This paper is concerned with application of linear decision functions to the pattern identification problem and describes an experimental pattern recognition system for the magnetic ink character font now used in the banking industry.

The system is based on a linear decision function determined by means of a variant of an "adaptive training" technique due to Rosenblatt.

The system has been partially implemented (in part, through simulation with aid of a digital computer and, in part, by hardware) and experimental results in using the system are reported.

A pattern identification system using linear decision functions

by J. S. Griffin, Jr., J. H. King, Jr., and C. J. Tunis

The first part of the paper reviews some of the previous work on the pattern identification problem. The second part discusses the technique employed to determine a suitable linear decision function. Finally, an experimental system and its implementation are described, and results obtained in testing are reported.

Introduction to the pattern identification problem

identification systems

Every pattern identification systems consists of two fundamental parts: a *transducer*, which senses the patterns to be identified and converts the information acquired into electrical signals; and a *processor*, which accepts these signals and by some means interprets them so as to achieve the required identification. There may, of course, be other parts to the system. For example, the system might include: a device for presenting the patterns to the transducer, such as a paper transport; or a device which utilizes the information provided the processor, such as a set of gates which direct documents into bins according to the particular pattern identified. But we shall not be concerned with such peripheral apparatus.

The patterns to be identified could consist, for instance, of a family of characters printed on paper, or a vocabulary of spoken words. In the former case the transducer might consist of a lens system and a means of measuring the darkness of various parts of the resulting image; or the characters might be printed in

magnetic ink, and the transducer could then consist of one or another of the various kinds of magnetic reading heads. In the case of speech signals, the transducer could be simply a microphone.

The processor accepts signals that are produced by the transducer when a pattern is present and must extract enough information to correctly identify the pattern (with a high degree of probability). In the case of printed characters, the first step might be to periodically sample the output from the transducer and to quantize the result into two levels in such a way as to produce a binary matrix which is a direct electrical image of the character. The matrix could be interpreted in many ways, including: those based on correlation techniques, searching for the presence or absence of certain critical features, or by the linear method reported in this paper. Of course, there are many other ways to construct a processor for printed characters. In the case of speech signals, the processor often consists of a filter bank which determines the energy present at various frequencies, and some means of analyzing the resulting pattern.

example of an identification system

An early example of a pattern identification system was described by Eldredge, Kamphoefner, and Wendt[1,2,3] and published under the acronym ERMA. In this instance the patterns to be identified were the ten digits and four special symbols of a specially designed font. These were printed with an ink containing iron ferrite particles, so that upon being magnetized each character acquired a field which was peculiar to the class to which it belonged. The transducer was a suitably designed magnetic reading head; a magnetized character, upon being passed under this reading head, caused the production of an electrical signal which went to the processor for interpretation. The processor operated essentially as follows. The signal from the read head was sampled at n successive points in time resulting in values x_1, x_2, $\cdots$, x_n. Let these numbers be the components of the vector X. Let the values which would be produced by a perfect pattern from the ith class be w_1^i, w_2^i, $\cdots$, w_n^i, and let these be the components of the vector W^i; here i will evidently run from 1 to 14. The processor identified the pattern as belonging to the ith class provided

$$\theta W^i \cdot X > W^i \cdot X \qquad \text{for all} \quad j \neq i, \tag{1}$$

where θ was a fixed value lying between 0 and 1; otherwise the processor responded that identification was not possible.

Upon recalling the relation between the dot product of two vectors and the cosine of their included angle, one realizes that the inequalities (1) describe a region of n-dimensional space roughly in the shape of a cone, or more nearly a prism, with vertex at the origin. Included in this region are the standard vector $W^{(i)}$ associated with the ith pattern, together with almost all of the vectors which arise from patterns belonging to the ith class. The regions associated with the various classes are, of

course, non-overlapping, and in fact do not exhaust the whole space. Patterns which produce signals whose vectors do not lie inside any of these regions are, of course, not identified. Note that by decreasing θ these regions could be diminished, the likelihood of announcing an incorrect identification being thereby decreased, but at the expense of increasing the regions corresponding to non-identification. This, in turn, would cause an increase in the rejection rate.

statistical decision theory

Soon after the appearance of the Eldredge, Kamphoefner, and Wendt papers, C. K. Chow[4] observed that the task of the processor could be stated as a problem in statistical decision theory. Chow's analysis may be summarized as follows. He noted that the signals which arise from the presentation of patterns to the transducer could, after certain preliminary transformations in the processor, be regarded as points of a *measurement space.* The consequent identification problem was to determine which pattern was presented to the transducer, given that it had generated a certain point in the measurement space. Hence, the operation of the processor could be represented by a decision rule, i.e., by an assignment to each point of the measurement space either one member of the class of patterns or the statement "no identification is possible." Chow postulated that with each class of patterns there was associated a probability distribution on the measurement space, this distribution being a description of the likelihood of occurrence of a given point in the measurement space upon presentation to the transducer of a member of its corresponding class of patterns. For example, in the system described above, the preliminary transformation consists in sampling the waveform from the transducer at n points, and the measurement space is an n-dimensional vector space. The decision rule has already been stated; namely, for each i it assigns the ith class to each point of the region defined by the inequalities (1) and the statement "no identification is possible" to all other points. Whatever its exact nature, the probability distribution associated with the ith class of patterns is evidently concentrated within this same region, for otherwise this decision rule would not be effective. Chow also postulated that the result of each possible decision (correct identification, erroneous identification, and failure to make any identification) could be evaluated on a unit cost basis. Specifically, suppose m classes are to be identified, say $S_1, S_2, \cdots, S_m$, and let us assign the cost c_{ij} to the decision "a member of S_i is identified as belong to S_j". Then c_{ii} is the cost (perhaps negative) of correctly identifying a member of S_i, whereas if $i \neq j$ then c_{ij} is the cost of incorrectly identifying a member of S_i as a member of S_j. Let c_{i0} be the cost of a failure to make any identification when the pattern presented belongs to S_i. In general, of course, if $i \neq j$ and $i \neq 0$ then $c_{ij} > c_{i0} > c_{ii}$. This corresponds to the usual notion of utility in statistical decision theory.

For purposes of calculation, any particular decision rule can

be represented as follows. If $1 \leq i \leq m$ and X is any point of measurement space, let

$$\delta_i(X) = \begin{cases} 1 & \text{if the } i\text{th class is assigned to } X \\ 0 & \text{otherwise} \end{cases}$$

and similarly let $\delta_0(X)$ take the value 1 if there is assigned to X the statement "no identification is possible" and the value 0 elsewhere. Finally we let p_i be the probability of occurrence of the ith pattern, i.e., the relative frequency with which members of S_i are presented to the transducer. Using this convention, Chow calculated the expected unit cost of operation of this system due to identifying the ith character as the jth to be

$$a_{ij} = \int_M \beta_i(X) c_{ij}\ \delta_j(X)\ dX$$

where M is the measurement space and β_i is the probability distribution on M associated with the ith pattern. The value $j = 0$ is, of course, to be interpreted as the average cost of failure to make any identification when the ith character appears. It follows that the total average cost of operation of the system will be

$$A = \sum_{j=0}^{m} \sum_{i=1}^{m} p_i a_{ij}.$$

Now

$$A = A_0 + A_1$$

where

$$A_0 = \sum_{i=1}^{m} p_i c_{i0} \text{ and } A_1 = \int_M \sum_{j=0}^{m} Z_j(X)\ \delta_j(X)\ dX$$

where

$$Z_0(X) = 0 \text{ and } Z_j(X) = \sum_{i=1}^{m} (c_{ij} - c_{i0}) p_i \beta_i(X)$$

for $1 \leq j \leq m$. The quantities $Z_j(X)$ may be interpreted as measuring the excess of the cost of identifying a pattern which gives rise to the point X of measurement space as belonging to S_j over the cost of failure to make any identification. Chow observed that the total average cost A may be minimized by associating with X the class S_j for which $Z_j(X)$ is least. He let

$$\delta_j(X) = 1$$

if

$$Z_j(X) \leq Z_i(X) \qquad \text{for all} \quad i \neq j$$

(ties are decided arbitrarily).

It may be noted that the particular decision function which Chow obtained may be described as *optimum*, in the sense that it minimizes the cost of operation for a fixed relation between the patterns and the measurement space; and the processor, insofar

as it implements this decision function, may also be called optimum. However, this adjective cannot be applied to the transducer or to that part of the processor whose function is to convert the signals from the transducer into points of the measurement space. It is mainly a matter of experimentation to select adequate transducers and to properly extract information from their output signals. Other limitations of Chow's analysis include: the unit cost assumption is not always tenable; and there may be dependences among the successive patterns of a sequence, as when a self-checking account number or a fixed format for control characters is used. And finally, as Chow himself remarked, even if the probabilities of occurrence of the various patterns and the distributions which they generate are accurately known, it may still be very difficult to implement the optimum decision functions given by this algorithm.

linear decision functions

Some kinds of decision functions happen to be quite convenient to implement, and it has proved expedient to use certain of these even when they bear little relation to the optimum decision functions in Chow's sense. Acceptable performance generally has to be achieved by incurring costs elsewhere in the system, e.g., by using better transducers and preliminary processing, or in some cases by controlling the input patterns; but this kind of trade-off is familiar in systems design.

The work to be reported here centers around the so-called *linear decision function*, a broad discussion of which has been given by Highleyman.[5,6] In simplest terms, the measurement space is taken to be a vector space, say, of dimension n, and a linear decision function is any partitioning of the space by one or more hyperplanes (each of dimension $n - 1$). The question, in which region of the partition does a given vector lie, evidently can be reduced to the question, on which side of each hyperplane does this vector lie. The utility of this notion is based first on the ease with which a mechanism for answering this latter question can be constructed.

Indeed, a typical implementation is by means of a current summing network. Suppose that n measurements are made on the signal from the transducer resulting in voltages on n lines having the values $v_1, v_2, \cdots, v_n$. The vector having these numbers as components will be designated by V. If the lines are connected through resistors to a current measuring device, then the current I which is observed to flow will be $g_1v_1 + g_2v_2 + \cdots + g_nv_n$, where g_i is the conductance (reciprocal of the resistance) of the ith resistor. One may then determine whether the vector V lies on one or the other side of the hyperplane with equation

$$g_1v_1 + g_2v_2 + \cdots g_nv_n - t = 0$$

by noting whether the current I exceeds or is less than t. Negative conductances may be implemented by inverting the corresponding input voltages. Thus, one such network as this will be required for each hyperplane involved in the linear decision function.

Obviously the effectiveness of a linear decision function in identifying a given family of patterns is contingent upon the possibility of specifying an adequate linear decision function in terms of an economically reasonable number of hyperplanes. For example, we have found it feasible to use one hyperplane to separate the signals arising from patterns of any one category from the signals arising from patterns from all other categories. This means that, altogether, there would be as many hyperplanes as there were classes of patterns.

A second important attribute of linear decision functions is the ease with which suitable hyperplanes can often be found. This attribute will become apparent in the following section devoted to finding a particular decision function.

Determination of a suitable linear decision function

general approach

The particular method to be described in this section is a variant of the adaptive training or programmed error correction technique used by Rosenblatt in his PERCEPTRON experiments.[7,8]

Suppose that one has a family of patterns to be identified and that a transducer together with a preliminary processing has been decided upon, so that the presentation of a pattern to the transducer will produce a known vector in measurement space. In the following sections, we address the problem of determining an appropriate linear decision function. Later in the paper we give an explicit description of the algorithm or training procedure for the simplest possible case, namely when there are only two classes of patterns to be identified and the separating hyperplane may be presumed to pass through the origin of measurement space. We will also show how to modify this procedure so as to give two parallel hyperplanes placed symmetrically about the origin; this allows for a zone of indecision, i.e., a rejection region, and may incidentally shorten the length of the training procedure required. A further modification which frees these planes from their special relation to the origin will be described, as well as an extension of these techniques to provide for the identification of more than two classes of patterns.

The effectiveness of these procedures will depend primarily on the distribution of the images of the various patterns in measurement space. Generally speaking, if the vectors produced by patterns from S_1 are concentrated in a region R_1, and those produced by S_2 are concentrated in a region R_2, and if there is a hyperplane which lies between R_1 and R_2, then this training procedure may be expected to produce a satisfactory decision rule. Thus, the transducer and the preliminary part of the processor must be designed to meet this condition. Failure to obtain a satisfactory decision rule after a reasonably lengthy training procedure would suggest that the design should be reconsidered. Theoretical arguments have been adduced to justify this position, but our view is mainly heuristic: it has turned out to be practical

to design pattern identification systems using this approach.

two-class identification algorithm

Suppose initially that there are just two classes of patterns to be identified, say S_1 and S_2. We look for a linear decision function which will suffice to distinguish members of these two classes. We seek a vector W such that if a vector X is produced by the presentation of a pattern from the class S_1 then (with a high degree of probability)

$$X \cdot W > 0, \tag{2}$$

whereas if X is produced by a member of S_2 then

$$X \cdot W < 0. \tag{3}$$

If such a vector can be found, then an unknown pattern will be identified as belonging to S_1 or S_2 according to the following decision rule: if the vector X produced in measurement space by the pattern satisfies condition (2) then the pattern is identified as belonging to S_1; if X satisfies (3) then the pattern is identified as belonging to S_2; if neither of these conditions is satisfied, i.e., if $X \cdot W = 0$, then no decision is rendered (or, as we say, the pattern is *rejected*). Speaking geometrically, we will have a hyperplane passing through the origin, with almost all of the vectors produced by members of S_1 lying on one side of it, and with almost all of those produced by S_2 lying on the other.

We attempt to find W by a trial-and-error technique. Let $p_1, p_2, \cdots, p_k$ be a sequence of patterns, some from S_1 and the remainder from S_2, and let $X_1, X_2, \cdots, X_k$ be the sequence of corresponding vectors arising in measurement space from the presentation of these patterns to the transducer. Let T_1 be any vector (typically T_1 is taken to be the zero vector). We define a sequence of vectors $T_2, T_3, \cdots, T_{k+1}$ iteratively, as follows:

$$\text{if } p_i \text{ is from } S_1 \text{ and } T_i \cdot X_i > 0 \text{ then } T_{i+1} = T_i \tag{4a}$$

$$\text{if } p_i \text{ is from } S_1 \text{ but } T_i \cdot X_i \leq 0 \text{ then } T_{i+1} = T_i + X_i \tag{4b}$$

$$\text{if } p_i \text{ is from } S_2 \text{ and } T_i \cdot X_i < 0 \text{ then } T_{i+1} = T_i \tag{4c}$$

$$\text{if } p_i \text{ is from } S_2 \text{ but } T_i \cdot X_i \geq 0 \text{ then } T_{i+1} = T_i - X_i. \tag{4d}$$

In other words, if the vector T_i behaves as desired with regard to the ith pattern, then it is left unchanged (statements (4a) and (4c)); but, if not, then it is corrected (statements (4b) and (4d)). That statements (4b) and (4d) do in fact represent corrections is clear: if, for example, p_i is from S_1 but

$$T_i \cdot X_i < 0$$

then

$$T_{i+1} \cdot X_i = T_i \cdot X_i + X_i \cdot X_i > T_i \cdot X_i$$

so that T_{i+1} is an improvement, at least as far as the ith pattern is concerned. The last pattern in this sequence, namely T_{k+1}, is a tentative choice for W.

Such procedures as this are frequently described in anthropomorphic terms: we speak of the procedure as a *training routine*, of the statements (4) as *training rules*, and of the processor as being *trained* by the application of these rules.

This process can be expected to converge (i.e., for some $k < \infty$, $T_{i+1} = T_i$ for all $i > k$) only under the special condition that all X_i from S_1 may be separated from all X_i from S_2 by a hyperplane passing through the origin. Generally, this condition is not met, but perhaps a majority of the X_i from S_1 may be separated from the majority of the X_i from S_2 by some hyperplane, $W \cdot X = 0$, when k is taken sufficiently large. The question here is obviously not whether we can obtain convergence in the strict sense but, rather, how many iterations of the *training sample* are necessary in order to guarantee that no substantial improvement will result with further iterations. At present, this question remains largely unanswered in the general case even though there are several proofs of strict convergence for a finite number of iterations when S_1 and S_2 are linearly separable.[7,9,10]

There is no a priori reason to believe that the choice $W = T_{k+1}$ is acceptable, i.e., to believe that the above decision rule with T_{k+1} substituted for W will represent a processor with satisfactory performance. The next step, therefore, is to estimate the frequency with which errors and rejections will occur if the choice $W = T_{k+1}$ is made. This estimate is made by testing the performance, for a particular choice of W, on the training sequence which is presumed to be representative. (By *error* we understand a substitution, i.e., the identification of a pattern as belonging to one class when in fact it belongs to another.) If the choice $W = T_{k+1}$ does not prove to be acceptable, then one may augment the sequence of patterns and continue the training procedure; or perhaps one will decide to abandon the search for this particular W.

Once a satisfactory plane is found, it can be implemented using a circuit of the type described later. The output of this circuit can be quantized to two levels, say 0 and 1, so that e.g., a 0 will be interpreted as indicating that a member of S_1 is present whereas a 1 will indicate a member of S_2.

A simple but significant improvement results from choosing a positive number d and then replacing the training rules (4) by:

$$\text{if } p_i \text{ is from } S_1 \text{ and } T_i \cdot X_i > d \text{ then } T_{i+1} = T_i \tag{5a}$$

$$\text{if } p_i \text{ is from } S_1 \text{ but } T_i \cdot X_i \leq d \text{ then } T_{i+1} = T_i + X_i \tag{5b}$$

$$\text{if } p_i \text{ is from } S_2 \text{ and } T_i \cdot X_i < -d \text{ then } T_{i+1} = T_i \tag{5c}$$

$$\text{if } p_i \text{ is from } S_2 \text{ but } T_i \cdot X_i \geq -d \text{ then } T_{i+1} = T_i - X_i. \tag{5d}$$

Correspondingly one might modify the decision rule to read: if the vector V produced by a pattern satisfies the condition

$$W \cdot X > d \tag{6}$$

then the pattern is identified as belonging to S_1; if the vector produced satisfies the condition

$$W \cdot X < -d \tag{7}$$

then the pattern is identified as belonging to S_2; and if neither of these conditions is satisfied then the pattern is rejected.

This decision rule can be visualized in terms of a pair of hyperplanes placed symmetrically about the origin; between them lies the rejection region, and of the other two regions, one is identified with S_1 and the other with S_2. A moment's reflection will show that these training rules may be understood similarly: they represent an attempt to place a pair of parallel hyperplanes between the regions in which the images of patterns from the two classes are concentrated with the proviso that as the training routine progresses, the vectors T_i tend to get longer, so that in effect the two hyperplanes drift toward the origin. Thus there is a relation between the choice of d and a suitable length of the training routine. Intuitively what one expects is for the necessary length of the training sequence to increase with increase of d. Also, by choice of a suitable large value for d the ratio $|d|/|W|$ will be maximized, i.e., the relative separation will be maximized. But the main point here is that by using two hyperplanes one may expect to get a better fit.

Finally, we note that the decision rule may be further modified: one could replace the inequalities (6) and (7) by

$$W \cdot X > \theta d \tag{8}$$

and

$$W \cdot X < -\theta d \tag{9}$$

respectively, where generally θ is chosen between 0 and 1. Note that the effect of increasing θ would be to decrease substitution errors but at the expense of increasing the number of rejections. This, of course, presupposes that S_1 and S_2 may be "separated", in the large, by a linear boundary passing through the origin. The general case where this condition is not met may be handled by one further simple modification explained below.

It is also easy and worthwhile to free these hyperplanes from their peculiar relation to the origin; that is, we can find and use a single hyperplane which need not pass through the origin, or a pair of parallel hyperplanes which are not necessarily symmetrically placed about the origin. The simplest way to accomplish this is to append a fictitious component to the vectors produced by the presentation of patterns to the transducer, and to always take this component to have the value 1. The training procedures described earlier in the paper may be used to produce a vector, the last component of which may be taken to be the constant term in the equation of the desired hyperplane.

To be more explicit, suppose that we are looking for a single

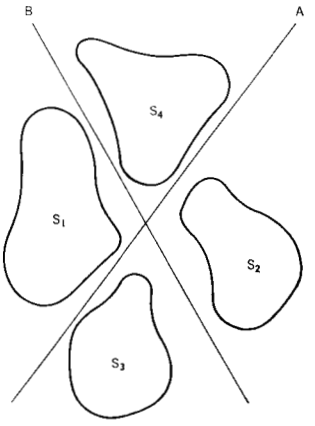
hyperplane and that we have the sequence $X_1, X_2, \cdots, X_k$ of vectors in measurement space, as before. Let the dimension of measurement space be n, and let the sequence $X_1', X_2', \cdots, X_k'$ of $(n + 1)$-dimensional vectors be defined as follows: for each index i, the first n components of X_i' are the components of X_i, and the $(n + 1)$st component of X_i' is 1.

We now define a sequence $T_1, T_2, \cdots, T_k$ by the training rules stated earlier in the paper, but with X_i' replacing X_i. Finally, the vector T_k is obtained and we define W to be the n-dimensional vector whose components are the first n components of T_k, and we let t be the $(n + 1)$st component of T_k.

We now use the following decision rule: if the vector X results from the presentation of a pattern to the transducer, then X is identified as having belonged to S_1, to S_2, or is rejected, according to whether $W \cdot X + t$ is positive, negative, or zero.

The treatment of a pair of parallel hyperplanes may be similarly modified.

Figure 1 An example in two-dimensional measurement space



m-class identification algorithm

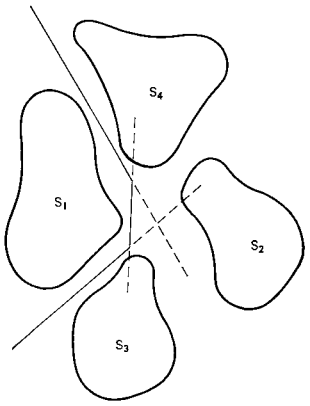
Ordinarily in pattern identification work one must deal with several distinct classes of patterns rather than just two classes. If there are m classes, then one may dichotomize the family of classes p times, where p is the least integer which is as large as $\log_2 m$; the identification of a pattern could then consist in the determination of which half of each of these dichotomies the class containing the relevant pattern belonged to (in other words, only p bits of information are required to specify one object out of m). Interpreting this remark in terms of measurement space, we see that in principle it is possible to use just p hyperplanes to identify a pattern as having come from one of m classes, subject of course to the requirement that the regions in which the images of various classes are concentrated be well spread out in measurement space. However, it has not proved to be practical to implement so economical a scheme as this because we do not know of a simple way to recognize which dichotomies of a family of classes of patterns can be implemented with a hyperplane in measurement space.

This has been called the coding assignment problem; the essential difficulty is illustrated in Figure 1, which is intended to suggest the regions of concentration of the measurements in a two-dimensional measurement space arising from each of four classes of patterns, say S_1, S_2, S_3, and S_4. It is evident that if we group S_1 and S_4 together into one class and S_2 and S_3 together into another, then we may expect the training procedure to yield the hyperplane (line) A. We would obtain therefrom an assignment, say, of 0 to members of S_1 and S_4 and of 1 to members of S_2 and S_3. If next we take S_2 and S_4 to form one class and S_1 and S_3 to form the other, then we should arrive at the hyperplane B and the assignment of 0 to members of S_2 and S_4 and 1 to members of S_1 and S_3. On taking these two together, we would identify patterns according to the scheme shown in Table 1. But, if we had the misfortune to put S_1 and S_2 together into

Table 1

00	S_4
01	S_1
10	S_2
11	S_3

Figure 2 An example of class pair separation



one class and S_3 and S_4 into the other, we could not expect to find a suitable hyperplane.

Another scheme which one would certainly expect to be quite effective consists in the use of a hyperplane (or a pair of parallel hyperplanes) to distinguish between members of each pair of classes; thus $m(m-1)/2$ hyperplanes (or pairs of hyperplanes) are required. Each such hyperplane (or pair of hyperplanes) can of course be found by using the methods described earlier. The decision rule must take into account the fact that if a hyperplane is suitably located to differentiate between two particular classes, then the location with respect to this hyperplane of any vector arising from a pattern from any third class will contain no information.

This method has been called *class pair separation*; the three hyperplanes which would separate S_1 from S_2, S_3 and S_4 are indicated in Figure 2. The disadvantage of this method is the comparatively large number of hyperplanes required; if there were 14 characters in the font, then 91 hyperplanes would be needed, and if there were 26 characters then 325 hyperplanes would be necessary.

We have had some success with a scheme intermediate between these two, namely, the use of one hyperplane to separate the vectors arising from the presentation of patterns from one class from those arising from the presentation of members of all other classes taken together. Thus, to distinguish among the members of m different classes of patterns m hyperplanes are required. To express this more formally, suppose we let S_1, S_2, $\cdots$, S_m be the m classes of patterns to be identified. For each value of i between 1 and m, let W^i and t_i be chosen (using the method described earlier in the paper) so that the hyperplane with equation

$$W^i \cdot X + t_i = 0$$

distinguishes the vectors arising from the presentation of members of S_i from those arising from the presentation of members of all other classes. An appropriate decision rule is: if a pattern produces the vector X in measurement space, then this pattern is identified as belonging to S_i provided

$$W^i \cdot X + t_i > r$$

and for all $j \neq i$

$$W^j \cdot X + t_j < -r$$

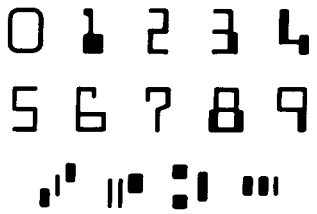
for some suitably chosen value of r; if these conditions are not satisfied for any value of i, then the pattern is rejected.

Another decision rule which may be implemented using these same vectors W^1, W^2, $\cdots$, W^m and constants t_1, t_2, $\cdots$, t_m is the following: if a pattern produces the vector X in measurement space, then it is identified as having come from S_i provided

$$W^i \cdot X + t_i > W^j \cdot X + t_j + \epsilon \tag{10}$$

for all values of j different from i, where ϵ is a positive number chosen in advance; if this condition is not satisfied for any value of i, then the pattern is rejected. It is evident that this rule can be useful only if the vectors W^1, W^2, $\cdots$, W^m bear a suitable relation to one another, as for, example, might be true if they were all unit vectors, so that the linear forms $W^i \cdot X + t_i$ would represent signed distances of the vector X from the corresponding hyperplanes. Our experience indicates that the performance of a processor using this rule is about an order of magnitude better than that of a processor using the rule given in the last paragraph. This is the decision rule on which we have concentrated our attention; we have referred to it as a *ramp method* because of the circuitry used in its implementation.

Figure 3 The American Bankers Association E-13 B font



Geometrically, this ramp method amounts to class-pair separation. In fact, this becomes quite clear if the inequality (10) is rewritten in the form

$$(W^i - W^j) \cdot X + (t_i - t_j) > \epsilon \qquad \text{for all} \quad i \neq j,$$

and we note further that there is no restriction on the values of the quantities

$$(W^k - W^j) \cdot X + (t_k - t_j)$$

when both k and j are different from i. Because of its relative simplicity and familiarity, we chose to base our experimental work on the fourteen patterns of the magnetic ink character recognition font now in use in the banking industry. This font is shown in Figure 3.

the experimental system

The pattern identification system operated as follows. The characters were printed in magnetic ink, as described earlier. Before presentation to the transducer, they were magnetized with an alternating field such that seven complete cycles spanned the width of the widest character. The transducer consisted of a column of thirty reading heads arranged to scan a character along thirty horizontal rows, the tallest characters being nominally covered by a contiguous group of eight of these heads. Ten channels were derived from the thirty outputs by forming the ten linear combinations of the output of every tenth head. This technique solved the vertical registration problem. The initial part of the processor sampled the output of each of the ten channels at seven equally spaced times and quantitized the result into two levels such that a measurement was produced on a 7×10 cylindrically connected matrix which resembled the original printed character if viewed from the proper orientation.

It was apparent that, in effect, ten different measurements were performed on each character scanned and there remained the problem of selecting the measurement or measurements on which to attempt recognition. This problem was resolved by positioning the image of the pattern in the matrix with a set of positioning rules.

The processor may be most conveniently thought of as divided

into two parts. The first part we have just described. The remainder of the processor accepted the selected measurements produced for each character scanned and performed the required identification; this second part we refer to as the *categorizer*. Functionally, this whole system is identical with the pattern identification system used in the IBM 1210 Reader-Sorter which is now in commercial use; however, in the 1210 the categorizer is based on Boolean logic, whereas our system is based on linear decision functions as described above.

We have not actually constructed such a pattern recognition system in its entirety. Instead we have used the transducer and the initial part of the processor of the 1210 to record the 70-bit patterns on magnetic tape; the training and testing of the categorizer was then simulated using the IBM 7090 computer, i.e., the training and testing of our simulated categorizer was done only with measurements produced by the initial part of the 1210. The simulation program was exactly an implementation of the scheme described earlier in the paper: for training we used one pair of parallel hyperplanes to separate each class from all other classes, and for recognition we used the ramp method. Some indication of the results obtained with this program are described below. In order to relate to reality these simulation results, a hardware implementation of a limited version of the categorizer was constructed. This machine accepted 70-bit patterns set manually with switches and identified a pattern as a 0, 1, 2, or 3; this machine was not an adaptive network, but was constructed using the results of the simulation program. Its successful performance demonstrated that the simulation results did in fact have the meaning purported. This machine and one of the problems arising in its construction are described later in the paper.

experimental results

The main source of data for our experimental work was a magnetic tape upon which was recorded the result of presenting slightly over one million characters to the IBM 1210. Mint documents with nominally perfect printed characters were used. Our tape contained about 27,000 distinct binary patterns; to save handling time, it was edited so as to list each pattern only once, but to indicate with each pattern its frequency of occurrence. Thus, in effect, we worked with a typical distribution of patterns produced by mint documents; all recognition results refer to this distribution. For training purposes, we extracted about 5,000 of these patterns and recorded them on a separate tape.

The result of a training routine was a set of fourteen vectors in 70-dimensional space, or *masks* as we have called them. Three of the masks are shown in Figures 4, 5, and 6. The upper parts of these figures are the ideal or nominally perfect patterns as seen by the categorizer, while the lower parts are the masks themselves. One can well think of these in terms of contour maps of surfaces; in this instance we have encircled the higher parts of the ridges and shaded the deeper parts of the valleys. In general, the peaks will be contributed by the character itself, whereas the valleys

Figure 4 Ideal 1 pattern (top) and mask (bottom)

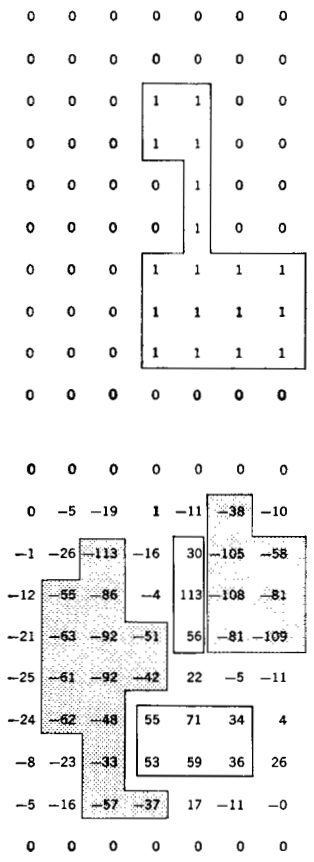


Figure 5 Ideal 3 pattern (top) and mask (bottom)

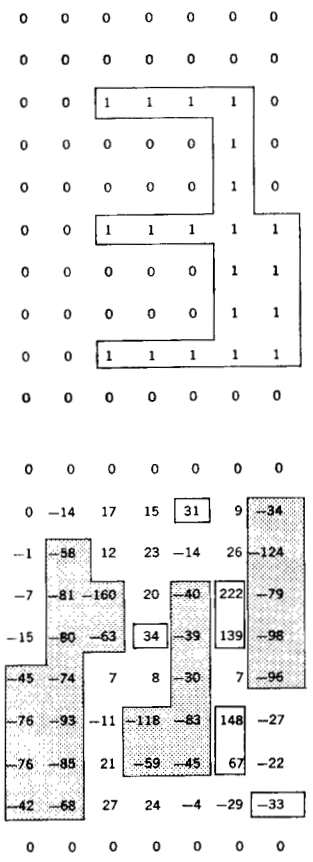
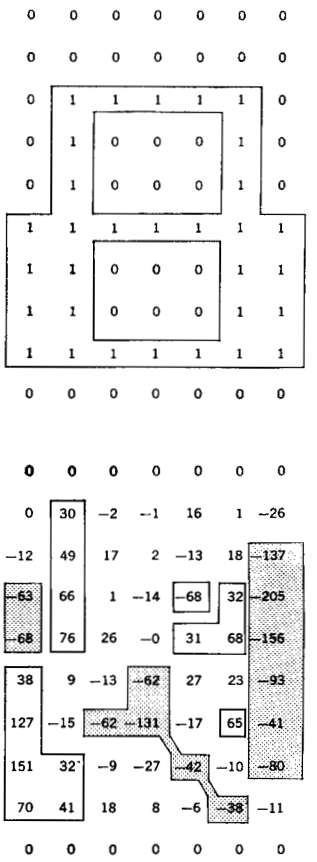


Figure 6 Ideal 8 pattern (top) and mask (bottom)



will be due to other characters which overlap the character in question in a significant way.

We refer to the entries in the masks as *weights*. The variation of these quantities over the entire family of masks is of some interest, for it provides an indication of the range of values required of a variable weight in order that it be useful in an adaptive device. For this particular system and font, the weights ranged, in increments of 1, from 1 up to about 200. But this is not meant to imply that accuracy to within 1/2 of 1% is required or even useful: actually, we have not yet ascertained how the performance of the system deteriorates as the weights are rounded off or otherwise perturbed.

Figure 7 illustrates the performance typical of the simulated system. In this particular instance, d was chosen to be 200 (here d has the same meaning as in the training rules described earlier in the paper). As suggested earlier in the paper, in order to use the ramp method some normalization of the masks (vectors) is required; in this instance we merely divided each weight in a particular mask by the sum of the absolute values of the weights

Figure 7 Substitution rate and reject rate plotted against the discrimination level

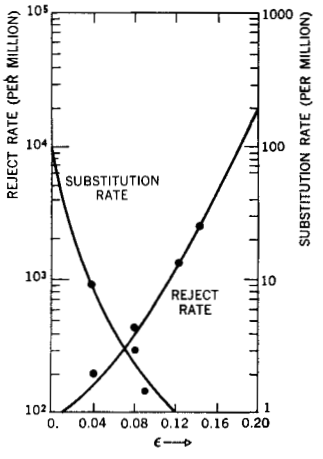
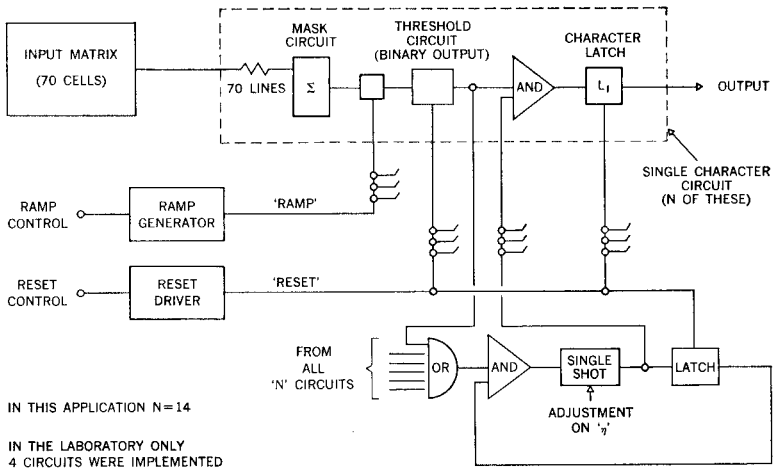


Figure 8 Identification system



which originally appeared in that mask. Thus the output of any one mask ranges over some interval of length 1. As described earlier in the paper, ϵ represents the minimum permissible difference between the maximum signal and the next largest one. Note that as ϵ is decreased, the rejection rate is decreased, but at the expense of permitting substitution errors.

description of the categorizer

The categorizer which was actually constructed in hardware is illustrated in Figure 8. Provision was made for entering 70-bit patterns manually by setting switches. Four circuits representing masks with weights determined by the simulation program described above were constructed, one of these circuits being for each of the characters 0, 1, 2, and 3; for our purposes there seemed to be very little need to build all fourteen. For a given input pattern X, the output of each of these cricuits was proportional to the corresponding quantity $x = W^i \cdot X + t_i$. This output could have been either a current or a voltage; we elected to use voltage. Provision was made to determine which circuit had the largest output, and whether this output exceeded the next largest output by a predetermined amount which we will call η; the various possible outcomes were indicated by means of lights.

A convenient method of comparing the outputs of the several circuits is as follows. As indicated in Figure 8, a "ramp control" is added to the threshold circuits which follow each of the mask circuits, there being but a single ramp generator for the entire system. Initially, the input from the ramp generator is sufficiently great to cause all threshold circuits to be off, no matter how large the output of the mask circuits. Then, at some time during the character cycle, the ramp voltage decreases linearly. When the first threshold circuit comes on, it sets its latch and a single shot fires, the width of the single shot pulse being proportional to η. If any other threshold circuit comes on while the single shot is on, it also sets its latch, but those coming on after the single shot goes

off do not set their latches. At the end of the ramp cycle, if just one latch is on then the pattern is identified at the corresponding character, but if more than one is on then the pattern is rejected.

component deviation problem

One of the problems which we considered in the construction of a physical implementation of the categorizer was to take into account the deviations of commercially available components from their nominal values. We made the appropriate analyses for both the case of a voltage output and a current output; because it is somewhat more transparent, we will describe the current case, although for circuit reasons we actually built mask circuits with voltage outputs.

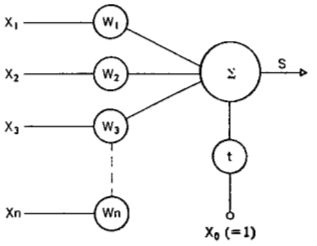
A schematic representation of a mask circuit with current outputs is shown in Figure 9. The output signal is to be proportional to

$$S = w_1x_1 + w_2x_2 + \cdots + w_nx_n + t$$

where $x_1, x_2, \cdots, x_n$ are binary variables (assume the values 0 or 1) and the numbers $w_1, w_2, \cdots, w_n$ are arbitrary subject to the condition that

$$|w_1| + |w_2| + \cdots + |w_n| = 1.$$

Figure 9 Schematic representation of mask circuit

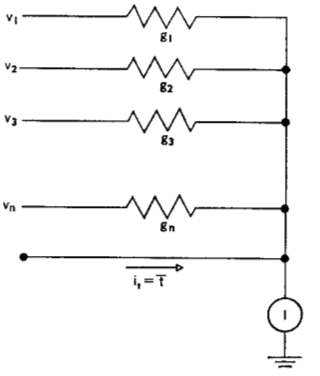


A suitable physical analog is the current summing network of Figure 10. The output current I is given by

$$I = g_1v_1 + g_2v_2 + g_3v_3 + \cdots + g_nv_n + \bar{t}.$$

In this expression, $g_1, g_2, \cdots, g_n$ are conductances chosen to be proportional, respectively, to $|w_1|, |w_2|, \cdots, |w_n|$. If $w_i > 0$, v_i takes on the values 0 and V, respectively, as $x_i = 0$ or $x_i = 1$ and if $w_i < 0$, v_i takes on the values 0 and $-V$, respectively, as $x_i = 0$ or $x_i = 1$. The current I flowing in the network will thus be proportional to the output signal S for any binary pattern. However, the three voltages present, namely V, 0, and $-V$, are an inconvenience to the circuit designer, and therefore it is worthwhile to make the following alteration, which is familiar in the field of Boolean threshold logic: the variables $v_1, v_2, \cdots, v_n$ are replaced by $v_1', v_2', \cdots, v_n'$, where

Figure 10 Physical analog of mask circuit



$$v_i' = v_i \quad \text{if} \quad w_i > 0$$

$$v_i' = v_i + V \quad \text{if} \quad w_i < 0$$

so that

$$I = g_1v_1' + g_2v_2' + \cdots + g_nv_n' + t'$$

where

$$t' = t - V \sum g_i,$$

the sum ranging over those i for which $w_i < 0$. Now if $w_i > 0$, v_i' takes on the value 0 or V according to whether x_i is 0 or 1, but if $w_i < 0$, then v_i' assumes the value V or 0 as x_i takes on the value 0 or 1. Geometrically this amounts to moving the configuration consisting of a cube V units on its edge and a hyperplane passing through it parallel to itself until the cube lies in the

"first 2^n-ant", i.e., until that cube coincides with the cube whose edges are the n vectors $(V, 0, 0, \cdots, 0)$, $(0, V, 0, \cdots, 0)$, $\cdots$, $(0, 0, 0, \cdots, V)$. Note that this same translation scheme could be used more generally to cause the circuit to operate between any two voltages V_1 and V_2. Thus, at any rate, we see that a current summing network can be constructed which is analogous to any given mask and which uses as inputs only the two voltages 0 and V.

Let us suppose, then that a mask is to be implemented using this circuit. The conductances $g_1, g_2, \cdots, g_n$ are to be implemented using resistors which may deviate from their nominal values by as much as a certain fixed percentage, so that the actual conductances used will also deviate from their nominal values by about the same percentage (at least for sufficiently precise resistors). Thus, there is a certain number p such that for each i the value of the ith conductance lies between $(1 - p)g_i$ and $(1 + p)g_i$. Similarly the voltages nominally equal to V and 0 may lie between $V - \delta$ and $V + \delta$ and between $-\delta$ and δ respectively, where δ is a constant. The number l can be determined rather more accurately than $g_1, g_2, \cdots, g_n$ and its deviations from nominal will be ignored. Suppose we let c be the sum of the conductances:

$$c = g_1 + g_2 + \cdots + g_n.$$

Now when a pattern is presented, some of the input lines will receive a nominal voltage of V; the remainder will nominally receive 0 volts. Let the sum of the conductances associated with the first of these sets of lines be c_1, and the sum of the other conductances be c_2. Then the nominal value of the current will be

$$I_{\text{nom}} = c_1 V + l,$$

the largest possible value will be

$$I_{\max} = (1 + p)c_1(V + \delta) + (1 + p)c_2\delta,$$

and similarly the smallest possible value will be

$$I_{\min} = (1 - p)c_1(V - \delta) + (1 - p)c_2(-\delta).$$

We find then that

$$I_{\max} - I_{\text{nom}} = c\delta + pc_1V + cp\delta$$

and since $c_1 \leq c$ we conclude that

$$I_{\max} - I_{\text{nom}} \leq c(\delta + pV + p\delta).$$

A similar calculation may be made for $I_{\min}$, and therefore we conclude that the actual current will differ from the nominal by at most $cV(p + q + pq)$ where we have set $q = \delta/V$. Evidently the maximum current which can flow through any mask circuit is cV. Also,

$$\frac{cV(p + q + pq)}{cV} = p + q + pq \simeq p + q$$

since $pq \ll 1$. Thus, the tolerance of the output of the mask circuit is $100(p + q)\%$. This shows explicitly the relation of the voltage and conductance tolerances to the tolerance of the mask circuits. The deviations of the threshold detector and the ramp generator from nominal are ignored, for they can be controlled quite precisely.

Now suppose that there are to be k such mask circuits and for a binary pattern X, let

$$f_i(X) = W^i \cdot X + t_i$$

where $i = 1, 2, \cdots, k$. If we let $\phi_i(X)$ be the (actual) output of the ith mask circuit, since the nominal output of the ith mask circuit is $cVf_i(X)$, we conclude that

$$|\phi_i(X) - cVf_i(X)| \leq (p + q)cV. \tag{11}$$

Suppose that it has been decided to use a certain discrimination level ϵ, i.e., that we want to use the decision rule: the binary pattern X is identified as having come from the ith character provided

$$f_i(X) > f_j(X) + \epsilon \quad \text{for all} \quad j \neq i. \tag{12}$$

This inequality is equivalent to

$$cVf_i(X) > cVf_j(X) + \epsilon cV \quad \text{for all} \quad j \neq i.$$

In view of (11), in order to insure (12) it is sufficient to require

$$\phi_i(X) > \phi_j(X) + \epsilon cV + 2(p + q)cV,$$

in other words, to chose the parameter as

$$\eta = cV(2p + 2q). \tag{13}$$

Thus if η is so chosen, we can infer that if the ith light turns on, inequalities (12) hold. Of course, in any specific device the deviations of the actual values from the nominal values may well be such that the choice of η given by (13) imposes rather more stringent requirements than those given by (12).

In the particular case of the model we built, we used 1% resistors, so that $p = 0.01$; V was 12 volts and δ was 0.78 volts, so that $q = 0.065$. For example, if one wanted to guarantee (for this sample) no errors at all, then one might choose $\epsilon = 0.12$, and η would be 3.24 c. Or one might choose $\epsilon = 0$ and $\eta = 1.8\ c$, in which case one would be sure (again for this sample) that there would be no more than 100 substitution errors per million characters.

For a number of patterns we measured the outputs of the mask circuits and compared them with the corresponding (properly scaled) quantities in the simulated categorizer; agreement was found to be within 1%. This agreement was well within the limits set by the pessimistic design philosophy upon which the above analysis was based, and there is a strong suggestion that these criteria are too severe. Thus it appears that in the case of our

model ϵ is very nearly proportional to η, i.e., that our model uses the decision rule: the binary pattern X is identified as having come from the ith character provided

$$f_i(X) > f_j(X) + \frac{\eta}{cV} \quad \text{for all} \quad j \neq i.$$

Since the input to this categorizer was manual, it was not economically feasible to test it with a large sample of patterns, and therefore we have no extensive experimental curves corresponding to those shown in Figure 7. However, the response of this categorizer to each of 100 patterns presented to it was identical to the response of the simulated categorizer to the same patterns. Thus we feel able to conclude that the simulated categorizer can be designed to operate substantially as predicted, and in particular that the curves shown in Figure 7 very nearly describe our categorizer. The categorizer simulated, designed, and tested in the present work represents the most straightforward application of linear decision functions to a pattern recognition task, inasmuch as the categorizer inputs were simple measurements representing individual spots of ink in the input pattern. One intention was to determine the capability of such a simple network when realistic devices and component tolerances are taken into account. A second purpose was to test the utility of adaptive learning techniques in handling realistic patterns.

More complex networks capable of improved performance immediately suggest themselves; indeed some of these have already been simulated. Some of the modifications that may be made include: the addition of a layer of Boolean logic operating on the raw measurements (for instance to accomplish feature detection), the use of additional class-pair planes to resolve particular class-pair conflicts remaining in the existing categorizer, and the use of several layers of threshold circuits wherein the early layers are trained according to codes indicating the presence of particular features.

ACKNOWLEDGMENT

The authors would like to acknowledge the assistance of members of the Scientific Computation Laboratory, in particular, R. N. Ascher and F. E. McFarlin. C. E. Kiessling, I. G. Akmenkalns, and L. J. LaBalbo of Advanced Electrical Technology also contributed materially to this project.

Essentially this same paper was presented at the *Symposium on Learning, Adaptation and Control in Information Systems* (Northwestern University, June 17 and 18, 1963) sponsored by the Office of Naval Research. Appreciation is due the ONR for permission to publish.

CITED REFERENCES

1. K. R. Eldredge, F. J. Kamphoefner, and P. H. Wendt, "Teaching Machines to Read," *SRI Journal*, First quarter, 18–23, 1957.

2. K. R. Eldredge, F. J. Kamphoefner, and P. H. Wendt, "Automatic Input for Business Data Processing Systems," *Proceedings of the EJCC*, 69–73, Dec. 10–12, 1956.
3. W. T. Booth, G. M. Miller, and O. A. Schleich, "Design Considerations for Stylized Font Character Readers," 115–128 of *Optical Character Recognition*, edited by Fischer et al., Spartan Books, 1962.
4. C. K. Chow, "An Optimum Character Recognition System using Decision Functions," *IRE Transactions on Electronic Computers*, **EC-6**, no. 4, 247–254, December 1957.
5. W. H. Highleyman, "Linear Decision Functions, with Application to Pattern Recognition," *Proceedings of the IRE*, **50**, no. 6, 1501–1514, June 1962.
6. W. H. Highleyman, *Linear Decision Functions, with Application to Pattern Recognition*. Ph.D. Dissertation, Polytechnic Institute of Brooklyn, N. Y., June 1961. Available from University Microfilms, Ann Arbor, Michigan.
7. H. D. Block, "The Perceptron: A Model for Brain Functioning. I," *Reviews of Modern Physics*, **34**, no. 1, 123–135, January 1962.
8. H. D. Block, B. W. Knight, Jr., and F. Rosenblatt, "Analysis of a Four-Layer Series-Coupled Perceptron. II," *Reviews of Modern Physics*, **34**, no. 1, 135–142, January 1962.
9. A. G. Konheim, "A Geometric Convergence Theorem for the Perceptron," IBM Research Paper RC-621, 16 Feb. 1962.
10. A. Novikoff, "On Convergence Proofs for Perceptrons," presented at the Symposium on Mathematical Theory of Automation, Polytechnic Inst. of Brooklyn, 24 April 1962.