

FEATURE COMPRESSION

Satosi Watanabe

*University of Hawaii
Honolulu, Hawaii*

1. THE ROLE OF "FEATURES" IN PATTERN RECOGNITION

1.1. Four Kinds of Pattern Recognition and Features

This chapter does not pretend to be a comprehensive survey of the works that have been done in the field of feature compression in pattern recognition. It is simply a short description of the way the present author looks at the problem area. Correspondingly, the references are limited mostly to the works closely related to his own project. The only exception is the materials used in Section 2, which he found useful to illustrate some of the points he wanted to make.

"Feature" seems to be one of those words whose extreme familiarity makes it so much the more difficult to define. The Webster dictionary, which is a customary resort in such a case, is not of much help, at least for our purpose, except that it provides us with a few useful synonyms: mark, quality, traits, peculiarity, characteristic, lineament. As a tentative definition, however, we may start by saying that a feature is a property or properties which make a class of objects distinguishably what they are. It is true that we speak sometimes of a feature or characteristic of an individual object instead of a class of objects, but that does not require a change in this tentative definition, since, first, the class can consist of one object, and second, we often mean by an object a collection of different aspects of an object such as different profiles of a person's head. This tentative definition al-

ready brings out the double role that has to be played by a feature: on the one hand, a feature must be a common tie binding the members of a class together, and on the other, it must be a distinctive mark separating one class from the other. The purpose of this chapter is to explicate the concept of feature so as to make it useful in pattern recognition problems, to illustrate it by a concrete example, and to describe some basic methodologies serving the purpose of extracting features from a given set of data.

“Pattern” can be any object. But we can hardly call a liter of water, or the sun, a pattern. In ordinary language, a pattern must have some shape or structure or complex properties and must be identifiable by certain distinctive marks—features. In fact, “pattern” and “feature” are sometimes even used synonymously. Being identifiable by certain features implies that a pattern is an object *qua* a member of a class. Pattern recognition thus means identification of an object with a certain set of properties as a member of some class. The set of properties can include not only global variables such as total weight and size, but also structural variables with respect to the constituent parts within the object.

The broad sense of the term pattern recognition thus defined can further be refined according to the fashion in which the classes are defined. At one extremity, there is the case where unambiguous criteria for class belonging are externally given from the beginning, so that the actual process of placing objects into classes according to their properties becomes just a matter of mechanical *sorting*. At the other extremity, we have cases where no information is available as to what the classes are to be and what the class samples are. In other words, we (the classifiers) have first to create classes on the basis of mutual similarity and dissimilarity in the given set of objects and then to place them in the classes. This is the case of *clustering*. To be rigorous, the clusters to be generated by a clustering program need not be bound together by similarity but by some more sophisticated cohesion^(1,2). In such a case, a cluster is not a class of similar objects. Between these two extremities, mechanical sorting and clustering, there are two intermediate cases. In one of them, the number of classes is given, and each class is not defined or characterized by criteria, but illustrated by a certain number of sample objects (paradigms). The task of classifier is to extract features characterizing classes from these paradigms and to place newcomers (i.e., objects with known properties and unknown class belonging) into the classes in simulation of the paradigms. This case is what is meant by *pattern recognition* in the narrow sense of the word. In another intermediate case, the classes are not enumerated, but a set of rules is given by which the class criteria can be generated. The classifier is then supposed to find out the par-

ticular combination of rules which generates the class to which an object belongs. One may speak of *generative sorting* in this case. This approach is suitable when the number of possible classes is extremely large or infinite. It is untrue that generative sorting does not need the concept of features. The particular combination of generative rules which results in a class is precisely the characteristic feature of the class which can be determined by the properties of the members.

1.2. Component and Composition—Structure Analysis

When we are faced with a pattern, our natural way of describing it seems to diverge into two directions. On the one hand, we assess or measure various variables, including two-valued predicates that are directly applicable to the object. On the other hand, we are also interested in the question as to what constituent parts the object consists of and how these constituent parts are put together. For instance, when a black-and-white photograph of a face is given, the direct-measurement approach will start with measuring the overall size, overall grayness, etc., and when required to produce the maximum precision it will probably divide the area into small meshes and measure the grayness of each mesh. The structural approach will first decompose the face into nose, mouth, eyes, forehead, cheeks, etc. and describe both the properties of each part (long nose, almond-shaped eyes, etc.) and the relations among the parts (the angle subtended by the two eyebrows, the distance between two eyes, etc.).

It is, however, a mistake to consider these two directions as mutually exclusive. The mesh in the direct-measurement approach can be considered as a component and the relative positions of the meshes are the relation among these components. Conversely, the relations in the sense of the structural approach are in some sense also measurable variables and they are certain functions, complicated as they may be, of the measured variables in the maximal description in the direct-measurement approach. After all, an object (a constituent part is an object) is a bundle of properties which is localizable, hence a localizable feature can be converted into a constituent part (see Section 2.6). The essential difference between the two approaches is that in the structural approach the most difficult part of pattern recognition is shifted to the problem of recognition of constituent components. Obviously, it is not an easy task to make a machine to recognize a nose in a portrait.

The structural approach is—it should be admitted—very natural in the human habit of recognition in daily life and in science. Stereochemistry is a paradigm case of the structural description. The atoms are components

and the three dimensional relative positions of the atoms are the structural relations. The grammatical analysis of a sentence is another example of structural description. The parts of speech are the components and the order in which they appear is the structural relation. It may be noted that the matter is considerably simplified because not all the arrangements of components are allowed by the underlying laws. In chemistry, the rule of valence and other physicochemical rules limit the possibilities. In languages, grammatical rules delimit the number of allowed strings of words. Hence, an economical way of describing the structure is to characterize each structure by the combinations of generative rules by which such a structure is made possible.

In chemistry and linguistics, the natural way of defining components is fairly clear; it is more or less dictated by the way in which the objects are produced. It is the case also with Chinese pictographs, where the components are more or less traditionally well defined, although it is an open question whether the technical task of pattern recognition should comply with these traditional definitions. But, in great many other cases, the definition of components is a matter of convenience, often arbitrary, and sometimes it is even dispensable. An amusing case is the speech recognition of vowels. It has been, and still is, customary to characterize the power spectrum of a vowel by two formants (which are components) and their relations. But I have shown that we can obtain an equally good result in recognizing vowels without paying any attention to the formants ⁽³⁾.

It may also be pointed out that the components are usually considered mutually exclusive (disjoint) and, hopefully, exhaustive, as is the case with the atoms and the alphabetic letters. But we can sometimes obtain a more flexible and useful description by allowing the components to be nondisjoint and nonexhaustive. Some components can even refer to the absence rather than the presence of parts of the object. The holes in Gruyère cheese are not constituent parts of the cheese, yet they are a good descriptive element. It may be called anticomponent, in analogy to the matter and antimatter. It is interesting to note that the block capital letters of the alphabet can be rather neatly characterized by the number of straight lines, the number of curved lines (arches), the number of bays, and the number of lakes. The letter A has three straight lines, one bay, and one lake; the letter B has one straight line, two curved lines, and two lakes, etc. These four numbers are good features. But the last two of the four components are in reality anticomponents.

Summarizing, we may say that the most subtle part of the structural approach is found in the "selection" of components (or anticomponents) and the recognition algorithm for these components. The selection must obviously be made so as to maximize the efficiency of the entire recognition

task, but there is no way of solving such a maximization problem except by hunch and trial-and-error approach. One obvious trouble is that the components easy for human recognition are usually very difficult for machine recognition. Some components can be defined by definite criteria, but in many cases we can only illustrate a component by paradigms. This is characteristically a task of pattern recognition in the narrow sense mentioned in Section 1.1.

The remaining task in the structural approach—which is sometimes called, somewhat misleadingly, grammatical approach, linguistic approach, descriptive approach, etc.—is the description of the relations among the components by a combination of generative rules. It is important to develop a special fixed way to express such a description in a linear form, so that the computer can handle it. The structural formula for a molecule with single bonds, double bonds, hexagons, etc. is a two-dimensional shorthand for a three-dimensional structure. It is a well-studied problem to develop a linear translation of structural chemical formulas. The structural approach of pattern recognition has to go through this linearization problem once somewhere along the line. After such a linear (well-formed) formula is developed, we have to devise an algorithm to place each object in one particular "formula." This is the problem of generative sorting as introduced in Section 1.1.

It should be noted that it is a gross overstatement on the part of some proponents of the "descriptive method" to claim that their descriptive approach is not a problem of classification. As was made clear, there are usually a number of objects that fit in a single formula, hence a formula corresponds to a class. The purpose of the structural approach is to place each object in an appropriate class represented by a formula. The only difference from other classificatory problems is that we usually do not enumerate all the possible classes that can exist. The formula itself may be considered the feature of the class. This is a kind of sorting problem because the possible classes are in principle predetermined and well defined by the generative rules although they are not all enumerated.

For the rest of this chapter, we limit ourselves mainly to the pattern recognition problem in the narrow sense of the word, and to the clustering problem, and refer only occasionally to the two kinds of sorting problems.

1.3. Pattern Recognition As Induction

It is often said that pattern recognition is inductive in nature. This usually implies vaguely that some nonlogical elements are inevitable in the process

of pattern recognition. But how logically indeterminate the whole problem of pattern recognition is, and in what fashion the process of pattern recognition fits in the general form of inductive inference have not been much studied, except in two papers by the author (^{1,4}).

Suppose we are given m objects (paradigms) of class I and m objects (paradigms) of class II together with the observational results on those objects with regard to n predicates. It is assumed that no two objects among these $2m$ objects are identical with each other with regard to the n predicates. Suppose that we are now given a new object which is identical with none of the $2m$ objects, and we are asked to place it in one or the other of the two classes. This is a schematized model of the pattern recognition problem. Our conclusion is that, from a purely logical point of view, there is no reason to prefer one class to another in placing the new object, no matter how strongly it may resemble certain paradigms of one class. This is so because the theorem of the Ugly Duckling (¹) states that the number of predicates shared by the m paradigms of class I and the new object is always exactly the same as the number of predicates shared by the m paradigms of class II and the new object. Eligible for this counting is any logical function of the original n predicates. The total number of such eligible predicates is 2^{2^n} if there are no logical constraints* among the original predicates, and the number of shared predicates in the above sense is $2^{(2^n - m - 1)}$ for either class. This shows definitively that there is no logical solution to the pattern recognition problem. Actually, this is a special case of a more general theorem to the effect that, from the logical point of view, there is no such thing as a class of similar objects (²).

If, however, there exists no class of similar objects at all in the world, we cannot form a concept or build up a scientific knowledge. There must exist classes in our world of experience. What was wrong in our logical argument above is that we gave an equal weight to every possible predicate. In real life, some predicates are more important than others. In fact, those predicates closely correlated with the properties of objects which affect our survival, reproduction, or other desires or values receive in our life higher importance than other predicates. In science, those predicate-variables are important which make theory building possible or easy. Thus, the importance of a predicate is inseparable from the purpose for which the resulting concepts are to be used. Classes exist because some of them are useful for

* If one of the n predicates implies another, it counts as one constraint. Every time there is one constraint, the exponent 2^n is reduced by one. The conclusion, however, does not change.

some purposes. Classes depend on the purposes which they serve. Thus, classes can change from one occasion to another. This is a manifestation of inductive ambiguity. The features are important properties in this sense, depending on what we want to do with the recognized patterns.

The inductive nature of pattern recognition can be made more obvious by formulating the problem of pattern recognition in the following way. We are dealing with three spaces: a hypotheses space $\{H_j\}$, a class space $\{A_k\}$, and an n dimensional representation space $\{x\}$. The former two spaces are here assumed to be discontinuous, while the last one is assumed to be continuous. Everytime we choose one hypothesis H_j , it determines the (deductive) probability that an arbitrary member of class A_k is located in the volume element $(x, x + dx)$ of the representation space. We denote this probability by $p(x|A_k, H_j) dx$. If we denote the (*a priori*) credibility of hypothesis H_j by $p(H_j)$ and the probability that an object of unknown properties belongs to class A_k by $p(A_k)$, we may equate the joint probability of x , A_k , and H_j to $p(x|A_k, H_j)p(A_k)p(H_j) dx$.

In the training period of pattern recognition, we are given class A_k and measurement data x , and we are actually evaluating the credibilities of various H_j , $p(H_j|x, A_k)$. We simplify this situation saying that we are choosing the right hypothesis (distribution), which can be identified with the hypothesis with the highest credibility. The conditional credibility $p(H_j|x, A_k)$ is an inverse conditional probability of the deductive conditional probability $p(x|A_k, H_j)$, the inversion being made with respect to hypothesis H_j and data x . In this sense, it is an inductive conditional probability which inevitably depend on the prior probability $p(H_j)$ of H_j in the sense of the Bayesian inversion formula.

In the application period, we are given an adopted hypothesis H_j and a set of measurements x and we are evaluating the probability of A_k on this basis, i.e., we are now evaluating $p(A_k|H_j, x)$, which is an inverse of the deductive conditional probability $p(x|A_k, H_j) dx$, inversion being made with respect to A_k and x . In this sense, this is also an inductive conditional probability which inevitably depends on the prior probability $p(A_k)$. When one particular hypothesis H_j is not chosen, the decision of class must be based on $\sum_j p(A_k|H_j, x)p(H_j)$, where $p(H_j)$ is now the updated value of the credibility of H_j , which in turn depends on its prior credibility.

This analysis shows that the pattern recognition is a double induction and the result depends crucially on the prior credibilities of hypotheses H_j . These credibilities reflect our evaluation of the "goodness" of hypotheses on the basis logically prior to the evidence (which are paradigms here). The "extraevidential" factors can contain anything from aesthetic evalua-

tion to experience in other fields. Other than this arbitrariness with regard to the prior credibility of H_j , we have another more basic arbitrariness with regard to the choice of the set $\{H_j\}$, because the above-mentioned Bayesian theory of induction is concerned with the evaluation of hypotheses when the available hypotheses are fixed. If a new hypothesis is introduced, the result has to be changed completely. The preparation of the set $\{H_j\}$ is an act of "abduction" and there is no fixed prescription for it. The actual process of abduction is known to be nonlogical and nonnecessary. The pattern recognition thus reflects not only the arbitrariness of the prior probabilities of available hypotheses, but also the arbitrariness of the choice of the hypotheses to be examined. Of course, verbally, we can put these two kinds of arbitrariness into one by considering the set of all possible or potential hypotheses. Such a set, however, probably exists only in words.

It is interesting to note that clustering is, in a sense, comparable to abduction as defined above because it has to create the set $\{A_k\}$ and the set $\{H_j\}$. The so-called parametric approach to pattern recognition may be compared to a strict hypotheses evaluation, but the nonparametric approach has to carry out some degree of abduction too.

1.4. Decision Procedure and Features

In the last section, we considered three spaces, of which one was the hypothesis space $\{H_j\}$, and assumed that each H_j determines a fixed distribution $p(x|A_k, H_j) dx$. We can, alternatively, assume the distribution $p(x|A_k) dx$ to be a yet-to-be-determined unknown function and suppress the symbol H_j from our description. In this picture, the deductive probability $p(x|A_k) dx$ and the "recognition probability" $p(A_k|x)$ are related by

$$p(x|A_k)p(A_k) = p(x, A_k) = p(A_k|x)p(x) \quad (1)$$

The prior probability $p(H_j)$ is hidden in this formula, but it can influence the desirability of the function form of $p(x|A_k)$, hence of $p(x, A_k)$ and $p(A_k|x)$.

The first task of pattern recognition in this description is to determine the function (1) from the paradigm points. If we agree that the paradigm points and the later arrivals are all taken from the same population, $p(A_k)$ can be considered a fixed quantity. The function $p(x)$ can be estimated from the overall density of the paradigm points. Hence, if we have the function (1), then we can derive the recognition probability $p(A_k|x)$. In the so-called parametric approach, the functional form of the deductive probability

$p(x|A_k)$ is assumed and a few parameters are left undecided. From the paradigm points, we can make an estimate of these parameters by some rules taken from estimation theory. But in most of the practical problems, the distributions have irregular forms and do not fit well in any simple known distribution functions with a few parameters. In the nonparametric case, the arbitrariness of the function is too great to let the function be determined by a relatively small number of paradigms. A sound policy in restricting the too great arbitrariness would be to exclude any function which changes appreciably within a distance comparable to the average distance between neighboring paradigm points. There is no logical necessity for this policy, and it reflects our latent preference represented by $p(H_j)$ of the last section.

To be practical, we really do not need the recognitive probabilities themselves. What is important is to know which $p(A_k|x)$ has the largest value at each point and for different A_k . For instance, if there are two classes I and II, all that counts is to know which sign holds in

$$p(\text{I}|x)/p(\text{II}|x) \geq 1 \quad (2)$$

Any function $f(x)$ may be called a "decision function" if it satisfies the condition that the three alternative signs in

$$f(x) \geq 0 \quad (3)$$

coincide with the three alternative signs in (2). The task of pattern recognition may be said to lie in finding an appropriate decision function $f(x)$. It is desirable to choose a simple function for $f(x)$ which is more or less in agreement with the paradigm points.

Many algorithms bypass the probabilistic consideration given above and define $f(x)$ by the condition that

$$\begin{aligned} f(x) &> 0 & \text{for all } x \in \text{I} \\ f(x) &< 0 & \text{for all } x \in \text{II} \end{aligned} \quad (4)$$

Unless the same point belongs to both classes, (4) can always be satisfied if a sufficiently complicated function is allowed for $f(x)$. But, in accordance with the spirit of the smoothing policy mentioned above, we may require that none of the radii of curvature of the hypersurface $f(x) = 0$ be comparable to or smaller than the average distance of nearest neighbors. This requirement again is attributable to extraevidential consideration and is not logical in nature.

Now, if a reasonable decision function $f(x)$ of (3) or (4) is obtained, we may consider the variable $z = f(x)$ a final discriminatory feature, and the task of pattern recognition characterized as a search for features seems to have achieved its goal. But, the matter is not that simple.

The basic flaw in the above discussion lies in the fact that we have not given attention to the number of dimensions of the representation space $\{x\}$ and to the number of paradigms available. In most practical problems, the number of measured quantities for each object (i.e., the number of components in the original representation vector) is very large. It can very easily become more than 100. The number of paradigms, on the other hand, is usually small; very often it is also of the order of 100. Under such a condition, the distribution of paradigm points in the representation space is so sparse that we can hardly think of a density, and even less a function $f(x)$ in the sense of (3). As far as the decision function in the sense of (4) is considered, a linear function $f(x)$ can automatically satisfy (4) no matter how the points are distributed, provided the number of dimensions is large enough for a given number of paradigms. Roughly speaking in order to be able to talk about a decision function in the sense of (3) or (4), it is necessary (probably not sufficient) that the region of the representation space be finite and the number of paradigm points be considerably larger than the number of dimensions of the space.

This entails that, in order to make the pattern recognition, it is necessary to decrease the number of dimensions before we can even start the main portion of class discrimination. Of course, in decreasing the number of dimensions, we have to eliminate “less-important” variables and keep “more-important” variables. This is actually the vital part of the entire process of feature compression.

It is known that the decision based on the nearest-neighbor policy (placing the newcomer to the class to which its nearest paradigm belongs) is simple and usually proves to be quite good. Mathematically, it is shown that, if the paradigm density becomes infinite, the nearest-neighbor decision becomes as good as the Bayesian decision. But it is a mistake to claim that the reduction of dimensions or feature compression is not necessary because the distance can be directly calculated in the original high-dimensional representation space and the nearest neighbor can easily be found. A naïve statistical intuition which says that it is too risky to depend on the single nearest neighbor and that we should rather see the trend of a certain number of near neighbors may not be entirely unfounded. In fact, the above-mentioned mathematical theorem says nothing when the paradigm density is rather low, as is usually the case anyway. A realistic approach would be,

after all, to relatively increase the paradigm density by decreasing the dimensions and to consider the effect of a reasonable number of near neighbors. This number can be gradually decreased as the paradigm density increases.

1.5. Selection of Variables

Pattern recognition is a matter of selection of variables from the beginning to the end. Selection, obviously, implies two things: (1) by selection, the number of items in the collection becomes less, and (2) the selected ones are more important, in some sense, than those not selected. Thus, the aim of pattern recognition is, so to speak, to narrow down to as few and as important variables as possible. At the very start, we have to collect data by measuring a certain number of variables on the objects. The entirety of information contained in these raw data is called the "scope." By an ulterior selection or transformation, the scope cannot be increased. The determination of the scope is already a selection of variables. It is desirable to make the number of variables to be measured as small as possible without losing the important part of the information. At the very last stage of pattern recognition, we finally choose one function such that its positive value, say, indicates that the object belongs to class I and its negative value indicates that it belongs to class II. Between these two selections of variables at the beginning and at the end, there are intermediate stages of the selection of variables explained in the last section. These intermediate stages are by no means less important than the other two. If we use somewhat loosely the word feature as meaning important variables, the entire process of pattern recognition can be characterized as a sequential compression of features, not quite unlike a process of chemical purification of some rare element.

To make a selection, we have to have some criterion or standard of importance. As we have already stated, the ultimate source of importance lies in the utility of the resulting classification. But, unfortunately, translation of the utility of resulting classification in terms of the importance of individual variables is, in most cases, impossible. For this reason, we have to use various other standards of importance in the process of selection with the hope that the resulting classification will turn out to be useful. We cannot mention all possible standards of importance, but we can mention a few basic classifications of these standards.

One way of classifying the standards of importance is to see whether a particular standard is "external" or "internal." Usually, in the problem of clustering and pattern recognition, we have at hand a collection of objects

with their known properties. Some standards are derivable from the statistical properties of this collection. In that case, we speak of an internal standard. On the other hand, if the standard is imposed by anything else than the statistical ensemble of the objects at hand, we speak of an external standard.

Another way of classifying the standards is to determine whether the standard is obtained by a universal method applicable to any kind of object or is applicable only to a specific type of object.

Another way to characterize a standard of selection is by its being dependent on or independent of the paradigms (class samples) of classes. In the former case, we speak of a postparadigm standard and in the latter case we speak of a preparadigm standard. This distinction should not be confused with the distinction between the internal and external standard. The internal criterion can depend on the collection of objects whose class belonging is unknown, while the postparadigm standard depends only on the objects with known class affiliation. In other words, postparadigm standards form a subclass of internal standards.

Finally, we should differentiate a purely automatic standard and a man-tutored standard. We can expect a better result if a human judgment is used at an appropriate stage of selection, but it is more challenging and desirable to devise purely mechanizable standards which are just as good.

One very common standard of selection is the invariance for a certain class of transformation. For instance, most of the written characters of different languages allow for the following transformations: (1) displacement, (2) uniform enlargement and contraction, (3) rotation by an angle, within certain limits, (4) slight deformations which do not change topological connections. The knowledge of this invariance can be translated into a standard of selection requiring that any property which is not invariant for these transformation be eliminated. Not all invariant properties are good variables, but certainly this selection rule considerably decreases the number of variables. The only difficulty is that no simple mathematical expression for these transformations is available, particularly for (3) and (4), which have certain undefined limitations. Conversely, mathematically definable transformations (such as Fourier transformation, rigid-body transformation) do not exactly correspond to the actually permissible transformation. Yet, a proposal to use integral geometry for character recognition, which lies along this line, is highly interesting⁽⁶⁾. Criteria of invariance is an external rule, because it does not depend on the collection of letters you have at hand. It is universal in the sense that it remains almost the same for Chinese characters and the Latin alphabet, but specific in the sense that

it is not applicable, say, to speech recognition. It is preparadigm because it is external. It is meant to be automatic, because once we have fixed the rules in computer-legible language, a machine can do the job without our help. Of course, it may not be so easy to translate the allowable limitations in (3) and (4) into a computer language.

Formally speaking, any selection of variables can be interpreted as an extraction of invariant variables because we can, at least in principle, always imagine some kind of transformation which leaves the selected variables invariant and which alters the values of eliminated variables. But the point is that, if the transformation in question does not allow a simple interpretation, it does not help us much.

1.6. Distance and Feature

What we said in connection with the Theorem of the Ugly Duckling may be restated in the form: the Hamming distance between any two arbitrary object vectors is the same if all possible predicates are taken into account. This destroys any possibility of class (concept) formation. It follows that if we want to see classes, we have either to modify the concept of Hamming distance by giving uneven weights to different predicates, or else to keep the Hamming distance but eliminate unimportant predicates from the list. The second alternative is actually a special case of the first alternative if some weights are allowed to become zero and the remaining weights have the same value.

This shows that the existence of classes implies the existence of unequal distances among objects. Conversely, unequal distances among objects leads to some sort of grouping, since we can put "near" objects together in a box and call it a group or class. Thus, the concept of distance and the concept of class are inseparable. Feature being the essence of class formation, it is also inseparable from the concept of distance.

The concept of distance alone admittedly does not determine classes, yet it helps, in some measure, the intermediate stage of feature compression. The reason why this is possible is as follows. When the points (objects) are given with their mutual distances, it is to be expected that some of the variables (vector components) do not contribute very much to the mutual distances of the points at hand. For instance, if a variable has the same value for all points, it certainly has no contribution to the distances. Such unimportant variables can be eliminated, making one step forward toward feature compression. If there are no such unimportant variables among the original variables, we can apply some kind of transformation to the varia-

bles in such a way that some of the new variables become unimportant. This is the basic idea of the method of SELFIC which we explain in Section 3. SELFIC is an external, universal, preparadigm, totally automatic method of feature compression. A slight modification of SELFIC called CLAFIC is a postparadigm method.

2. A CONCRETE EXAMPLE OF FEATURE COMPRESSION—HANDWRITTEN ZIP CODE READER

2.1. Nature of the Problem

In Section I, we explained the necessity of sequential feature compression in pattern recognition on an abstract level. In Section 2, we will briefly describe an actual machine which provides us with a beautiful example of successive feature compression. The machine in question is the automatic letter sorter based on handwritten ZIP codes which is actually in operation at Tokyo central post office and fifteen other post offices. This machine was developed by Toshiba Electric Co., and the main engineers who were responsible for this project were Hiroshi Genchi, Ken'ichi Mori, Sadakazu Watanabe, and Sumio Katsuragi (?).^{*} In other countries, test machines have been developed to read typewritten ZIP codes, but the Toshiba machine can rightly claim to be not only the first ZIP code reader ever to be placed in practical use, but also to be the first ZIP code reader that reads handwritten ZIP codes. It is desirable that they further improve the machine to reduce the error rate, which is at present comparable to that of a human sorter.

As in other industrial countries, the mail-sorting problem is becoming in Japan one of the major bottlenecks of daily communications. The problem in Japan is doubly difficult because approximately 90% of ordinary letters are handwritten and this percentage cannot be expected to go down appreciably due to the nature of the language (making typewriting a complicated and slow task). This means that the automatic ZIP code reader in Japan has to be geared to handle handwritten numerals, and to compete with human sorters who are considerably faster than in other countries (probably due to the Chinese ideograms, which facilitate visual recognition).

^{*} I wish to thank these authors not only for allowing me to use their results in this chapter, but also for personally demonstrating their machine to me in Tokyo. They are developing a series of newer machines.

The problem is further complicated by the variety of writing instruments used: ball-point pen, ordinary pen, fountain pen, pencil, fiber pen, and writing brush. The present speed of the automatic ZIP code sorter seems to be about one order of magnitude faster than a human sorter (without ZIP), and the error rate is approximately of the same order of magnitude as a human sorter ($0.1 \sim 1.0\%$ for 15 stackers).

The method of compression used in this machine is, according to the classification given in Section 1, *specific* because it is designed specifically for the Arabic numerals handwritten by Japanese. It is *external* in the sense that the sorting process is not affected by the actual letters (and cards) which are placed in the hopper, but it is *internal* and *postparadigm* in the sense that all the features are derived from a huge collection of real handwritten numerals. The process is, of course, fully automatic, too, leaving room for further adaptation. It is very interesting to note that the feature-compression process used in this machine can be regarded also as a case of structure analysis (so-called descriptive method) in some sense. We shall explain this point after we have described the actual process of feature compression. The feature compression is done in five successive stages in this machine, and, roughly speaking, the information is reduced by a factor $\frac{1}{3}$ or $\frac{1}{4}$ at each stage, according to the designers' report.

2.2. Compression of Invariants

As explained before, practically all facets of feature compression can be interpreted as extraction of invariants, but such interpretation becomes particularly meaningful when the allowed transformations have simple meaning. We shall examine how the various invariances are treated in the Toshiba sorter.

(a) Displacement Invariance. The envelopes and cards are presently provided with four boxes of dimension 7×10 mm with 2 mm separation, printed in light blue near the top edge (Fig. 1). This determines the rough location and size of the numerals. But the machine does not recognize the boxes or the edges of the envelope or card. It locates the exact positions of characters by projecting the printed patterns horizontally and vertically, so that the continued dark line segments (projection of printed figure) serve as the reading windows. It is difficult to evaluate it numerically, but this masking represents a large reduction of possibilities, implying considerable information compression.

The window opening is scanned at every 0.2 mm. The blue box in which the numeral is supposed to be written is 7×10 mm, but in some cases the

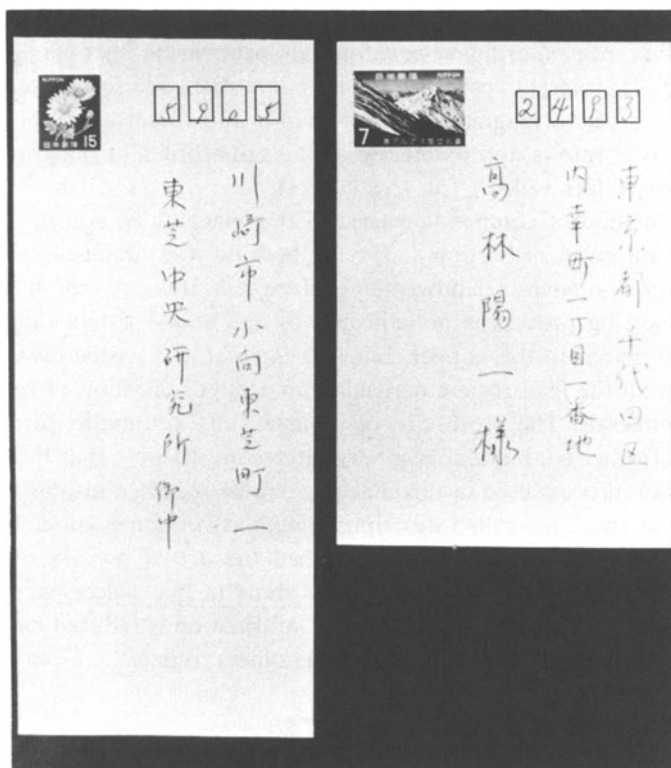


Fig. 1. ZIP codes handprinted inside light blue boxes near the top edge of an envelope and a card.

real window size has to be allowed to become about 20% larger than the box in linear dimension. The scanning spot size is intentionally fixed at a relatively large value, in order to fill out gaps in grayness of the written characters, to eliminate sharp noise, and, more generally, to obtain more smoothed-out input signals. (Note: This is also part of information compression.)

(b) Grayness Invariance. Compared with printed letters, handwritten letters have a wide range of grayness, and therefore require a careful study to determine an appropriate threshold producing a useful binary signal at each mesh point.

(c) Line-width Invariance. This is another invariance which we did not discuss in the last section. But it is an obvious characteristic of letters that we can change the thickness of lines (strokes) without altering the

identity of the letter. The large variety of writing instruments causes the thickness to vary from one meshpoint to about five meshpoints. The Toshiba machine has the property of reducing the thickness by one to two mesh points. They could repeat this thinning routine a second time, but they found it was unnecessary. The basic function of the thinning process is the following: If a black mesh point is found on the boundary of a stroke, it should be changed to white. But care is taken so that this does not cut an already thin stroke or enlarge a gap within a stroke. To determine whether a mesh point is to be black or white according to this principle, one has to check its vicinity. The machine therefore takes up one 3×4 submatrix at a time to determine a binary output at the center.

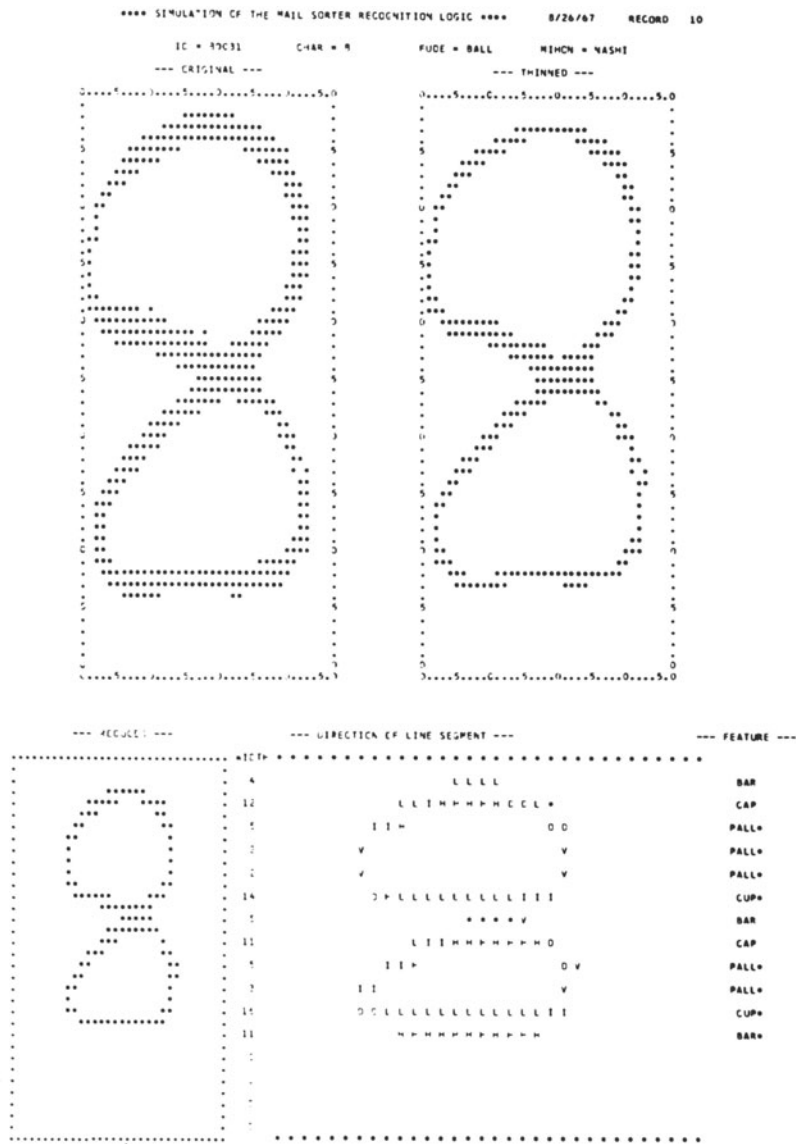
(d) Size Invariance. In this machine, a letter which is taller than a certain threshold is compressed by factor $\frac{1}{2}$ in linear dimension. To do this, they apply the majority rule to a 4×4 submatrix and obtain one binary output. This has a side effect of smoothing, sometimes bridging a one-bit gap in a stroke (which is desirable), sometimes filling up a one-bit white inside figure like 6 or 8 (which is undesirable). By this rule of reduction of oversize letters, the variation in the linear dimension of actual letters is confined within the max-min ratio of 2 to 1. The remaining variation in the size is further eliminated to a certain extent before the last recognition stage by what I shall call "run contraction" and explain later. See Fig. 2 for how the thinning and reducing algorithm work.

After the thinning and size-reducing routines, the signal of each letter is stored as a set of 32×32 mesh points, each being either black or white.

(e) Rotational and Topological Invariances. The feature-extraction method explained in the following two subsections partly takes care of these two invariances, but the remaining variety due to these two invariances is considered as a variety of character types of a letter and stored in the memory of the machine.

2.3. Local Features

It is well known that character recognition becomes easier if a variable is introduced to represent the slope of a stroke. In the Japanese ZIP code reader, a 3×3 submatrix is taken about a point and seven characterizations (classes) are determined depending on the distribution of the black and white points on these nine mesh points: blank, vertical, increasing, decreasing, high-horizontal, low-horizontal, and undecided. Figure 3 gives these classes schematically, but in reality, each of seven classes includes several pos-



sibilities. For instance, the “increasing” class consists of the five different configurations given in Fig. 4.

Of the 32×32 mesh points which are stored, only one-half of them, 32×16 , are chosen to be the centers of the 3×3 submatrices, namely, only

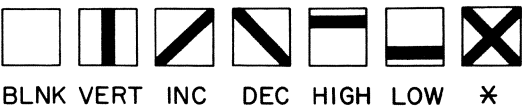


Fig. 3. Characterizations of the slope of a stroke.

every other horizontal row is taken for this local feature extraction. As far as a single 3×3 submatrix is concerned, $2^9 = 512$ possibilities are compressed into seven possibilities, i.e., nine bits are compressed into $\log_2 7 = 2.81$ bits. But a new redundancy is introduced by the overlapping of the 3×3 submatrices. The original representation can carry one bit at each of 32×32 points at maximum. (Of course, the majority of the possibilities does not correspond to any character.) In the new representation, the maximum information of $\log_2 7$ bits cannot be carried at each of the 32×16 points, because there are some necessary mutual constraints between neighboring points in this description. The average must be, of course, less than two bits per point because we cannot reconstruct the original representation from the second representation. But, as far as the memory device is concerned, the second representation would require even more space because we have to provide $32 \times 16 \times \log 7$ fits of capacity as compared with $32 \times 32 \times 1$ bits. (See the “direction of line segment” part of Fig. 2 for this stage of feature compression.)

2.4. Horizontal Zone Feature

If the local feature extraction described in the last subsection does not result in any substantial information compression, it prepares the ground for a large and meaningful compression at the next stage. Each (32-unit-long) horizontal zone can, in the absence of constraints, be any one of the 7^{32} possible strings because there are 32 points and each point has seven possible “states.” These 7^{32} items are classified into 16 classes (Fig. 5). Bit-wise, this is a compression from $\log_2(7^{32}) = 89.9$ bits down to $\log_2 16 = 4$ bits; the ratio is more than 10. Each of 16 classes therefore contains a large number of different strings of states. For instance, a string of the structure: blank–increase–high–decrease–blank is considered as a CAP, where blank

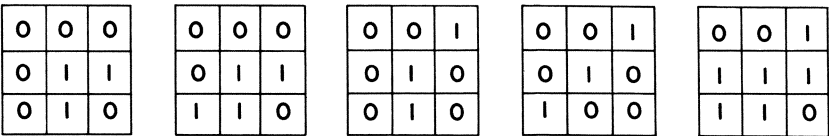


Fig. 4. The five configurations comprising the INC class of Fig. 3.

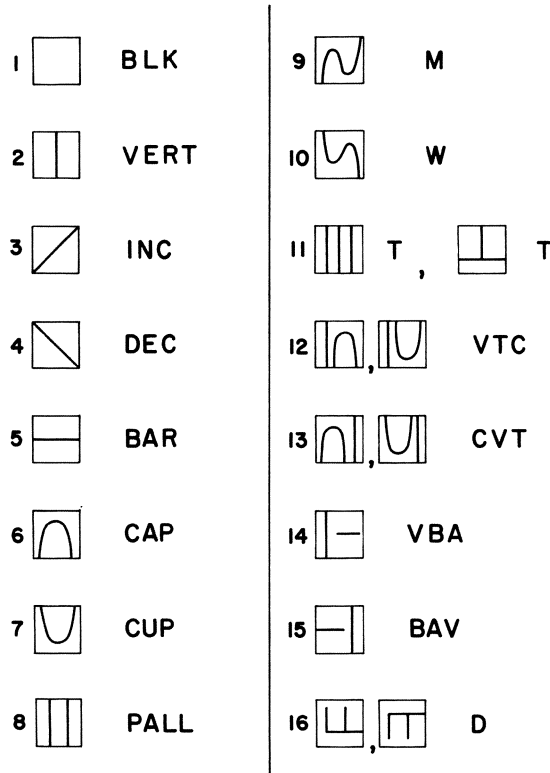


Fig. 5. Classification of horizontal zone features.

stands for a run of arbitrary number (of course, with a certain upper limit) of blanks and high can also be one, two, three, ... highs. This last rule is what I called "run contraction" above. The run contraction makes it more economical to use a sequential logic, because what counts is transition of states rather than the state of each position.

The designers of this sorter report that they have chosen horizontal zones rather than vertical zones because preliminary psychological experiments have revealed that horizontally dissected patterns contain more relevant information for recognition than vertically dissected patterns. (See the "feature" part of Fig. 2 for this stage of compression.)

2.5. Global Features

There are 16 horizontal zones and each zone can be any one of 16 states. The "relation" among the 16 zones is their order from the top to the bottom.

The designers of the sorter, however, added one more relation between the zones. This relation concerns the relative positions of the first (counting from the left) black point on the last row of a zone and that on the first new row added on the next zone. If these two points coincide within an allowance of ± 1 , we say that the second zone starts at the same place as the first, and if not, we say that the second zone starts at a different place. This predicate is really a relation between two consecutive zones, but, for convenience, it can be considered as a property of the second row. This can be expressed as one bit of information added to the zone. In Fig. 2, the asterisk means that the zone starts at the same place as the previous zone. This information effectively serves the purpose of discriminating "3" from "5". In "3", the first vertical line segment starts at a different place than the top bar, while in "5", the first vertical line segment starts at the same place as the top bar.

Thus, each zone has five bits or 32 varieties and there are 16 zones, totaling $32^{16} \approx 10^{23}$ possibilities or $23 \log_2 10$ bits. This must now be classified into ten classes or $\log_2 10 = 3.32$ bits. The bit reduction ratio is 23. For example, the vertical sequence: CAP-PALL-CUP-CAP-PALL-CUP can be interpreted as "8" (see Fig. 2). Here again, the run-contraction rule is applicable, at least for PALL's. Each input figure is thus translated into a transition diagram. Various standard transition diagrams (collected statistically) are stored in the machine, corresponding to each numeral. (About 70 standard diagrams are stored and they can be updated.) The input transition diagram is compared with the stored standard transition diagrams. A perfect match is not sought, but the minimum discrepancy is sought. If the minimum discrepancy is larger than a certain threshold or if the same amount of minimum discrepancy is obtained for more than one numeral, the input is rejected. If the signal is rejected, the horizontal zones are taken with respect to those rows which were not used as the center of the 3×3 matrices in the first pass and recognition procedure is tried again.

The recognition system has to be sequential due to the nature of the problem. Even then, there is no systematic method of design to cope with the tremendously large amount of possible branching that exists in the present problem. The actual design was made with the help of various heuristic methods.

2.6. Feature Compression As Structural Analysis

Each mesh point has two possible states, black and white, but it is assigned seven states according to the behavior of the figure in its vicinity.

This corresponds, as was seen in Section 2.3, to seven classes defined on the possible $2^9 = 512$ states of the 3×3 submatrix. Each one of these seven or the totality of the seven may be called feature(s) extracted from 512 alternative binary properties. Or we can introduce a variable with seven possible values corresponding to these seven cases, and call it a “feature variable.”

A horizontal zone is then a string of 32 positions, each one of them taking up one or the other of the seven possible states, thus producing 7^{32} alternatives. This is again classified into 16 categories. Each one of these 16 classes or the entire 16 classes may be considered again a feature or features. Or, correspondingly, we can introduce a “feature variable” with 16 possible values characterizing each zone. It should be noted that the majority of 7^{32} possibilities are meaningless ones.

This feature compression can also be considered as a structure analysis. We can, in fact, regard the seven features as constituent components, INC, DEC, UPPER-HOR, etc., which are placed in a row. The structural relations among these components determine the classes such as CAP, PALL, etc. Certain ways of arranging the constituent components are meaningful and some others are not. The meaningful ones are defined by certain rules. A description such as blank–increase–high–decrease–blank is not the designation of a particular object, but is already a rule because blank and high actually stand for any number of them. There are certainly a few universal rules governing all meaningful classes. For instance, the rule that there should not be more than two “increases” placed consecutively in a row, or something of the kind, must be true for all meaningful strings. Of course, it is doubtful that we can define the meaningful ones only by such universal rules.

This structure analysis may be compared to a grammatical analysis of sentences. Each one of 512 configurations of the 3×3 submatrix is a “word.” The seven classes covering the 512 words are “parts of speech.” The horizontal zone is a sentence expressed in terms of “parts of speech.” The meaningful horizontal zone is a “grammatical sentence,” and the types such as CAP, etc., are types of grammatical sentences. The rules defining the meaningful horizontal zones are grammatical rules. The only difference is that these grammatical rules cannot be formulated as universal rules as is the case in the usual grammatical rules of natural languages. In reality, in natural languages, too, some variations of the universal rules are applicable to certain types of meaningful sentences.

As was mentioned in Section 1, the rules, rather than the specification of each individual meaningful string, acquire their true superiority only if the number of individual strings becomes infinite so that the enumeration

of individual specifications becomes impossible. In practical pattern recognition problems, however, the number of individual cases is always finite, and the use of rules in preference to the collection of individual specifications is just a matter of convenience and not a matter of principle.

A similar consideration also applies to the final stage of vertical analysis. There are, in this case, 32 different components (parts of speech), counting twice each component according to the presence or absence of the asterisk. We can write the asterisk wherever necessary as if it were another component in the vertical string, although it is really not a component. We can consider this asterisk as a tool of linearization we talked about in Section 1.

3. DISCRIMINATORY FEATURE COMPRESSION—*SELFIC*

3.1. Rotations in Representation Space

The automatic ZIP code reader described in the preceding chapter is a good example showing that, once the features are selected and the decision method based on them is fixed, the remaining task becomes simply a matter of sorting according to the terminology of Section 1. The secret of success of a machine of this kind lies, of course, in the selection of features or good variables, including the final decision function. In the description of the ZIP code reader, the authors do not spend much space explaining why, for instance, they chose CAP, PALL, etc. as the features of horizontal zones. This crucial choice was probably guided by a semiheuristic trial-and-error method. It should be noted that the choice in this case has to be made exclusively for the specific purpose of reading the handwritten ZIP codes.

In this section, we want to explain a computer-feasible algorithm for choosing fewer and more important variables from a given number of observed variables. The method is called SELFIC (self-featuring information compression). The method is “intermediate” in the sense that it has nothing to do with both ends of the chain of sequential feature compression. In other words, it has nothing to do with the choice of “scope” (observed data) or the choice of the “decision function.” The method is “internal” in the sense that we assume that a certain number of objects with known properties (but not class-belonging) are given and the rule of selection is solely decided by this statistical collection and not by some other external consideration. The method is “universal” because it is applicable

to any pattern recognition or clustering problem insofar as each object is represented by a vector (a set of values of n variables). The price we pay for being universal is that the method may not be the best available for a specific problem. But, even in a specific problem, it is often advisable to use SELFIC as an auxiliary processing before (or after, or with) other specific methods of feature compression. It can also be combined with structure analysis.

We assume that we are given a collection of objects each of which is represented by an n -component vector $\{x_i\}$, $i = 1, 2, \dots, n$. The n -dimensional real space is the "representation space" of the objects. We may or may not know the class-belonging of some or all objects, but as far as SELFIC is concerned, this information is unused. In the CLAFIC method which is explained later, we consider objects *qua* class paradigms, but not in SELFIC. There are in the collection m different types of objects, $a = 1, 2, \dots, m$, i.e., m different vectors, each type a appearing with relative frequency w_a

$$w_a \geq 0 \quad \text{and} \quad \sum_{a=1}^m w_a = 1 \quad (5)$$

The weight w_a may be considered, if so desired, as the measure of importance of the type a instead of the mere relative frequency. The "ensemble" of objects is determined by two sets: $\{x_i^{(a)}\}$, $a = 1, 2, \dots, m$; $i = 1, 2, \dots, n$, and $\{w_a\}$, $a = 1, 2, \dots, m$.

When n is not very large, it is often profitable to include higher powers and products (multiplied by constant coefficients, if desired) of the original variables x_i as additional components of the representation vector as if they were independent variables. If we denote this new vector by $\{z_i\}$, $i = 1, 2, \dots, n'$, the transformation from $\{x_i\}$ to $\{z_i\}$ is always possible and unique, but the inverse transformation from $\{z_i\}$ to $\{x_i\}$ may sometimes not exist, in the sense that some points in the z -space may have no corresponding points in the x -space. In spite of this defect, inclusion of higher powers often proves to be useful for various reasons. One of them is that a hyperplane in the z -space becomes a curved hypersurface in the x -space, more suitable for decision making.

We can also consider a complete orthogonal function set $\{\varphi_i(x_1, x_2, \dots, x_n)\}$, $i = 1, 2, \dots, \infty$, in a certain domain of the x -space, where the labeling i is such that the higher the value of i , the more quickly the value of φ_i changes as function of $\{x_i\}$. Taking the first n' functions $\{\varphi_1, \varphi_2, \dots, \varphi_{n'}\}$, we can get a new n' -dimensional representation space. We can obtain a smoothing effect in the x -space by this procedure. However, we do not spend much space on these techniques of preliminary transformation

in this chapter. When we speak of the x -space, it may mean the original x -space or the representation space obtained through one of those preliminary transformations.

We consider, in the following, orthogonal transformations in the x -space,

$$y_j = \sum_{i=1}^n T_{ji} x_i, \quad x_i = \sum_{j=1}^n T_{ij}^{-1} y_j \quad (6)$$

with

$$(T^{-1})_{ij} = T_{ji} \quad \text{or} \quad T^{-1} = T^T \quad (7)$$

It is easy to see that the condition (7) is equivalent to the condition that the square of the vector length remains invariant,

$$\sum_{j=1}^n y_j^2 = \sum_{i=1}^n x_i^2 \quad (8)$$

or, equivalently,

$$(y^T \cdot y) = (x^T \cdot x)$$

for every vector $\{x_i\}$, or to the condition that the distance between two vectors remains invariant,

$$\left(\sum_{j=1}^n |y_j^{(2)} - y_j^{(1)}|^2 \right)^{1/2} = \left(\sum_{i=1}^n |x_i^{(2)} - x_i^{(1)}|^2 \right)^{1/2} \quad (9)$$

for any arbitrary pair of vectors $\{x_i^{(1)}\}$ and $\{x_i^{(2)}\}$, or to the condition that the scalar product between any two vectors remains invariant:

$$(y^{(2)T} \cdot y^{(1)}) = \sum_{j=1}^n y_j^{(2)} y_j^{(1)} = \sum_{i=1}^n x_i^{(2)} x_i^{(1)} = (x^{(2)T} \cdot x^{(1)}) \quad (10)$$

The orthogonality condition (7) implies

$$(\det |T|)^2 = \det |T| \det |T^T| = \det |TT^{-1}| = 1 \quad (11)$$

hence

$$\det |T| = \pm 1 \quad (12)$$

We speak of *rotations* when the upper sign applies in (12) and *reflections* when the lower sign applies. In our discussion, however, we can limit ourselves to rotations without losing the versatility of our mathematical tools.

As explained in the earlier sections the concept of distance is vital in pattern recognition. We use in our discussion the Euclidean distance (9)

as the measure of distance (dissimilarity) of two objects. Our discussion in the following is limited to linear transformations of variables which do not change the measure of dissimilarity as expressed by (9).

On the other hand, the preliminary transformation changes the concept of distance. For instance, if we start with two variables x_1 and x_2 and if we add $c_{11}x_1^2$, $c_{12}x_1x_2$, and $c_{22}x_2^2$ as the additional variables x_3 , x_4 , x_5 , the distance becomes

$$\begin{aligned} & \{(x_1^{(1)} - x_1^{(2)})^2 + (x_2^{(1)} - x_2^{(2)})^2 + c_{11}^2[(x_1^{(1)})^2 - (x_1^{(2)})^2]^2 \\ & + c_{12}^2[x_1^{(1)}x_2^{(1)} - x_1^{(2)}x_2^{(2)}]^2 + c_{22}^2[(x_2^{(1)})^2 - (x_2^{(2)})^2]^2\}^{1/2} \end{aligned} \quad (13)$$

instead of the Euclidean distance $\{(x_1^{(1)} - x_1^{(2)})^2 + (x_2^{(1)} - x_2^{(2)})^2\}^{1/2}$. Correspondingly, the invariant for the rotations will be of the form (13). This is not such an unreasonable thing in many practical problems.

The distance concept involved in the transformation from $\{x_i\}$, $i = 1, 2, \dots, n$, to $\{\varphi_i(x)\}$, $i = 1, 2, \dots, n'$, is rather interesting. The Euclidean distance in the new space $\{\varphi_i(x)\}$ between two points $\{x^{(1)}\}$ and $\{x^{(2)}\}$ is zero, of course, at $x_i^{(1)} = x_i^{(2)}$, $i = 1, 2, \dots, n$, and remains small if

$$\sum_{i=1}^n (x_i^{(1)} - x_i^{(2)})^2 < \Delta$$

and become finite if

$$\sum_{i=1}^n (x_i^{(1)} - x_i^{(2)})^2 > \Delta$$

where the inequality symbol is to be understood in the sense of the order of magnitude. The threshold Δ is determined by the value n' of truncation process mentioned before, and is, in an order-of-magnitude sense, the distance in which the functions φ_i with $i > n'$ would oscillate fast. Roughly speaking, the distance in the modified representation has two values: far and near. This notion will remain unchanged by subsequent rotations T .

By rotation T , a vector $\{x_i\}$ is transformed into vector $\{y_i\}$. As a result, the ensemble $\{x_i^{(a)}, w^{(a)}\}$, $a = 1, 2, \dots, m$, is transformed into $\{y_j^{(a)}, w^{(a)}\}$, $a = 1, 2, \dots, m$ where $y_j^{(a)}$ and $x_i^{(a)}$ are related by T as in (6). The transformation is conceivable only for vectors whose length

$$\left\{ \sum_{i=1}^n (x_i^{(a)})^2 \right\}^{1/2} = \left\{ \sum_{j=1}^n (y_j^{(a)})^2 \right\}^{1/2}$$

is finite. In many practical problems, we are more interested in the relative

values of vector components than the vector length. On this ground, we assume from now on that all the vectors are normalized:

$$\sum_{i=1}^n (x_i^{(a)})^2 = \sum_{j=1}^n (y_j^{(a)})^2 = 1 \quad (14)$$

for all x . If the vector length has some useful meaning, its value should be retained as an additional scalar variable.

3.2. Minimum-Entropy Principle

Our purpose in introducing rotations in the representation space is to exploit it to obtain eventually fewer and better variables. Of course, by a rotation, the number of dimensions (variables) does not change. Suppose, however, that we rotate the axes in such a way that some of the new axes (variables) become very “unimportant” in some sense. Then we may allow ourselves to eliminate those unimportant variables from consideration, retaining fewer and more important variables. To be able to do this operation at all, the degree of “importance” must be “unevenly” distributed over different variables. In fact, if the degree of importance were “evenly” distributed over different variables, we would not know which ones to eliminate. We therefore now have to explicate the two vaguely defined notions of “importance” and “evenness” in a more mathematical language so that we can further proceed on this program.

Consider the quantity

$$\varrho_j^{(a)} = (y_j^{(a)})^2 \quad (15)$$

which satisfies the probability axioms due to the normalization (14):

$$\varrho_j^{(a)} \geq 0 \quad \text{and} \quad \sum_{j=1}^n \varrho_j^{(a)} = 1 \quad (16)$$

This $\varrho_j^{(a)}$ can be considered as the measure of importance (usefulness) of the j th axis in the new (i.e., rotated) coordinate system in expressing the a th object. It will be zero if the object vector has no projection on the j th axis and will take its maximum value, one, if the object vector lies in the direction of the j th axis. Then, the average of $\varrho_j^{(a)}$ in the ensemble

$$\varrho_j = \sum_{a=1}^m w^{(a)} \varrho_j^{(a)} \quad (17)$$

will also satisfy the probability axioms due to (5) and (16):

$$\varrho_j \geq 0 \quad \text{and} \quad \sum_{j=1}^n \varrho_j = 1 \quad (18)$$

This ϱ_j may be considered as the “average importance (usefulness) of the j th coordinate” of the rotated coordinate system in expressing the objects in the ensemble.

Since the ϱ 's satisfy the probability axioms, it will be very convenient to use the entropy function

$$S = - \sum_{j=1}^n \varrho_j \log \varrho_j \quad (19)$$

as the measure of “evenness” of distribution. It will take its minimum value, zero, when the ϱ 's are concentrated on just one coordinate axis, and take its maximum value, $\log n$, when they are uniformly distributed at the rate of $1/n$ on each coordinate. This S , of course, depends not only on the ensemble, but also on the y -coordinate system, or on the T if we fix the x -coordinate system. Explicating thus the notions of importance and evenness, we can now translate the guiding idea of the first paragraph of this section as the following normative principle:

Principle of Minimum Entropy. We should use an optimal coordinate system, which is defined as one which minimizes the entropy (19).

To obtain an optimal coordinate system, we have to introduce the Gibbs ensemble matrix (autocorrelation function) G_{ik} which is defined by

$$G_{ik} = \sum_{a=1}^n w^{(a)} x_i^{(a)} x_k^{(a)} \quad (20)$$

This matrix is symmetric and satisfies

$$G_{ii} = \varrho_i = \sum_{a=1}^n w^{(a)} (x_i^{(a)})^2 \geq 0$$

and

$$\text{tr } G = \sum_{i=1}^n \varrho_i = \sum_{a=1}^n w^{(a)} = 1 \quad (21)$$

An eigenvector ξ_i of G is defined as a vector satisfying

$$\sum_{k=1}^n (G_{ik} - \lambda \delta_{ik}) \xi_k = 0, \quad i = 1, 2, \dots, n \quad (22)$$

This set of n homogeneous linear equations for ξ_k has a solution if and only if

$$\det |G_{ik} - \lambda \delta_{ik}| = 0 \quad (23)$$

This n th degree algebraic equation for λ has, in general, n nonnegative solutions (called eigenvalues of G) which, for later convenience, we agree to label in the descending order of the magnitude of λ :

$$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_j \geq \cdots \geq \lambda_n \geq 0 \quad (24)$$

The eigenvector ξ_k obtained from (22) with a particular eigenvalue λ_j will be denoted by $\xi_k^{(j)}$,

$$\sum_{k=1}^n (G_{ik} - \lambda_j \delta_{ik}) \xi_k^{(j)} = 0 \quad (25)$$

It is easy to see that two eigenvectors corresponding to two different eigenvalues are orthogonal. Changing j to l in (25), we obtain

$$\sum_{i=1}^n (G_{ik} - \lambda_l \delta_{ik}) \xi_k^{(l)} = 0 \quad (26)$$

Multiplying (25) by $\xi_i^{(l)}$ from the left and multiplying (26) by $\xi_i^{(j)}$ from the left, we obtain, respectively,

$$\sum_{i=1}^n \sum_{k=1}^n \xi_i^{(l)} (G_{ik} - \lambda_j \delta_{ik}) \xi_k^{(j)} = 0 \quad (27)$$

and

$$\sum_{i=1}^n \sum_{k=1}^n \xi_i^{(j)} (G_{ik} - \lambda_l \delta_{ik}) \xi_k^{(l)} = 0 \quad (28)$$

If we interchange the dummy indices i and k in (28) and use the fact that G_{ik} is symmetric ($G_{ik} = G_{ki}$), we obtain from (28)

$$\sum_{i=1}^n \sum_{k=1}^n \xi_i^{(l)} (G_{ik} - \lambda_l \delta_{ik}) \xi_k^{(j)} = 0 \quad (29)$$

Subtracting (29) from (27), we obtain

$$(\lambda_j - \lambda_l) \sum_{i=1}^n \xi_i^{(l)} \xi_i^{(j)} = 0 \quad (30)$$

which implies

$$\sum_{i=1}^n \xi_i^{(l)} \xi_i^{(j)} = 0 \quad (31)$$

provided

$$\lambda_j \neq \lambda_l \quad (32)$$

If λ is a ν -ple root of (23), Eq. (22) determines a ν -dimensional subspace which is orthogonal to other eigenvectors. Within this ν -dimensional subspace we can arbitrarily take ν mutually orthogonal directions and choose them as the eigenvectors. By doing this, we obtain a set of n orthogonal eigenvectors which we agree to normalize so that

$$\sum_{i=1}^n \xi_i^{(j)} \xi_i^{(l)} = \delta_{jl}, \quad j, l = 1, 2, \dots, n \quad (33)$$

If we write an expression corresponding to (20) in the y -coordinate system, we shall have

$$G_{jl} = \sum_{a=1}^n w^{(a)} y_j^{(a)} y_l^{(a)} \quad (34)$$

which can also be obtained from (20) with the help of (6) by the transformation

$$G_{jl} = \sum_i \sum_k T_{ji} G_{ik} T_{kl}^T \quad (35)$$

Equation (35) is in agreement with the general rule of matrix transformation because $T^T = T^{-1}$.* The equation (34) shows that the diagonal elements of G in the y -coordinate system are the ϱ_j of (17),

$$\varrho_j = G_{jj} \quad (36)$$

which are nonnegative and add up to unity. If we require the transformed matrix G_{jl} to be diagonal, we can write, introducing some constants λ_l ,

$$\sum_i \sum_k T_{ji} G_{ik} T_{kl}^{-1} = \lambda_l \delta_{jl} \quad (37)$$

Multiplying T^{-1} from the left, we obtain

$$\sum_{k=1}^n (G_{ik} - \lambda_l \delta_{ik}) T_{kl}^{-1} = 0 \quad (38)$$

Comparison of (25) with (38) shows that we can put

$$\xi_i^{(j)} = T_{ji} \quad (39)$$

* We are using the index symbol i for the coordinate system $\{x_i\}$ and the index symbol j for $\{y_j\}$. Accordingly, G_{ik} refers to the first coordinate system and G_{jl} to the second coordinate system.

The ortho-normal condition of the eigenvectors (33) becomes precisely the condition of orthogonal transformation (7). In other words, the eigenvectors of G provide the coordinate system which diagonalizes G . In fact, we have, in this special coordinate system,

$$G_{jl} = \lambda_j \delta_{jl}, \quad \text{where } \lambda_j = \varrho_j \quad (40)$$

3.3. Basic Theorem of SELFIC

We are now prepared to introduce a theorem which allows us to obtain the optimal coordinate system.

Theorem 3.1. The eigenvectors of G provide an optimal coordinate system in the sense used in the Principle of Minimum Entropy.

Proof. In the foregoing, we started from G_{ik} in the x -coordinate system and obtained the transformed G_{jk} by TGT^T as in (35). But we can now start from G_{jl} of (40) and write G in any other coordinate system as

$$G_{ik} = \sum_{j=1}^n \sum_{l=1}^n U_{ij} G_{jl} U_{lk}^T \quad (41)$$

where U should satisfy the condition of orthogonal transformation (7), i.e.,

$$U^{-1} = U^T \quad (42)$$

If the coordinate system after the transformation coincides with the original x -coordinate system, we should have $U^{-1} = T$, because, by U , we go backward to the transformation expressed by T , but otherwise U should be an arbitrary orthogonal transformation. Theorem 3.1 asserts that

$$-\sum_{i=1}^n \varrho_i \log \varrho_i \geq -\sum_{j=1}^n \varrho_j \log \varrho_j \quad (43)$$

where ϱ_i and ϱ_j are the diagonal elements of G after and before the transformation U , and are connected by

$$\varrho_i = \sum_{j=1}^n A_{ij} \varrho_j \quad (44)$$

where

$$A_{ij} = U_{ij} U_{ji}^T = (U_{ij})^2 \geq 0 \quad (45)$$

because G_{jl} in (41) is $\varrho_j \delta_{jl}$ as in (40), and

$$\varrho_i = G_{ii} \quad \text{and} \quad \varrho_j = G_{jj} \quad (46)$$

The orthogonality condition $U^T = U^{-1}$, (42), is now, due to (45), translated into the double stochasticity of A_{ij} , i.e.,

$$\sum_{j=1}^n A_{ij} = \sum_{i=1}^n A_{ij} = 1 \quad (47)$$

Hence, the theorem will be proved if we can prove the following lemma, which is the simplest type of H -theorem:

Lemma. If

$$\varrho_j \geq 0 \quad \text{and} \quad \sum_{j=1}^n \varrho_j = 1 \quad (48)$$

$$A_{ij} \geq 0 \quad (49)$$

$$\sum_{j=1}^n A_{ij} = 1 \quad (50)$$

$$\sum_{i=1}^n A_{ij} = 1 \quad (51)$$

and

$$\varrho_i = \sum_{j=1}^n A_{ij} \varrho_j \quad (52)$$

then

$$-\sum_{i=1}^n \varrho_i \log \varrho_i \geq -\sum_{j=1}^n \varrho_j \log \varrho_j \quad (53)$$

Proof. It should be noted first that (48) and (49) allow us to interpret A_{ij} as a transition probability from j to i , so that, if the ϱ_j , $j = 1, 2, \dots, n$, are probabilities in the sense of (47), then the ϱ_i , $i = 1, 2, \dots, n$, defined by (52), also become probabilities in the sense that

$$\varrho_i \geq 0 \quad \text{and} \quad \sum_{i=1}^n \varrho_i = 1 \quad (54)$$

On account of the inverse normalization (50), we can say that the transition matrix A_{ij} is doubly stochastic.

Suppose that a real function $g(u)$ of a real argument u is convex upward, i.e.,

$$d^2g(u)/du^2 \leq 0 \quad (55)$$

in a domain $u_0 \leq u \leq u_1$. Assume that u is a stochastic variable, so that $g(u)$ is also a stochastic variable. Let us write the average value of any

stochastic variable by $\langle \cdot \rangle$. Then it is easy to show that

$$g(\langle u \rangle) \geq \langle g(u) \rangle \quad (56)$$

Let us now take as $g(u)$ the function

$$g(u) = -u \log u, \quad g(0) = 0 \quad (57)$$

which satisfies (55) in the domain $0 \leq u \leq 1$. Let u be a stochastic variable which takes value ϱ_j ($0 \leq \varrho_j \leq 1$) with some probability A_j which satisfies

$$A_j \geq 0 \quad \text{and} \quad \sum_{j=1}^n A_j = 1 \quad (58)$$

Then (56) becomes

$$-\left(\sum_{j=1}^n A_j \varrho_j\right) \log \left(\sum_{j=1}^n A_j \varrho_j\right) \geq -\sum_{j=1}^n A_j \varrho_j \log \varrho_j \quad (59)$$

Now, let us introduce a set of n such probability distributions, each distribution being labeled by a new index i , so that A_j will be now denoted by A_{ij} and (58) becomes

$$A_{ij} \geq 0 \quad \text{and} \quad \sum_{j=1}^n A_{ij} = 1 \quad (60)$$

and (59) becomes

$$-\left(\sum_{j=1}^n A_{ij} \varrho_j\right) \log \left(\sum_{j=1}^n A_{ij} \varrho_j\right) \geq -\sum_{j=1}^n A_{ij} \varrho_j \log \varrho_j \quad (61)$$

We now require as an additional condition that

$$\sum_{i=1}^n A_{ij} = 1 \quad (62)$$

Suppose now we sum (61) with respect to i over $i = 1, 2, \dots, n$. The right-hand side simply becomes, on account of (62),

$$-\sum_{j=1}^n \varrho_j \log \varrho_j \quad (63)$$

whereas the left-hand side becomes

$$-\sum_{i=1}^n \varrho_i \log \varrho_i \quad (64)$$

if we write

$$\varrho_i = \sum_{j=1}^n A_{ij} \varrho_j \quad (65)$$

This completes the proof of the lemma. We did not use the condition $\sum_{j=1}^n \varrho_j = 1$, but we assumed $0 \leq \varrho_j \leq 1$, and, on account of (60) and (62), this also entails $0 \leq \varrho_i \leq 1$ and $\sum_{i=1}^n \varrho_i = \sum_{j=1}^n \varrho_j$.

In the case of degeneracy, i.e., if some of the λ 's are multiple roots, the eigenvectors do not determine a coordinate system uniquely. This ambiguity corresponds precisely to the case where the equal sign holds in (53). See reference 2 for a detailed discussion of such a degenerate case. In actual applications, this ambiguity does not cause much trouble because any one of the optimal coordinate systems is good enough for a practical purpose.

3.4. Discriminatory Feature Space and SELFIC

Suppose that we have m nonidentical objects in the ensemble and we take $w^{(a)} = 1/m$. Then, $m\lambda_j$ can be interpreted as the sum of the squared components in the y_j -direction of all the objectvectors in the ensemble, because $m\lambda_j = \sum_a (y_j^{(a)})^2$. Consequently, according to the convention (24), the coordinate axes with higher values of j have less and less total components along their direction. Thus, it would be a natural idea to eliminate those coordinates with higher values of j and retain, say, n^* coordinate axes so that

$$\sum_{j=1}^{n^*} \lambda_j = \sigma \quad (66)$$

This n^* -dimensional subspace contains $(100 \times \sigma)\%$ of the total squared components of all the vectors in the ensemble. We may say that this subspace provides us with a representation of the ensemble with "fidelity" σ . If $w^{(a)}$ is not uniform (i.e., $\neq 1/m$ for some a), we cannot speak in terms of the total squared components but we can still interpret λ_j as the measure of importance of the y_j coordinate and we can define a representation of fidelity σ by (66). If we generalize (66), and write

$$\sum_{i=1}^{n^*} \varrho_i = \sigma \quad (67)$$

for any arbitrary coordinate system, it is easy to show that, for a given val-

ue of n^* , the fidelity σ becomes maximum when we use the optimal coordinate system. See reference 2 for a proof. Since this n^* -dimensional subspace is subtended by the n^* most important coordinates, we may call it a "feature space" of fidelity σ . We have tried this method of automatic feature compression on many actual problems of different natures and found that n^* becomes considerably smaller than n even for a relatively high value of fidelity, such as 0.85 or 0.90.

If $\{x_i^{(a)}\}$ is normalized, $\{y_j^{(a)}\}$ will also be normalized in the full n -dimensional space as in (14). If we now introduce a quantity $\tau^{(a)}$ by

$$\tau^{(a)} = \sum_{j=1}^{n^*} (y_j^{(a)})^2 \quad (68)$$

it will lie in the domain: $0 \leq \tau^{(a)} \leq 1$, and will measure the squared projection of the object vector a on the feature subspace. An object with large $\tau^{(a)}$ is one having a large share of statistically important properties, and an object with small $\tau^{(a)}$ is one having a large amount of statistically unimportant properties. We may thus call this quantity the "representativeness" of object a . By requiring that the representativeness be larger than a certain threshold ($\tau^{(a)} > \theta$), we can extract a small group of representative objects. We call this process REPRES (representative extraction).

It should be noticed that, if we apply this method directly to the collection of object vectors, it is easily expected that the eigenvector corresponding to the largest λ will become a kind of average of all the vectors in the ensemble. When we are interested in bringing out the differences between the vectors, as would be the case when we want to divide the objects into different classes, it is advisable to subtract the average vector of the ensemble from each vector at the outset and then normalize according to (14). Another way of avoiding the average appearing as the first eigenvector would be to replace each vector $\{x_i^{(a)}\}$ by $\{x_i^{(a)'}\}$, which is defined by

$$x_i^{(a)'} = (x_i^{(a)} - c_i) / \left[\sum_{i=1}^n (x_i^{(a)} - c_i)^2 \right]^{1/2} \quad (69)$$

where the constant vector c_i should be determined by the following n equations:

$$\sum_{a=1}^m w^{(a)} (x_i^{(a)} - c_i) / \left[\sum_{i=1}^n (x_i^{(a)} - c_i)^2 \right]^{1/2} = 0 \quad (70)$$

The n^* -dimensional subspace of fidelity σ thus obtained may be called

a “discriminatory feature space.” This is discriminatory in the sense that the property common to all the objects in the entire ensemble is already deleted. This does not exclude the possibility that a feature obtained after this subtraction may become a common feature of a subensemble within the entire ensemble. SELFIC is the name given to this type of selection of discriminatory variables. The idea of representativeness of an object is perfectly tenable also in the SELFIC method. A representative object here is not one with a strong dosis of the common properties of the entire ensemble, but one with a strong dosis of the properties which statistically serve effectively the purpose of discriminating one object from another or one subensemble from another in the ensemble.

The method of SELFIC is useful as a preprocessing for any kind of pattern recognition and clustering. But in the clustering problem, the idea of representative objects becomes particularly useful for the following reason. When we tackle with a clustering problem of any kind, we usually encounter sooner or later a dilemma: Either we start forming clusters from one corner of the entire collection of objects and finish up with a certain cluster structure, but do not know if we obtain the same or a similar cluster structure if we started from another corner; or else we obtain a systematic clustering method which guarantees a unique solution, but the method can work only for a very small number of objects. The method of REPRES combined with the second alternative, however, allows us to surmount the difficulty. By the REPRES method in SELFIC, we extract a small number of representative objects which the systematic clustering method can handle, leading to a unique system of clusters (classes) in which these representative objects are placed. The remaining task of clustering then becomes one of placing the nonrepresentative objects into these classes which are exemplified by the representative objects. This last problem is nothing but a typical pattern recognition problem in which the roles of paradigms are played by the representative objects. This method of reduction of clustering to a problem of pattern recognition through SELFIC-REPRES is explained and illustrated by an application in reference 10.

From the point of view of mathematical formalism, the method of SELFIC has much in common with the so-called factor analysis and with the Karhunen-Loève expansion (^{3,9}). The contribution of the present author lies in the fact that he showed that one comes to this mathematical tool from the explicit purpose of feature compression. In particular, the principle of minimum entropy and the basic theorem 3.1 were introduced (and the latter proven) by him in 1964 in his report to the Aerospace Medical Group (⁸) (also see reference 3).

3.5. Object-Predicate Reciprocity

The computer program for SELFIC involves an $n \times n$ matrix diagonalization, where n is the number of predicate variables. With present-day computers, any program with n larger than, say, 150 becomes considerably more difficult. On the other hand, the number m of objects does not matter very much because we just add new terms in the formation of the matrix (20). This is an asymmetry between m and n arising from the nature of the mathematical computation involved in this method. There is another asymmetry between m and n , which can be found in practice in most of the actual problems of classification with a large value of m . Among the m object vectors there are usually many duplications or at least many similar ones, whereas there are fewer identical or similar predicate variables.

The idea of asymmetry discussed in the above paragraph presupposes a possible symmetry between the object and predicate variables. In fact, there is a fundamental symmetry in our basic mathematical formulation, because the set of values $\{x_i^{(a)}\}$, $i = 1, 2, \dots, n$, $a = 1, 2, \dots, m$, can be considered either a collection of m n -component "object vectors" or a collection of n m -component "predicate vectors." Our explanation in this section so far has been based on the former point of view, but the same thing must have been possible on the basis of the latter point of view (of course, we have to normalize the predicate vectors this time). The idea of exploiting this fundamental symmetry in problems of classification was first suggested in reference 11 and fully described in reference 12.

Taking advantage of this basic symmetry, we can overcome the difficulty in a problem which has an asymmetric structure. When both the number of predicate variables and the number of objects are large, we pick a small number (less than, say, 150) of objects at random because there are often many very similar objects. Then, we can apply the SELFIC method in the reciprocal way, i.e., by interchanging the roles of predicate and object. By this process, we can obtain a limited number n^+ of representative predicates instead of representative objects. By making this number n^+ less than, say, 150, we can now apply the SELFIC in the normal way, reinstating all the objects that existed in the original collection. This will lead to a classification of objects.

In cases in which both approaches, normal and reciprocal, are feasible, it is worthwhile trying both. There is no reason why the two approaches should result in exactly the same classification and there is no reason either why one result should be *a priori* better than the other. They could complement each other.

4. CHARACTERISTIC FEATURE COMPRESSION—CLAFIC

4.1. Class-Feature Space

In pattern recognition problems in the narrow sense of the word (see Section 1.1), we receive during the training period a collection of paradigms of which we know not only the properties (object vectors), but also the class affiliations. In such a case, our attention is naturally drawn to the characteristic features of each class. We are thus primarily interested in those properties which are conspicuous in each class, and then later we can ask ourselves a question as to how these conspicuous features differ from one class to another. The method of CLAFIC is a mathematical implementation of such an idea.

Let $\{x_i^{(A,a)}\}$, $i = 1, 2, \dots, n$, $a = 1, 2, \dots, m_A$, be the set of paradigm object vectors belonging to class A , $A = 1, 2, \dots, v$, where m_A is the number of available paradigms of class A and v is the number of classes. We do not subtract the class-average vector. (Note: $A = 1, 2, \dots$ corresponds to $A_k = \text{I, II, ...}$ of Section 1.) We apply the method of optimal coordinate systems to each set $\{x_i^{(A,a)}\}$ separately and obtain for each A the eigenvectors $\{\xi_i^{(A,j)}\}$, $i = 1, 2, \dots, n$, and $j = 1, 2, \dots, n$, corresponding to eigenvalues

$$\lambda_1^{(A)} \geq \lambda_2^{(A)} \geq \dots \geq \lambda_n^{(A)} \quad (71)$$

Setting a fixed fidelity value σ_0 , we can select $n^{(A)*}$ most important class features,

$$\sum_{i=1}^{n^{(A)*}-1} \lambda_i^{(A)} < \sigma \leq \sum_{i=1}^{n^{(A)*}} \lambda_i^{(A)} \quad (72)$$

The subspace spanned by these $n^{(A)*}$ eigenvectors may be called class-feature subspace. A convenient mathematical tool to express a subspace is the projection operator corresponding to it, which, in the present case, becomes

$$\mathcal{P}_{ik}^{(A)} = \sum_{j=1}^{n^{(A)*}} \xi_i^{(A,j)} \xi_k^{(A,j)} \quad (73)$$

which is a symmetric nonnegative matrix satisfying

$$\mathcal{P}^{(A)} \mathcal{P}^{(A)} = \mathcal{P}^{(A)} \quad (74)$$

$$\text{tr } \mathcal{P}^{(A)} = n^{(A)*} \quad (75)$$

If we apply this matrix to any arbitrary vector $\{x_i\}$, we obtain

$$x_i^{(A)} = \sum_{k=1}^n \mathcal{P}_{ik}^{(A)} x_k \quad (76)$$

which is the projection of $\{x_i\}$ on the subspace of class A . The squared cosine of the angle between the vector $\{x_i\}$ and the subspace A is given by

$$\sum_i x_i x_i^{(A)} = (x^T \mathcal{P}^{(A)} x) \quad (77)$$

This quantity is a good measure of the degree to which the vector $\{x_i\}$ belongs to subspace A , hence, the degree to which the object represented by this vector belongs to class A . It will be unity if the vector lies in the subspace, and zero if the vector is perpendicular to the subspace. If it is larger than the fidelity value σ , the object can be declared to belong to the class with maximum certainty. This representation of class is called CLAFIC, class-featuring information compression.

4.2. Subspace Model Versus Zone Model

In the usual pattern recognition procedure, the representation space is divided into zones (volumes) and each zone is assigned to a class. In the CLAFIC method, a subspace rather than a zone is assigned to a class. It is not generally noticed, but this means a revolutionary departure from the traditional concept of pattern recognition. That the subspace model is very close to reality, at least in a representation space with a relatively large number of dimensions, has been confirmed again and again by myself and my associates with a large variety of examples. It is not unusual that, starting with a number of dimensions of the order of 100 and requiring a fidelity value of 0.9 or so, we land a subspace of dimensions between 5 and 10.

If there are two classes $A = 1$ and 2, and if the two subspaces are diagonal, i.e.,

$$\mathcal{P}^{(1)} \mathcal{P}^{(2)} = 0 \quad (78)$$

then these two subspaces together subtend a subspace of dimension

$$n^* = \text{tr } \mathcal{P}^{(1)} + \text{tr } \mathcal{P}^{(2)} \quad (79)$$

The total dimension n^* is usually still quite small compared with n , and we can achieve considerable economy by reducing everything to this sub-

space. The projection operator of this last subspace is obviously

$$\mathcal{P}^{(1 \cup 2)} = \mathcal{P}^{(1)} + \mathcal{P}^{(2)} \quad (80)$$

If $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$ are not orthogonal, but if they commute with each other,

$$\mathcal{P}^{(1)}\mathcal{P}^{(2)} - \mathcal{P}^{(2)}\mathcal{P}^{(1)} = 0 \quad (81)$$

then there exists a single set of coordinate vectors $\{\xi_i^{(j)}\}$ such that we can express both $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$ in the form (73) with the same set $\{\xi_i^{(j)}\}$, taking, of course, some different terms under the summation. These terms which appear in both summations form the largest common subspace of $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$, expressed by

$$\mathcal{P}^{(1 \cap 2)} = \mathcal{P}^{(1)}\mathcal{P}^{(2)} \quad (82)$$

The total subspace is then expressed by

$$\mathcal{P}^{(1 \cup 2)} = \mathcal{P}^{(1)} + \mathcal{P}^{(2)} - \mathcal{P}^{(1)}\mathcal{P}^{(2)} \quad (83)$$

Of course, the earlier case of orthogonal subspaces is a special case of this commutable case.

When $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$ do not commute, we should consider an operator

$$\mathcal{Q}^{(N)} = \mathcal{P}^{(1)}\mathcal{P}^{(2)} \dots \mathcal{P}^{(1)}\mathcal{P}^{(2)} \quad (84)$$

where N is the number of $\mathcal{P}^{(1)}$ and the number of $\mathcal{P}^{(2)}$ appearing alternately in this product. It is easy to see that the limit

$$\mathcal{Q}^{(\infty)} = \lim_{N \rightarrow \infty} \mathcal{Q}^{(N)} \quad (85)$$

is a projection operator. It was previously pointed out by the author^(2,13) that the space defined by this projection operator is precisely the largest common subspace of two subspaces $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$

$$\mathcal{P}^{(1 \cap 2)} = \mathcal{Q}^{(\infty)} \quad (86)$$

It is easy to see that (86) becomes (82) if $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$ commute. Let us next consider

$$\mathcal{P}^{(N)} = (1 - \mathcal{P}^{(1)})(1 - \mathcal{P}^{(2)}) \dots (1 - \mathcal{P}^{(1)})(1 - \mathcal{P}^{(2)}) \quad (87)$$

where $2N$ is the number of factors appearing on the right side. The total

subspace subtended by the vectors in $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$ is then expressed by the projection operator

$$\mathcal{P}^{(1 \cup 2)} = 1 - \mathcal{P}^{(\infty)} \quad (88)$$

which again reduces to (83) in the commutative case.

4.3. Decision Procedures by Projection and by Entropy

Suppose we have obtained during the training period the class-feature subspaces $\mathcal{P}^{(1)}$ for class 1 and $\mathcal{P}^{(2)}$ for class 2 with fidelity requirement σ [see (72)]. If we receive a new arrival $\{x_i\}$ which we are suppose to classify, we can expect $(x^T \mathcal{P}^{(A)} x)$ to be on the average about σ if it belongs to class A . This gives a decision procedure: we should classify $\{x_i\}$ into class 1 or class 2 according as the first or second inequality holds in the following:

$$\begin{aligned} (x^T \mathcal{P}^{(1)} x) / (x^T \mathcal{P}^{(2)} x) &> \theta \\ (x^T \mathcal{P}^{(1)} x) / (x^T \mathcal{P}^{(2)} x) &< 1/\theta \end{aligned} \quad (89)$$

with a suitable value of $\theta > 1$.

If the overlapping of the two spaces $\mathcal{P}^{(1)}$ and $\mathcal{P}^{(2)}$ is large, $(x^T \mathcal{P}^{(1)} x)$ and $(x^T \mathcal{P}^{(2)} x)$ may become close to each other and the evaluation of the difference could become inaccurate. In such a case, we should subtract the common subspace from each of the class-feature subspace:

$$\mathcal{P}^{(1)*} = \mathcal{P}^{(1)} - \mathcal{P}^{(1 \cap 2)}, \quad \mathcal{P}^{(2)*} = \mathcal{P}^{(2)} - \mathcal{P}^{(1 \cap 2)} \quad (90)$$

The modified class-feature subspaces $\mathcal{P}^{(1)*}$ and $\mathcal{P}^{(2)*}$ are not orthogonal with each other, but have no common subspace any more. Instead of (89), we should then use

$$\begin{aligned} (x^T \mathcal{P}^{(1)*} x) / (x^T \mathcal{P}^{(2)*} x) &> \theta \\ (x^T \mathcal{P}^{(1)*} x) / (x^T \mathcal{P}^{(2)*} x) &< 1/\theta \end{aligned} \quad (91)$$

Since $(x^T \mathcal{P}^{(A)*} x)$ is appreciably smaller than $(x^T \mathcal{P}^{(A)} x)$, we can raise the fidelity value to compensate it and achieve a higher accuracy in the final result.

Another decision procedure exploits the following fact. The eigenvectors $\{\xi_i^{(A,j)}\}$ are such that

$$S = - \sum_j \langle c_j^{(A)*} \rangle \log \langle c_j^{(A)*} \rangle \quad (92)$$

is minimized, where $\langle c_j^{(A)^2} \rangle$ is the average of the square of the components of the object-vectors $\{x_i^{(A,a)}\}$ along the direction $\{\xi_i^{(A,j)}\}$,

$$\langle c_j^{(A)^2} \rangle = \sum_{a=1}^{m_A} w^{(A,a)} (x^{(A,a)T} \cdot \xi^{(A,j)})^2 \quad (93)$$

[see (15)–(19)]. S is minimized in the sense that if we used a coordinate system other than $\{\xi_i^{(A,j)}\}$, it would become larger.

From this fact, we can conjecture that the entropy would be small for each vector of class A if we use the eigenvectors of the same class as the coordinate. The entropy in question for a vector $\{x_i\}$ of unknown class will be defined

$$S_A(x) = - \sum_j c_j^{(A)^2} \log c_j^{(A)^2} \quad (94)$$

where

$$c_j^{(A)} = (x^T \cdot \xi^{(A,j)}) \quad (95)$$

The above conjecture is that, if $\{x_i\}$ belongs to class A , $S_A(x)$ will be relatively small. This suggests the following decision procedure: For a given new arrival $\{x_i\}$, calculate (94) for $A = 1$ and $A = 2$, and place $\{x_i\}$ in the class which makes (94) smaller. This decision rule may not sound well-founded, but it works unexpectedly well in practice (14).

5. IMPLICATIONS OF SUBSPACE MODEL—FUZZY CLASS

5.1. Modular Nondistributive Predicate Lattice

In this section, we try to investigate, from the point of view of logic, the implications of the subspace model of classes we have arrived at. We have seen that paradigm vectors of a class lie with a small average error of $(1 - \sigma)$ in a subspace of relatively small dimensions. We may conveniently divide the paradigms into two categories: pure paradigms and mixed paradigms, the difference being that the pure-paradigm vectors lie entirely within the subspace, whereas the mixed-paradigm vectors have some small components in the direction perpendicular to the subspace.

Class and predicate correspond to each other one to one. For instance, if the class is the collection of numerals “8” handwritten in many different ways, the predicate may be said to signify “8-ness;” if the class is the collection of all red objects, the corresponding predicate is redness. We can also reduce any proposition to a predicate because we can reformulate it

in the form: “the case is such that the proposition is true,” where the clause after “such that” is a predicate.

Let us consider a certain collection of predicates denoted symbolically by $A, B, C, \dots$. Generalizing the fact discovered in the CLAFIC method of pattern recognition, we may justifiably assume that each predicate corresponds to a subspace in some representation space. This subspace may be considered as being subtended by all pure paradigms satisfying the predicate, assuming that we can increase indefinitely the set of paradigms.

A subspace is fully specified by its projection operator. Hence, each predicate A can be mathematically expressed by the projection operator $\mathcal{P}(A)$ of the corresponding subspace. Conversely, we assume that any projection operator can be interpreted as a possible predicate.

The basic logical relation that can exist between two predicates is “implication”: “ A implies B ”, or, “if A , then B ,” or, symbolically, $A \rightarrow B$. This must mean that, if a vector is in $\mathcal{P}(A)$, then it also has to be in $\mathcal{P}(B)$. Such a geometrical relation can be expressed mathematically

$$\mathcal{P}(A) = \mathcal{P}(A)\mathcal{P}(B) = \mathcal{P}(B)\mathcal{P}(A) \quad (96)$$

which should be equivalent to

$$A \rightarrow B \quad (97)$$

Once we have defined implication, the conjunction and disjunction are unambiguously defined. For instance, the conjunction $C = A \cap B$ has to be defined by two conditions: (1) $C \rightarrow A$ and $C \rightarrow B$, (2) if $X \rightarrow A$ and $X \rightarrow B$, then $X \rightarrow C$. It can be shown⁽¹³⁾ that such a C can be expressed by $\mathcal{P}(C)$ which is an infinite product:

$$\mathcal{P}(C) = \mathcal{P}(A \cap B) = \dots \mathcal{P}(A)\mathcal{P}(B)\mathcal{P}(A)\mathcal{P}(B) \dots \quad (98)$$

We had already dealt with such a quantity in (86). The disjunction $C = A \cup B$ can be defined in a similar way as the conjunction, only changing the direction of arrows in (1) and (2) above. As we have seen in (87), this can be given by

$$\begin{aligned} \mathcal{P}(C) &= \mathcal{P}(A \cup B) = 1 - \\ &\dots (1 - \mathcal{P}(A)) (1 - \mathcal{P}(B)) (1 - \mathcal{P}(A)) (1 - \mathcal{P}(B)) \dots \end{aligned} \quad (99)$$

Corresponding to an absurd predicate $\emptyset$ and a tautological predicate $\square$, which we assume to be members of the set of predicates, we take

$$\mathcal{P}(\emptyset) = 0 \quad \text{and} \quad \mathcal{P}(\square) = 1 \quad (100)$$

for we should have $\emptyset \rightarrow A \rightarrow \square$ for any A . The negation $\neg A$ of A must also correspond to a subspace, but it should not contain any vector which is not orthogonal to any vectors of A , because otherwise such a nonorthogonal vector may appear as an impure paradigm of A with some probability. Hence, the negation $\neg A$ of A must be the largest subspace perpendicular to the subspace of A . The projection operator of such a subspace is given by

$$\mathcal{P}(\neg A) = 1 - \mathcal{P}(A) \quad (101)$$

Starting from the original set of predicates ($\emptyset$, $\square$, A , B , C , ...), we augment the set by all the predicates that can be formed from its existing members with the help of the three connectives (conjunction, disjunction, negation).

The definition (101) agrees with the basic laws involving negation, such as the law of double negation, the law of contraposition, the law of contradiction, the law of the excluded middle (see Section 5.2 for more about this.) Also, (98) and (97) combined with (101) agree with de Morgan's law: $\neg(A \cap B) = \neg A \cup \neg B$, $\neg(A \cup B) = \neg A \cap \neg B$. It is easy to see that the definitions (98) and (99) entail the idempotent law [$A \cap A = A$, $A \cup A = A$], commutative law [$A \cap B = B \cap A$, $A \cup B = B \cup A$], associative law [$A \cap (B \cap C) = (A \cap B) \cap C$, $A \cup (B \cup C) = (A \cup B) \cup C$], and absorptive law [$A \cup (A \cap C) = A$, $A \cap (A \cup C) = A$]. This guarantees that the set of predicates form a complemented lattice. This is a necessary condition for us to build a propositional calculus on the set of predicates.

If, in addition, we had the distributive law

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C) \quad (102)$$

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C) \quad (103)$$

the lattice would become Boolean (or distributive) and the ensuing propositional calculus will become the ordinary Boolean logic. However, the distributive law does not follow from the set in which there is originally only implication, as it is the case with us here. As a matter of fact, we can easily produce a counter example to the distributive law in our mathematical definition of predicates.

Suppose, for simplicity, that the entire representation space is three-dimensional. Let A be a one-dimensional subspace which is a straight line $y = \frac{1}{3}x$ lying in the $x y$ -plane; let B be a one-dimensional subspace which is a straight line $y = \frac{2}{3}x$ lying in the $x y$ -plane; and let C be the $z x$ -plane.

Then, $B \cap C$ is the common space of B and C which is $\emptyset$; hence the left-hand side of (102), $A \cup (B \cap C)$, becomes A , which is the straight line $y = \frac{1}{3}x$. On the right side of (102), the first factor $(A \cup B)$ will become the x y -plane, and $A \cup C$ will become the entire space $\square$. Hence, the right-hand side of (102), becomes the x y -plane. Thus, the left-hand side ($y = \frac{1}{3}x$) and the right-hand side ($\square$) are not the same, showing that the distributive law breaks down.

Suppose now we move gradually the straight line A until it coincides with the x -axis, so that $A \rightarrow C$. Under this condition, the left-hand side becomes the x -axis instead of $y = \frac{1}{3}x$. On the right side, $A \cup B$ is still the x y -plane, but $(A \cup C)$ becomes C , which is the z x -plane. As a result, the right-hand side becomes the intersection of the x y -plane with the z x -plane, which is the x -axis. In this case, the distributive law (102) holds. This is an example of the general rule that

$$\begin{aligned} &\text{if } A \rightarrow C \\ &\text{then } A \cup (B \cap C) = (A \cup C) \cap (A \cup B) \\ &\qquad\qquad = (A \cup B) \cap C \end{aligned} \tag{104}$$

We do not prove this rule in general here for economy of space, but this is a general law which holds when the lattice in question is a lattice made of subspaces. The law is called the “modular law” which amounts to admitting the distributive law (102) only under the condition $A \rightarrow C$, or to admitting the distributive law (103) only under the condition $C \rightarrow A$. The propositional calculus which ensues from the lattice with the modular law and that ensues from the lattice with the distributive law are different, and the former is a broader framework than the latter.

5.2. Implications of the New Logic

The substitution of the modular law for the distributive law changes our logic fundamentally, and we want to point out a few conspicuous points of difference in this section. Before doing that, however, it should be clearly understood that a Boolean lattice is a special case of a modular lattice, but a modular lattice is not a special case of a Boolean lattice. Hence, within a modular lattice, there could be a subset of predicates which form a Boolean lattice. In fact, in the case of a modular lattice formed by subspaces, consider a set of rectangular coordinates and the set of subspaces subtended by some of the coordinate axes. This subset of subspaces can be shown to satisfy the distributive law. The fact that the Boolean logic is upheld in a

subset of predicates justifies the fact that we usually use Boolean logic in many scientific problems and also the fact that we are using a Boolean language as the metalanguage with which we are explaining the non-Boolean logic.

The first surprising fact about the modular logic is that $\neg(A \cup B) = \square$ (which is often written $A \supset B = \square$) does not entail $A \rightarrow B$, although, if $A \rightarrow B$, then $\neg(A \cup B) = \square$. In a three-dimensional space, two vectors A and B both lying in the x y -plane satisfy $\neg(A \cup B) = \square$, unless A and B are perpendicular. More generally, we can show that $A \rightarrow B$ is equivalent to each of $A = A \cap B$ and $B = A \cup B$, and $A \rightarrow B$ entails $\neg(A \cup B) = \square$ and $A \cap \neg B = \emptyset$, but $A \rightarrow B$ ensues from each of $\neg(A \cup B) = \square$ and $A \cap \neg B = \emptyset$ only in the presence of the distributive law.

The so-called intuitionist logic refuses to identify $\neg(\neg A)$ with A , but the modular logic has no difference from the ordinary logic in this respect. The many valued logic denies the law of the excluded middle because a proposition takes more than two values of truth and falsehood. In the modular logic, the matter is a little more subtle in this respect. If A and B are mutually the negation of each other, then $A \cap B = \emptyset$ (law of contradiction) and $A \cup B = \square$ (law of the excluded middle) are true, but if two predicates A and B are only known to satisfy $A \cap B = \emptyset$ and $A \cup B = \square$, we cannot decide that A and B are the negation of each other except in the presence of the distributive law.

The basic difference stems from the following situation: The conjunction in our broader logic is not different from the customary conjunction, because the set of pure paradigms of $A \cap B$ is the set-theoretical conjunction of the set of pure paradigms of A and the set of pure paradigms of B . But the disjunction $A \cup B$ in our case is a much larger set than the set-theoretical disjunction of the set of pure paradigms of A and the set of pure paradigms of B . In fact, the pure paradigm vectors of $A \cup B$ consist of all possible linear combinations of a pure paradigm vector from A and a pure paradigm vector from B . Thus, $A \cup B$ contains new additional vectors which are neither a pure paradigm of class A nor of class B . We may, however, argue for taking them also as pure paradigms of $A \cup B$ on the following ground. Suppose that the subspace A and the subspace B are perpendicular to each other, and let x be a vector which is a linear combination of a vector in subspace A and a vector in subspace B . Then, it is obvious that we have

$$(x^T \cdot \mathcal{P}(A)x) + (x^T \cdot \mathcal{P}(B)x) = 1 \quad (105)$$

If we interpret $(x^T \cdot \mathcal{P}(A)x)$ as a kind of probability, formula (105) can

be interpreted as meaning that the probability of x belonging to A or B is unity. If A and B are not perpendicular, we can take subspace A and its complementary subspace of A within the space $A \cup B$ and argue in the same way.

In a similar way, the relation between A and $\neg A$ is slightly different from the ordinary case. In the ordinary case, any object must belong either to A or to $\neg A$. But, in our case, an object vector need not be a pure paradigm of either A or $\neg A$. In the whole space, there are a large number of vectors which do not belong to either subspace A or subspace $\neg A$. (If A is the x - y -plane in the three-dimensional space, $\neg A$ is the z -axis.) This, from our point of view, does not matter, because the “probability” of any vector “belonging” to A or to $\neg A$ is unity. But this situation gives rise to a good many unexpected situations in our logic.

5.3. Fuzzy Class

The basic difference between the zone model and subspace model of class can be formulated as follows: Let a be an object and A a class. In the zone model, an object must either belong or not belong to class A . Hence, introducing the characteristic function $f(A; a)$, we can describe the situation by

$$f(A; a) = 0 \quad \text{or} \quad 1 \quad (106)$$

In the subspace model, we have only the “probability” $(x^T; \mathcal{P}(A)x)$, hence we can describe the situation by

$$0 \leq f(A; a) \leq 1 \quad (107)$$

by regarding $(x^T \cdot \mathcal{P}(A)x)$ as the generalization of $f(A; a)$.

It is very easy to derive all the formulas of the Boolean logic for the A 's from (106). To derive the modular logic from (107), we have to add a few more additional postulates. In our present derivation, we have a subspace representation of predicates from the outset. As a result, the modular logic follows without any additional postulates.

The derivation of the modular logic starting from (107) is meaningful because the desirability of a continuous generalization (107) of the characteristic function (106) suggests itself in various contexts. References 16 and 19 explain various motivations for which we are tempted to investigate the consequences of (107). One which may be of interest is the idea of “fuzzy set” put forward by Zadeh⁽¹⁵⁾. He started with an expression like $f(A; a)$ of (107) which he called the membership function, and which should meas-

ure the degree to which an object belongs to a set. Unfortunately, his idea was not developed too far because he assumed that the concepts of set, conjunction, and disjunction were all known, and he tried to fit the membership function into this ready-made framework instead of reformulating these concepts anew as in this chapter. The earliest paper which introduced the concept of modular logic is the article by Birkoff and von Neumann (¹⁷) which tried to interpret quantum mechanics in the framework of a new type of logic. The author suggested on several occasions (^{2,13,16,18,19}) that this new logic may find applications in fields other than quantum physics. In the applications envisioned previously, the act of observation was assumed to change the state of the observed system. In the application we studied in the present chapter, this assumption was not made. In quantum mechanics, the result of observation is always yes or no and the $f(A; a)$ measures the relative frequency of obtaining the yes answer. After we have obtained the yes answer, the state of the object becomes such that if we repeat the same observation, the second observation will give yes with probability one. In our present application to pattern recognition, we cannot borrow this interpretation of the f -function because we cannot assume the state of the object changes after observation. Our theory of subspace logic in pattern recognition should not be considered as complete and consistent unless an objection-free interpretation of the f -function is found.

The breakdown of the ordinary Boolean logic hinted at in this section should not be taken too literally, because vectors of one class lie more or less in a subspace but not all vectors of the subspace correspond to actual members of the class. Yet, one may be justified to doubt the unconditional and universal validity of the Boolean logic even in fields other than atomic physics.

REFERENCES

1. S. Watanabe, in "Information and Prediction in Science," (S. Dockx and P. Bernays, eds.), p. 39, Academic Press, New York (1965).
2. S. Watanabe, "Knowing and Guessing," John Wiley and Sons, New York (1969).
3. S. Watanabe, "Trans. Fourth Prague Conf. on Information Theory, Statistical Decision Function, 1965", Czechoslovak Academy of Sciences, Prague (1967).
4. S. Watanabe, in "Methodologies of Pattern Recognition," (S. Watanabe, ed.), Academic Press, New York (1969).
5. S. Watanabe, *IBM J. Res. Develop.* **4**, 208 (1960).
6. E. Wong, in "Methodologies of Pattern Recognition," (S. Watanabe, ed.), Academic Press, New York (1969).
7. H. Genchi, K. Mori, Sadakazu Watanabe, and S. Katsuragi, Recognition of hand-written numerical characters for automatic letter sorting, *IEEE Proc.* **56**, 1292 (1968).

8. S. Watanabe, Annual Report to Aerospace Medical Division, A.F. Systems Command, 1963-64.
9. S. Watanabe, in "Cybernetic Problems in Bionics, Bionics Symposium 1966," (H. L. Oestreicher and D. R. Moore, eds.), Gordon and Breach, London (1968).
10. T. Kaminuma, T. Takekawa, and S. Watanabe, Reduction of clustering to pattern recognition, *Journal of Pattern Recognition* **1**, 195 (1969).
11. S. Watanabe, *Information and Control* **4**, 291 (1961).
12. S. Watanabe, in "Proc. IFIP Congress 1968, Edinburgh," North-Holland, Amsterdam.
13. S. Watanabe, *Progr. Theoret. Phys. Suppl.*, Nos. 37 & 38, p. 350 (1966).
14. S. Watanabe, in "Computer and Information Science," (J. Tou, ed.), Vol II, p. 91, Academic Press, New York (1966)
15. L. A. Zadeh, *Information and Control* **8** (3), 338 (1965).
16. S. Watanabe, in "Proc. Hawaiian Conf. on Philosophical Problems in Psychology 1968," in press.
17. G. Birkoff and J. von Neumann, *Ann. Math., 2nd Ser.* **37**, 823 (1936).
18. S. Watanabe, *Synthese* **13**, 261 (1961).
19. S. Watanabe, *Information and Control*, **15** (1), 1 (1969).