}\mathcal{R}$.*

(We again regard a matrix equivalent to the set of its column vectors.) The reader is referred to[1] for proofs of results stated here.

$\mathcal{R}$ is the set of reduced pattern vectors in the feature space. It is possible to obtain different sets $\mathcal{R}$ from the same $\mathcal{P}$ and $\mathfrak{F}$. However, $\mathcal{R}$ is characterized uniquely if the feature set $\mathfrak{F}$ is independent. This is stated in the

THEOREM: If $\mathcal{P}$, $\mathfrak{F}$ and $\mathcal{R}$ are sets of vector, $\mathcal{P} = \mathfrak{F}\mathcal{R}$, and $\mathfrak{F}$ is independent then $F_k \leq P_m$ if and only if $r_{km} = 1$.

This result furnishes a simple method for computing $\mathcal{R}$ from $\mathcal{P}$ and $\mathfrak{F}$, given $\mathcal{P} = \mathfrak{F}\mathcal{R}$.

An important consequence of matrix relationships is the concept of duality between patterns and attributes. From $\mathcal{P} = \mathfrak{F}\mathcal{R}$ we conclude $\mathcal{A} = \mathcal{R}^t\mathfrak{F}^t$, as $\mathcal{P}^t$ is just the attribute matrix $\mathcal{A}$. Hence, by the first theorem, $\mathcal{R}^t$ is a feature set for $\mathcal{A}$. Thus, to determine $\mathfrak{F}$ from $\mathcal{P}$ we can also set out with the attribute matrix $\mathcal{A}$, find its feature set $\mathcal{R}^t$, compute $\mathfrak{F}^t$ (using the rule given by the second theorem), and take its transpose to get $\mathfrak{F}$. We call this the indirect method. As long as only heuristic methods of feature extraction are available, this indirect route from patterns via attributes to features may possibly yield better features. Moreover, in most cases, $\mathfrak{F}$ is not an end in itself: it is used to obtain $\mathcal{R}$, the reduced pattern set. In such cases, the indirect method actually involves less computation.

A graph representation of the feature determination problem is as follows: Patterns and attributes are represented by nodes $P_1, P_2, \ldots P_M, A_1, A_2, \ldots, A_N$, and arcs $P_m A_n$ are drawn to indicate that the value of the n^{th} attribute in the m^{th} pattern is 1. The problem is to place intermediate nodes $F_1, F_2, \ldots, F_K$,

corresponding to features, and draw new arcs from P's to F's and F's to A's in such a way that A_n be reachable from P_m through some F_k if, and only if, there exists an arc $P_m A_n$ in the initial graph.

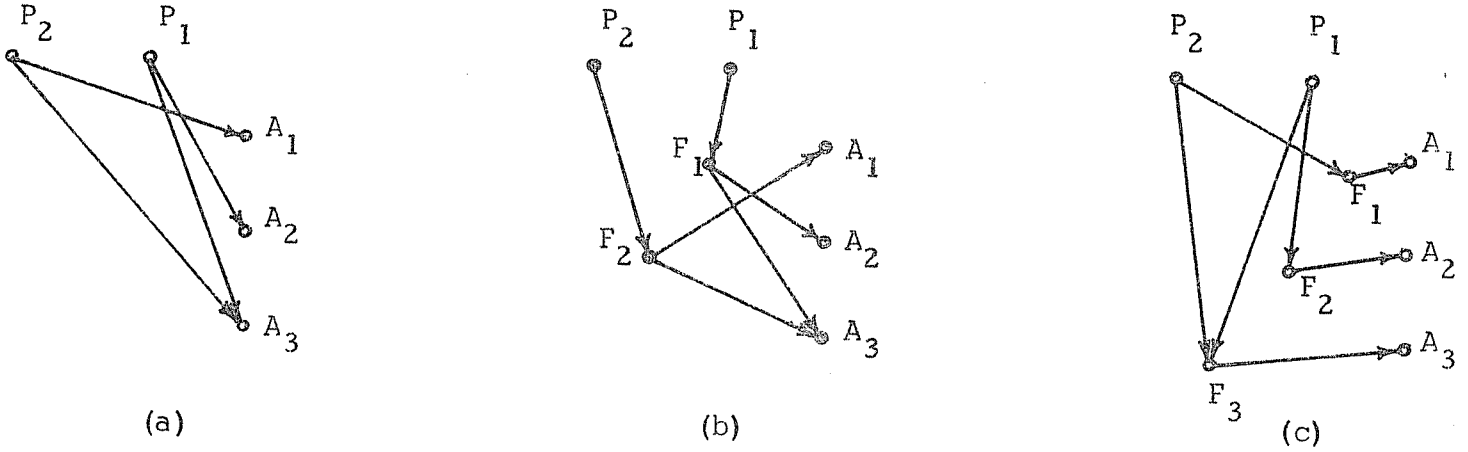


Fig. 2 Graph representation of patterns and features.

Fig. 2 illustrates a simple case: (a) shows two patterns $P_1 = (0,1,1)$ and $P_2 = (1,0,1)$; (b) and (c) present the two trivial solutions mentioned previously.

3. SOME FEATURE EXTRACTION METHODS

Boolean Matrix Factorization

We have seen that the problem of determining features is equivalent to that of factorizing the pattern matrix. Although this result is interesting conceptually, it does not offer a solution of the problem as such, since suitable algorithms for factorizing a B-matrix are not available yet.

A brute force approach is to attempt to solve the system of Boolean equations:

$$p_{nm} = \sum_{k=1}^{K} f_{nk}\, r_{km} \tag{1}$$

We can try different values of K until the above set of MN equations with $K(M + N)$ unknowns is consistent. Then all solutions can be found. Of course, for $K \geq \min(M,N)$ the system is always consistent.

The available methods for solving the system (1) become practically impossible to use for the dimensionality we consider. For example a classical method[4] requires use of 2 matrices of orders $K(M+N) \times 2^{K(M+N)}$ and $MN \times 2^{MN}$ to form another matrix of order $2^{K(M+N)} \times 2^{MN}$, in order to solve the system. For $M = N = 50$, $K = 8$, the typical values for a modest size problem, these matrices are of the orders 2500×2^{2500}, 800×2^{800} and $2^{2500} \times 2^{800}$! Of course, this method is very general, and considerable economies may be possible with short cuts to tailor it to the special type of equations in (1).

A similar problem is encounted in trying to use another method for solving Boolean equation[16]. Here, the size of rectangular maps of Boolean functions becomes too large to make this method practical for our use.

Modulo 2 Factorization

Using exculsive sum $(\overset{\oplus}{\Sigma})$ rather than inclusive sum (+), we can rephrase some of our previous definitions to obtain a different system of operation. In this system, if $\mathfrak{F}$ is a feature set for $\mathfrak{P}$, then

$$P_m = \overset{\oplus}{\underset{k \in k(m)}{\Sigma}} F_k, \text{ for some } k(m) \subseteq \{1,2,\ldots K\}, \text{ and,}$$

$$\mathfrak{P} = \mathfrak{F}\mathfrak{R}$$

i.e. $$P_{nm} = \overset{K}{\underset{k=1}{\overset{\oplus}{\Sigma}}} f_{nk} \cdot r_{km} .$$

The system $(\{0,1\},\oplus,0,1)$ is the field "modulo 2," and the vectors over this field form a vector space. Hence, the feature set can simply be taken to be the basis for the pattern set over this vector space.

Since $1 \oplus 1 = 0$, like attributes in two features cancel each other, and thus, patterns are not obtained any more as a simple superimposition of features. The "modulo 2," features, therefore, have no direct visual interpretation.

Classical methods of linear algebra are available for determining the basis, as well as for factorizing the pattern matrix. Fig. 3 shows a basis consisting of 13 vectors for the 15 patterns of Fig. 1. This basis was obtained by transforming the set of pattern vectors in row form to its equivalent row-reduced echelon form.

In spite of the fact that no visual interpretation can be attached to the "modulo 2" features, there can still be useful as a mathematical transformation of a set of patterns. However, the main objection to their use comes from the observation

P_1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
P_2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
P_3	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0
P_4	1	1	1	1	1	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0
P_5	1	1	1	1	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1
P_6	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1
P_7	1	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0
P_8	0	0	1	0	0	0	0	1	0	0	1	1	1	1	1	0	0	1	0	0	0	0	1	0	0
P_9	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	0	0	0	0	1
P_{10}	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1	1	1	1
P_{11}	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	0	0	1	1	1	1	1
P_{12}	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	1	1	1	1	1
P_{13}	1	0	1	0	0	1	0	1	0	0	1	0	1	0	0	1	0	1	0	0	1	0	1	0	0
P_{14}	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1	1	0	0	0	1
P_{15}	0	0	1	0	1	0	0	1	0	1	0	0	1	0	1	0	0	1	0	1	0	0	1	0	1

(a) Pattern Vectors

F_1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
F_2	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
F_3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
F_4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
F_5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	1	1	
F_6	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	1
F_7	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1
F_8	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
F_9	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0	1	0
F_{10}	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1
F_{11}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
F_{12}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
F_{13}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

(b) Feature Vectors

Fig. 3 Modulo 2 Features

that the number of such features is rather large. It is generally the case that the number of features needed to reconstruct the patterns using exclusive or is larger than that needed with exculsive or.

Pattern Intersection Method

A feature which is included in each of two patterns is, evidently, also included in their intersection. As a result, the patterns which share a common feature can be expected to have larger intersections than the patterns that do not share any feature. This is certainly true for the pattern and feature sets shown in Fig. 1. (Here, each intersection of patterns having common features contains at least 5 black points, while the intersections of patterns without common features have at most 4 black points.) In case the above supposition is true for any given pattern and feature sets, two patterns share a common feature if and only if their intersection is larger than a certain threshold. This property has been called the threshold condition by Block et al.[3]

If the threshold condition holds for a given pattern set, then a feature set can, in principle, be obtained in the following way. First, all possible intersections of the given patterns are generated. Then, from among these intersections, the ones whose size is smaller than the threshold, as well as all repetitions of the same intersection, are eliminated. The remaining intersections constitute a feature set.

The above procedure is obviously impracticable, since it would require the generation and examination of

$$\sum_{m=2}^{M} \binom{M}{m} \simeq 2^M$$

intersections.

Block et al. have presented two feature determination algorithms, which make use of a threshold-raising scheme to avoid examining all patterns intersections [3]. An algorithm will be proposed here, which does not employ threshold-raising, and attempts to search through all pattern intersections for potential features.

Suppose $P_1, P_2, \ldots, P_M$ are the given patterns satisfying the threshold condition, and let θ be the threshold. Let Q be a $M \times M$ matrix, whose initial entries $Q_{ij}(0)$ are given by

$$Q_{ij}(0) = P_i \cdot P_j, \quad (1 \leq i \leq M;\ 1 \leq j \leq M).$$

Then, the value of Q_{ij} at step k, $Q_{ij}(k)$, is given by

$$Q_{ij}(k) = Q_{ik}(k-1) \cdot Q_{kj}(k-1), \text{ if } \| Q_{ik}(k-1) \cdot Q_{kj}(k-1) \| \geq \theta,$$

$$= Q_{ij}(k-1), \text{ otherwise.}$$

$$(1 \leq i \leq M;\ 1 \leq j \leq M).$$

The above operations are performed in M steps, $k = 1, 2, \ldots, M$.

Once the M steps have been performed, the entries of the final eatrix $Q(M)$ are sorted in the order of frequency of occurrence, and the more frequent ones are selected as features.

The similarity of this method to the generalized Warshall algorithm is apparent [14]. Unfortunately, however, the operations used on Q_{ij}'s do not

satisfy the requirement of a "Q-semiring", so that it is not guaranteed that all possible pattern intersections are taken into account. The selection of features on the basis of the frequency of pattern intersections is, moreover, quite arbitrary. Nevertheless, the algorithm provides useful results when the threshold condition is satisfied, and its implementation as a computer program is very simple.

The number of operations in this algorithm is proportional to M^3, rather than to the 2^M operations, as required of an exhausive search through all pattern intersections. Further, since the initial matrix Q, as well as all subsequent operations, are symmetrical, it is sufficient to work with only a triangular half of the matrix.

The threshold and the number of features chosen are arbitrary decided. Had the threshold θ been known, features could have been determined from those final entries which are larger than θ, by simply eliminating repetitions. However, θ is not known, so that the frequency count is arbitrarily used for obtaining features. In practice, it is found advantageous to repeat the procedure for several values of the threshold, and check if it is possible to discriminate features from other intersections on the basis of prominent frequency differences.

The results obtained from the application of this algorithm to the patterns of Fig. 1 are summarized in Table 1. This table lists the frequency of occurrence of the 6 features of Fig. 1 in the final matrix Q, for the values 1 to 7 of the threshold. The results for $\theta \geq 7$ are identical to those for $\theta = 6,7$. The labels

"first" and "seventh" indicate, respecitvely, the most frequent intersection, and the one next after $F_1,\ldots,F_6$ in the order of frequency. The most frequent intersections for $\theta \geq 3$ is one of the features $F_1,\ldots,F_6$.

Intersection \ Threshold	1	2	3	4	5	6	7
F_1	0	0	4	10	10	4	4
F_2	0	0	4	10	10	4	4
F_3	0	0	4	10	10	4	4
F_4	1	0	7	10	10	4	4
F_5	0	0	7	10	10	4	4
F_6	1	1	7	10	10	4	4
First	36	15	7	10	10	4	4
Seventh			2	1	1	1	1

Table 1 Frequencies of final pattern intersections for the pattern given in Fig. 1.

In the case of Table 1, it is seen that the number of features to be selected is decided easily on the basis of changes in frequency of occurrence of pattern intersections, and that the results are stable over reasonably wide threshold variations.

For patterns not satisfying the threshold condition, the results obtained from the above algorithm are usually poor: The changes in the frequency of intersections are gradual which makes the choice of features difficult, and the features obtained are insufficient for pattern reconstruction.

4. ATTRIBUTE INCLUSION ALGORITHM

Some Restrictions on Features

In the previous discussion, our emphasis has been mainly on features being sufficient for pattern reconstructions. Although, this property of a feature set is important, the intuitive connotation of features includes additional desirable properties. Features, in the sense of traits, or syndromes, should bring out the interrelation of attributes found in a given set of patterns. In other words, one aim of feature extraction is to detect the presence of regularities or laws in a non-random attribute value distribution, and to express them in individual features.

The interrelation of attributes may reveal itself in several ways. A simple case is that of a constant valued attribute in all or most patterns. In some cases, several attributes may have similar value in many patterns. Such relations are best displayed by the pattern matrix. We consider, for example the patterns for Fig. 1. The pattern matrix, with some of the rows permuted or omitted to emphasize these relationships, is shown in Fig. 4. It is observed that the attribute pairs (2,4), (6,16) are equal valued, and the attributes 7, 9, 17, 19 are always zero. As a result, we expect that none of the features should include attributes 7, 9, 17, 19 with the value 1, and that at least one feature should contain each of the related pairs (2,4), and (6,16).

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}
A_1	1	1	1	1	1	0	1	0	0	1	0	0	1	1	0
A_2	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
A_4	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
A_6	0	0	1	0	0	0	1	0	0	1	0	0	1	1	0
A_{16}	0	0	1	0	0	0	1	0	0	1	0	0	1	1	0
A_7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_{12}	1	0	0	0	0	1	1	1	1	0	0	0	0	0	0
A_{15}	1	0	0	0	1	1	1	1	1	0	0	1	0	1	1
A_{17}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A_{19}	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 4 Pattern matrix for the patterns of Fig. 1

(Some rows have been omitted or interchanged).

A more general type of relation among attributes, embracing equality as a special case, is the dependence of an attribute on another: two attributes may be so related that, in whichever pattern the first one is present (i.e., has the value 1), so is the second one. (For example, attribute pairs (2,1), and (12,15) in Fig. 4). In such a case, it is reasonable to require that any feature containing the first attribute should also contain the second. Moreover, it is evident that we need at least one feature containing both of these attributes.

The dependence of two attributes is expressed by the relation of inclusion ($\leq$) between the corresponding attribute vectors. Thus, $\mathfrak{F}$ being the feature set for a pattern set $\mathcal{P}$ and attribute set $\mathcal{A}$, we can state the above requirement as

$$\text{(i)} \quad \text{If } 0 \neq A_i \leq A_j \text{ then } \exists F \in \mathfrak{F},\ U_i + U_j \leq F \ .$$

It is primarily because of this property sought in features, that the feature extraction algorithm, presented later in this section, is different from the existing methods.

If we insist on perfect reconstructibility of patterns from features, then we are in need of the provision that every attribute with value 1 in any pattern should also have the value 1 in a feature which falls in that pattern. We express this relation as:

$$\text{(ii)} \quad \text{if } U_i \leq P \in \mathcal{P} \text{ then } \exists F \in \mathfrak{F} \ni U_i \leq F \leq P$$

Conditions (i) and (ii) seem to be a reasonably good description of what one may expect from features.

However, we cannot overlook the importance of economy of description, so that the number of features should be as small as possible. We add another condition consequently:

(iii) K is a minimum.

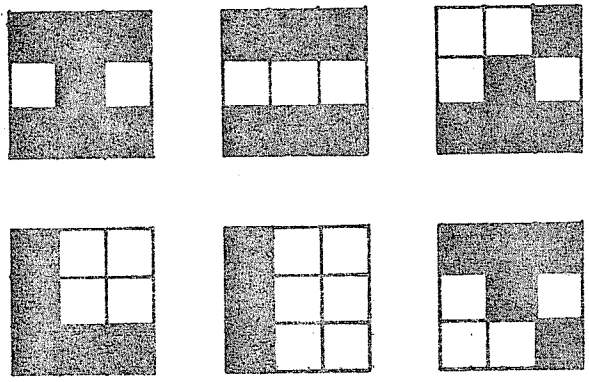
Here K is the number of features.

Unfortunately, these conditions are not altogether consistent. Thus, (ii) and (iii) together have the upper bound min (M,N) for K, where M,N are the numbers of patterns and attributes, respectively. In some cases it turns out, however, that we need $K > \min(M,N)$ in order to satisfy (i) as well.

Attribute Inclusion Method

Condition (i) of the previous section shows that any two attributes, whose corresponding vectors satisfy inclusion, belong together to a feature. Since inclusion is transitive, we can go farther, and assemble all attributes, which are related by successive inclusions, into a single feature. Thus the above condition, in effect, suggests the following algorithm for obtaining features from a given pattern set:

Test all non-zero attribute vectors for the ordering relation of inclusion, and, from these vectors, form the largest (possibly non-disjoint) partially ordered subsets which contain a single minimal vector. Then, the attributes corresponding to the vectors of each subset constitute a feature.



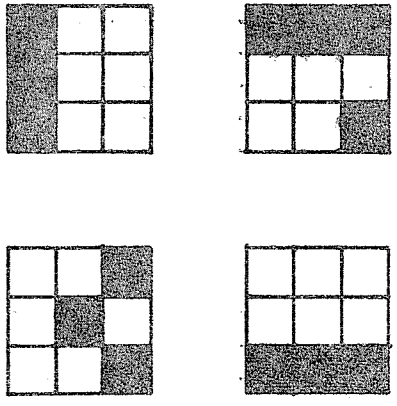
(a) Patterns

	P_1	P_2	P_3	P_4	P_5	P_6	$\|A\|$
A_1	1	1	0	1	1	1	5
A_2	1	1	0	0	0	1	3
A_3	1	1	1	0	0	1	4
A_4	0	0	0	1	1	0	2
A_5	1	0	1	0	0	1	3
A_6	0	0	0	0	0	0	0
A_7	1	1	1	1	1	0	5
A_8	1	1	1	1	0	0	4
A_9	1	1	1	1	0	1	5

(b) Pattern matrix

	4	2	5	3	8	1	7	9
4	1	0	0	0	0	1	1	0
2		1	0	1	0	1	0	1
5			1	1	0	0	0	1
d 3								
8					1	0	1	1
d 1								
d 7								
d 9								

(c) Attribute inclusion matrix



(d) Features

Fig. 5 An example of attribute inclusion method

We now present an example to illustrate the above method of feature detection. Six patterns and the corresponding pattern matrix are shown in Fig. 5 (a,b). Fig. 5 (c) shows the "attribute inclusion matrix," in which all rows and columns are labeled with attribute numbers. An entry (i,j) of this matrix is 1, if $A_i \leq A_j$, and 0, otherwise.

To test the inclusion of attributes in an efficient way, we consider the attributes in the ascending order of the moduli of the corresponding attribute vectors; thus, a pair of attributes is tested only once, in the order in which inclusion may possibly hold. Consequently, all partially ordered subsets are found from their "smallest" attribute vectors only, which ensures that the subsets are the largest possible. For example, the attribute 4 is found to be included in the attribute 1. Hence, all successors of 1 are necessarily those of 4; accordingly, we put a mark d (dependent) against the row for 1, and do not consider it for further inclusion tests. At the end, the attribute inclusion matrix shows that the attribute vector subsets (A_4, A_1, A_7), (A_2, A_3, A_1, A_9), (A_5, A_3, A_9), and (A_8, A_7, A_9) are the required largest partially ordered subsets. The features are constructed, one from each subset, by setting corresponding attributes to 1. (Thus, from the first of the above subsets, we construct the feature $F_1 = U_4 + U_1 + U_7$). The four features obtained in this way are shown in Fig. 5(d). It is clear that these features are sufficient for reconstructing the original patterns.

A graph representation of the above method is as follows: We represent each non-zero attribute vector by a node, and draw directed arcs ij between each pairs of nodes i,j for which $A_i \leq A_j$. The loops at each node are omitted; also,

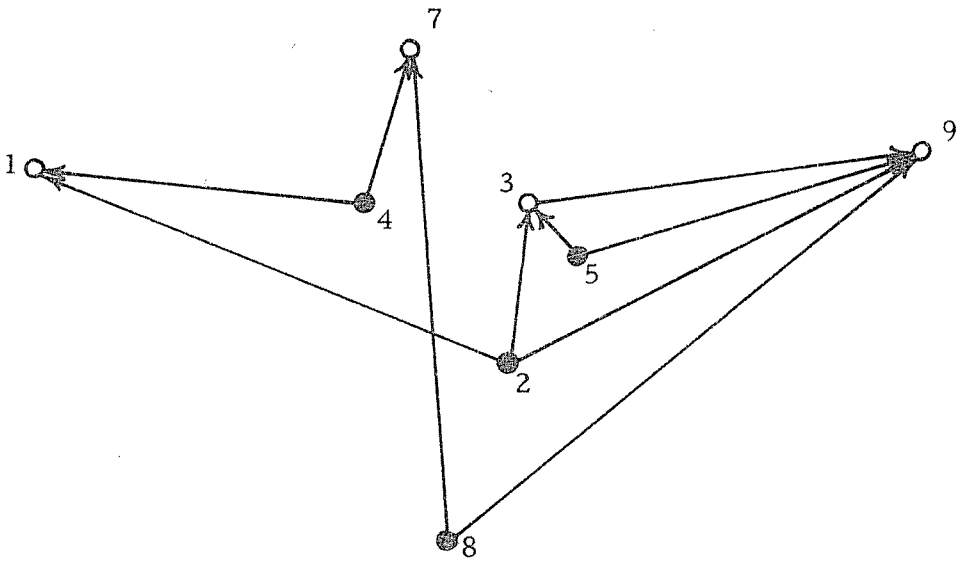


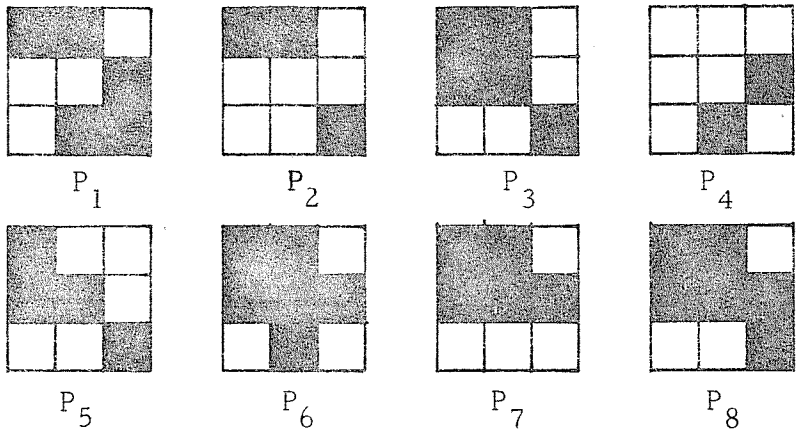
Fig. 6 Attribute inclusion graph for the patterns of Fig. 5(a).

whenever $A_i = A_j$, we draw only one arc ij, for $i < j$. We call the graph so obtained the "attribute inclusion graph." Now, each source node (i.e., one without incoming arcs) is grouped together with all nodes reachable from it. The attributes corresponding to each such node group constitute a feature. Fig. 6 shows the attribute inclusion graph for the example we considered above (Fig. 5). The source nodes are indicated in this graph by solid circles to distinguish them from others. The features obtained from this graph are the same as in Fig. 5.

Supplementary Features

The feature set obtained in the previous section contains useful features; but, in general, it is not sufficient for pattern reconstruction. For example, consider the problem shown in Fig. 7. The features $F_1, \ldots, F_4$, are obtained as a result of using the attribute inclusion method. Clearly, however, the patterns P_7, P_8 cannot be reconstructed from the features $F_1, \ldots, F_4$ only.

The reason for the insufficiency of the attribute inclusion method lies in our ignoring the dependent attributes. We investigate this in some more detail: Consider, for example the case of Fig. 7. In (c), the dependent attribute 3 is seen to be included with 2,5 in two features: (2,3,1,9), and (5,3,9). These features are included in all patterns in which the attributes 2 and 5 are present, namely, in $\{P_1, P_2, P_6\}$, and $\{P_1, P_3, P_6\}$, respectively. Now, since the attribute 3 is present in precisely the same patterns, and in no others, we do not need any other feature containing it. However, the case of the attribute 6 in Fig. 7 is different. Here, the only feature containing this attribute is (8,6,1), and although the attribute 8 (and hence, this feature) is present in $\{P_1, P_4, P_6\}$,



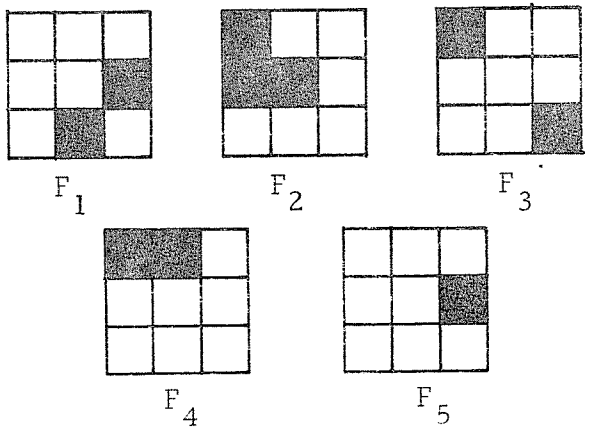
(a) Patterns

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	$\|A\|$
A_1	1	1	1	0	1	1	1	1	7
A_2	1	1	1	0	0	1	1	1	6
A_3	0	0	0	0	0	0	0	0	0
A_4	0	0	1	0	1	1	1	1	5
A_5	0	0	1	0	1	1	1	1	5
A_6	1	0	0	1	0	1	1	1	5
A_7	0	0	0	0	0	0	0	0	0
A_8	1	0	0	1	0	1	0	0	3
A_9	1	1	1	0	1	0	0	1	5

(b) Pattern matrix

	8	4	5	6	9	2	1
8	1	0	0	1	0	0	0
4		1	1	0	0	0	1
d 5							
d 6							
9					1	0	1
2						1	1
d 1							

(c) Attribute inclusion matrix



(d) Features

Fig. 7 Another example of attribute inclusion method

it is not present in P_7, P_8 which contain the attribute 6. Hence, in this particular case, we need, in addition to the above feature, another one containing the attribute 6. Now A_6 is not included in any other feature attribute vector, so that the attribute 6 by itself forms a feature, F_5.

From the above example, it would at first appear that the attribute inclusion matrix must be completed even for dependent attributes, and all rows of this matrix, (which are the partially ordered subsets of attribute vectors), be considered as potential features. As we shall see in the next section, the set of all such potential features is sufficient, (possibly, with redundance), for pattern reconstruction.

We now describe a method for the selection of non-redundant feature from among the potential features given by the rows of the attribute inclusion matrix.

As previously, the attributes are ordered in the increasing order of their moduli, ignoring the zero attribute vectors. With each row of the attribute inclusion matrix, we associate two lists, B and C: B enumerates all the patterns in which the attribute, corresponding to that row, is present. (It is, actually, a list equivalent of the attribute vector.) C is initially empty and will serve to enumerate the patterns in which the presence of a particular attribute has been taken care of by some feature. In addition to the dependence indicator d, we use two other row flags, e and s, for "equal" attributes, and "supplementary" features.

We enter 0's and 1's as previously, that is (i,j) is 1 if $A_i \leq A_j$. However, we test the attribute i for inclusion, if there is no mark against it,

		8	4	5	6	9	2	1	B	C
	8	1	0	0	1	0	0	0	1,4,6	
	4		1	1	0	0	0	1	3,5,6,7,8	
e	5								3,5,6,7,8	
s d	6				1	0	0	0	1,4,6,7,8	1,4,6
	9					1	0	1	1,2,3,5,8	
	2						1	1	1,2,3,6,7,8	
d	1								1,2,3,5,6,7,8	3,5,6,7,8;1,2;

Fig. 8 Complete solution for the patterns in Fig. 7

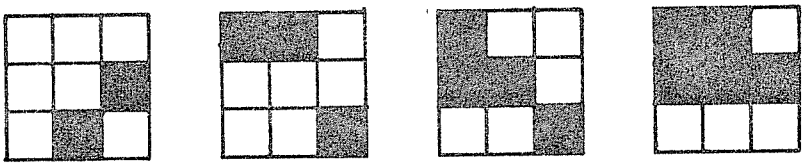


Fig. 9 A minimal feature set for the patterns in Fig. 3.4.

or in case of a mark d, only if the lists B and C do not have the same elements. In the latter situation, we mark the corresponding row with s, to show that the feature obtained from this row of dependent attributes is necessary to supplement the features obtained by the basic attribute inclusion method. While testing the attribute i for inclusion, the row j of the matrix is marked, (if it has not already been done), with e or d, in accordance to whether $A_i = A_j$, or $A_i < A_j$, respectively. In both of these cases, we add the elements of list B_i to list C_j. The features are constructed as before, after the matrix has been processed. To obtain features, we use those rows which are either not marked at all, or else contain s.

To illustrate the above procedure, the case of Fig. 7 is again worked out in Fig. 8. The resulting features are as obtained before, with the addition of F_5 (Fig. 7 (d)). The set of features $F_1,\ldots,F_5$ is easily seen to be sufficient and non-redundant for the reconstruction of $P_1,\ldots,P_8$.

The above feature set is not, however, minimal. Another feature set for the same patterns is shown in Fig. 9. This latter set is clearly sufficient, while containing only 4 features. Our algorithm will not obtain this feature set directly, for the attributes present in all individual features do not form partially ordered subsets. Specifically, one of the features contains the attributes 1,2,4,5, and 6, while inclusion does not hold in either direction between A_6 and any of A_1, A_2, A_4, and A_5. However, the above feature set can still be obtained using the present algorithm, if the indirect method of first reducing the attribute set is used.

The Algorithm

A formal description of the complete attribute inclusion algorithm more suitable for programming now follows.

Let $\mathcal{A} = \{A_1, A_2, \ldots, A_N\}$ be the attribute set of a given set of patterns. For simplicity, we assume that the attributes are numbered in such a way that

$$O \neq \|A_1\| \leq \|A_2\| \leq \ldots \leq \|A_N\|.$$

For bookkeeping purposes we employ vectors C_i, G_i and single Boolean elements d_i, s_i. (Vectors C_i are the vector equivalents of lists C used in the above discussion. Lists B are not needed as attribute vectors A_i serve in their place.) Note that U_i are unit vectors, and O is the null vector.

I. For each i=1,2,...,N do:

1. $G_i=C_i=O$

2. $d_i=0$

II. For each i=1,2,...,N do:

1. If $d_i=1$ and $C_i=A_i$, then $s_i=1$ else $s_i=0$.

2. If $d_i=0$ or else if $s_i=1$, then $G_i=U_i$.

3. For each j=i+1,i+2,...,N such that $A_i \leq A_j$ do:

a. $G_j=G_j+U_j$ b. $d_j=1$ c. $C_j=C_j+A_i$.

Then the feature set $\mathfrak{F} = \{F \mid F=G_i$ such that $d_i=0$ or else $s_i=1\}$.

The proof that this algorithm generates a complete, nonrundant feature set is given in[1], where several properties of the generated features are also derived. For example, the feature subset for which $d_i=0$ (primary features) is

independent; if the feature set consists of primary features only, then it is a minimal set. Another interesting relation between patterns and the features found by the algorithm is

$$F_k = \prod_{m \ni P_m \geq F_k} P_m,$$

a relation regarded as one of the defining properties of features by Block et al[3]. This is an interesting dual of

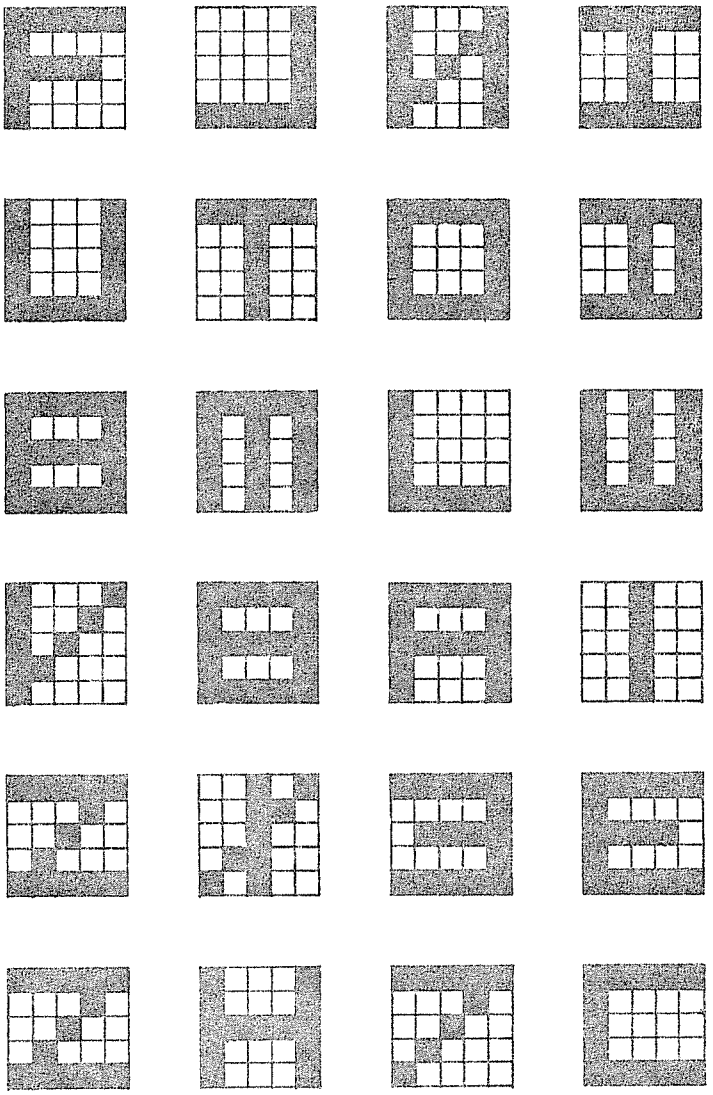
$$P_m = \sum_{k \ni F_k \leq P_m} F_k,$$

which is equivalent to the relation used in our definition of features.

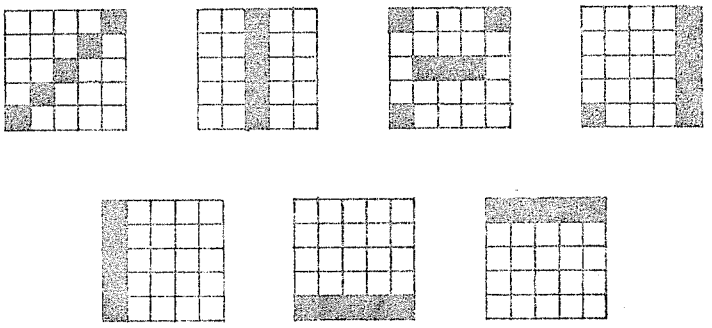
Comparison with Other Algorithms

The algorithm presented above requires a number of operations proportional to N^2, where N is the number of attributes in the given patterns. The computations involved are obviously simple and straight-forward. In fact, the algorithm is simple and rapid enough to allow problems with pattern and attribute dimensionalities of the order of up to 20-25 to be easily performed by hand, which certainly does not hold for other algorithms. Furthermore, it does not require any arbitrary parameters, such as threshholds or number of features, to be set a priori. Although the features set generated by the algorithm is nonredundant, it is not guaranteed to be minimal: in the worst case, single attributes will be produced as features.

The algorithm was used on the examples given in[3], and perfect features were obtained in each case. For example, the feature set of Fig. 1 was



(a) Patterns



(b) Features

Fig. 10 Patterns and Features (taken from (3))

found for the patterns in the same figure. An interesting case is that of the 24 patterns shown in Fig. 10. Clearly, the features in that figure form a sufficient and nonredundant feature set. The results of the application of Block et al.'s algorithm are shown in [3]. The best results are obtained with thresholds of one and two. The sets of 15 and 18 features so obtained contain the 7 desirable features along with other redundant ones. On applying the present algorithm, exactly the shown nonredundant features of Fig. 10 were obtained.

The algorithm was also applied to the non-artificial case of hand-printed characters, and interesting results were obtained. Some of this work is reported in [1].

The order in which the features are generated by the algorithm is also the order of increasing frequency of appearance of the features in the patterns. The features in the first part of the feature set are included in fewer patterns than the features in the last part. Depending on the intended use of the features, a general rule for the selection of a certain number of these from the whole set can be stated as follows: The features from the initial part are more useful for discriminating patterns, while those from the latter part are more suitable for pattern reconstruction.

For example, the attributes contained in only single patterns give rise to features, which will be quite similar to those patterns, and unlikely to be part of other patterns. Similarly, the last features are likely to consist of single attributes, common to several patterns.

Another remark concerning the minimality of feature sets is in order. As has been mentioned previously, the upper bound for the number of features in a MxN pattern set is min (M,N), where reconstructibility is the only criterion for feature formation. In the present algorithm, however, features are further constrained by the condition (i), so that their number can exceed M (in case $M < N$). For best results, both the direct and the indirect methods can be used, and the smaller of the two feature sets selected. Better features are more frequently found by treating the smaller sized vectors of the pattern-attribute matrix as attributes: this is explained easily, because more dependencies can be expected among a larger number of vectors of smaller dimension.

5. CONCLUSION

Features have been so defined as to express significant correlations between the attributes of a pattern set. These correlations are stated in terms of relationships between attribute vectors, and an algorithm has been proposed to form features by collecting the interdependent attributes. The features so obtained are sufficient for the reconstruction of patterns. It has been found that each feature is the common part of all the patterns containing it.

Although the feature set obtained by the algorithm is not, in general, minimal (with respect to the number of features), it has been observed that: (i) a feature set containing only the primary features is minimal, and (ii) of the two feature sets found by the direct and the indirect method, one often constitutes a minimal solution. Better solutions are normally obtained by using the direct method where the number of attributes is larger than that of the patterns, and vice versa.

In most practical cases, the data contain considerable noise and measurement errors, so that perfect features are not necessary to find. It is possible to obtain approximate features, by introducing tolerances in the tests for inclusion and equality between attribute vectors. For example, two patterns may be considered equal if they differ in fewer than a certain number of attributes. The tolerances can also be used in pattern reconstruction. Thus, to decide whether a certain feature is present in a pattern, a certain number of attributes present in the former and absent in the latter may be ignored. A third type of approximation is applicable to the size (modulus) of the smallest attribute vector

considered in feature construction. Attributes which are present in fewer than an arbitrary number of patterns may be ignored in computing the attribute inclusion matrix. (In the algorithm, all non-zero attribute vectors are used in the inclusion tests.)

A disadvantage of the pattern-synthesizing features lies in their sensitivity to position and orientation changes. For example, different features are needed to detect the same line segment, depending on whether it is in the left or the right part of a pattern, and whether it is horizontally oriented or vertically, etc. The tests for the inclusion of a feature should be extended to incorporate its local translations and rotations.

REFERENCES

1. S. K. Abdali, An Algorithm for Feature Extraction and Applications to Hand-Printed Character Recognition, Report of Computer Sciences Dept., University of Montreal (1968).

2. R. Bakis, N. Herbst and G. Nagy, An experimental study of machine recognition of hand-printed numerals, IEEE Trans. Sys. Sci. Cyb. SSC-4, 119 (1968).

3. H. D. Block, N. J. Nilsson and R. O. Duda, Determination and detection of features in patterns, In Computer and Information Sciences I (Edited by J. T. Tou). Spartan, Washington D. C. (1964).

4. M. Carvallo, Principles et Applications de l'Analyse Booléenne, p. 44. Gauthiers-Villars, Paris (1965).

5. C. H. Chen, A computer searching criterion for best feature set in character recognition, Proc. IEEE 53,2128 (1965).

6. M. B. Clowes, The use of multiple auto-correlation in character recognition, In Optical Character Recognition (Edited by G. L. Fisher et al.), p. 305. Spartan, Washington D. C. (1962).

7. L. A. Kamentsky and C. N. Liu, Computer automated design of multifont print recognition logic, IBM J. Res. Dev. 7, 2(1963).

8. P. M. Lewis, The characteristic selection problem in recognition systems, IRE Trans. Inf. Theo. IT-8, 171(1962).

9. T. Iijima, Basic theory of feature extraction for visual patterns, Electronics and Communications in Japan 46,292 (1963).

10. R. F. Meyer and V. E. Giuliano, Analytic approximation and translational invariance in character recognition, In Optical Character Recognition (Edited by G. L. Fisher et al.), p. 181. Spartan, Washington D. C. (1962).

11. I. B. Muchnik, Local characteristic formation algorithm for visual patterns, Automation and Remote Control 27, 1737 (1966).

12. J. H. Munson, Some views on pattern recognition methodology, 1968 Int'l. Conf. on Methodology of Pattern Recognition, Honolulu (1968).

13. G. Nagy, Feature Detection on Binary Patterns, IEEE Trans. Sys. Sci. Cyb. SSC-5, 273 (1969).

14. P. Robert and J. Ferland, Généralisation de l'algorithme de Warshall, Rev. Française Info. Rech. Opér. 2, 71 (1968).

15. T. J. Sublette and J. Tults, Character recognition by digital computers, RCA Review 23, 60 (1962).

16. A. Svoboda, An algorithm for solving Boolean equations, IEEE Trans. Elec. Comp. EC-12, 557 (1963).