4

Pattern Recognition

Most of the attendees of the Dartmouth Summer Project were interested in mimicking the higher levels of human thought. Their work benefitted from a certain amount of introspection about how humans solve problems. Yet, many of our mental abilities are beyond our power of introspection. We don't know how we recognize speech sounds, read cursive script, distinguish a cup from a plate, or identify faces. We just do these things automatically without thinking about them. Lacking clues from introspection, early researchers interested in automating some of our perceptual abilities based their work instead on intuitive ideas about how to proceed, on networks of simple models of neurons, and on statistical techniques. Later, workers gained additional insights from neurophysiological studies of animal vision.

In this chapter, I'll describe work during the 1950s and 1960s on what is called "pattern recognition." This phrase refers to the process of analyzing an input image, a segment of speech, an electronic signal, or any other sample of data and classifying it into one of several categories. For character recognition, for example, the categories would correspond to the several dozen or so alphanumeric characters.

Most of the pattern-recognition work in this period dealt with two-dimensional material, such as printed pages or photographs. It was already possible to scan images to convert them into arrays of numbers (later called "pixels"), which could then be processed by computer programs such as those of Dinneen and Selfridge. Russell Kirsch and colleagues at the National Bureau of Standards (now the National Institute for Standards and Technology) were also among the early pioneers in image processing. In 1957, Kirsch built and used a drum scanner to scan a photograph of his three-month-old son, Walden. Said to be the first scanned photograph, it measured 176 pixels on a side and is depicted in Fig. 4.1.[1] Using his scanner, he and colleagues experimented with picture-procesing programs running on their SEAC (Standards Eastern Automatic Computer) computer.[2]

4.1 Character Recognition

Early efforts at the perception of visual images concentrated on recognizing alphanumeric characters on documents. This field came to be known as "optical character recognition." A symposium devoted to reporting on progress on this topic was held in Washington, DC, in January 1962.[3] In summary, devices existed at that time for reasonably accurate recognition of fixed-font (typewritten or printed) characters on paper. Perhaps the state of things then was best expressed by one of the participants

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

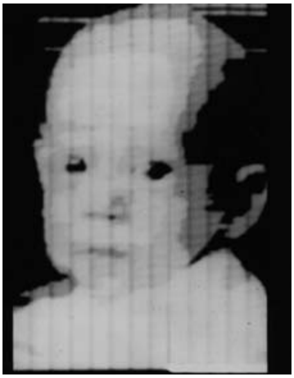


Figure 4.1. An early scanned photograph. (Photograph used with permission of NIST.)

of the symposium, J. Rabinow of Rabinow Engineering, who said “We think, in our company, that we can read anything that is printed, and we can even read some things that are written. The only catch is, ‘how many bucks do you have to spend?’ ”[4]

A notable success during the 1950s was the magnetic ink character recognition (MICR) system developed by researchers at SRI International (then called the Stanford Research Institute) for reading stylized magnetic ink characters at the bottom of checks. (See Fig. 4.2.) MICR was part of SRI’s ERMA (Electronic Recording Method of Accounting) system for automating check processing and checking account management and posting.

According to an SRI Web site, “In April 1956, the Bank of America announced that General Electric Corporation had been selected to manufacture production models. . . . In 1959, General Electric delivered the first 32 ERMA computing systems to the Bank of America. ERMA served as the Bank’s accounting computer and check handling system until 1970.”[5]

Most of the recognition methods at that time depended on matching a character (after it was isolated on the page and converted to an array of 0’s and 1’s) against prototypical versions of the character called “templates” (also stored as arrays in the computer). If a character matched the template for an “A,” say, sufficiently better than it matched any other templates, the input was declared to be an “A.” Recognition accuracy degraded if the input characters were not presented in standard orientation, were not of the same font as the template, or had imperfections.

The 1955 papers by Selfridge and Dinneen (which I have already mentioned on p. 50) proposed some ideas for moving beyond template matching. A 1960 paper by Oliver Selfridge and Ulrich Neisser carried this work further.[6] That paper is

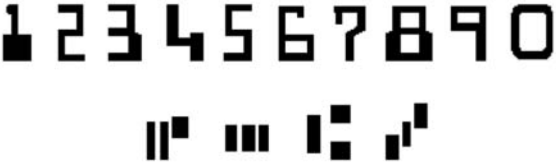


Figure 4.2. The MICR font set.

important because it was a successful, early attempt to use image processing, feature extraction, and learned probability values in hand-printed character recognition. The characters were scanned and represented on a 32 × 32 "retina" or array of 0's and 1's. They were then processed by various refining operations (similar to those I mentioned in connection with the 1955 Dinneen paper) for removing random bits of noise, filling gaps, thickening lines, and enhancing edges. The "cleaned-up" images were then inspected for the occurrence of "features" (similar to the features I mentioned in connection with the 1955 Selfridge paper.) In all, 28 features were used – features such as the maximum number of times a horizontal line intersected the image, the relative lengths of different edges, and whether or not the image had a "concavity facing south."

Recalling Selfridge's Pandemonium system, we can think of the feature-detection process as being performed by "demons." At one level higher in the hierarchy than the feature demons were the "recognition demons" – one for each letter. (The version of this system tested by Worthie Doyle of Lincoln Laboratory was designed to recognize ten different hand-printed characters, namely, A, E, I, L, M, N, O, R, S, and T.) Each recognition demon received inputs from each of the feature-detecting demons. But first, the inputs to each recognition demon were multiplied by a weight that took into account the importance of the contribution of the corresponding feature to the decision. For example, if feature 17 were more important than feature 22 in deciding that the input character was an "A," then the input to the "A" recognizer from feature 17 would be weighted more heavily than would be the input from feature 22. After each recognition demon added up the total of its weighted inputs, a final "decision demon" decided in favor of that character having the largest sum.

The values of the weights were determined by a learning process during which 330 "training" images were analyzed. Counts were tabulated for how many times each feature was detected for each different letter in the training set. These statistical data were used to make estimates of the probabilities that a given feature would be detected for each of the letters. These probability estimates were then used to weight the features summed by the recognizing demons.

After training, the system was tested on samples of hand-printed characters that it had not yet seen. According to Selfridge and Neisser, "This program makes only about 10 percent fewer correct identifications than human readers make – a respectable performance, to be sure."

4.2 Neural Networks

4.2.1 *Perceptrons*

In 1957, Frank Rosenblatt (1928–1969; Fig. 4.3), a psychologist at the Cornell Aeronautical Laboratory in Buffalo, New York, began work on neural networks under a project called PARA (Perceiving and Recognizing Automaton). He was motivated by the earlier work of McCulloch and Pitts and of Hebb and was interested in these networks, which he called *perceptrons*, as potential models of human learning, cognition, and memory.[7]

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

Figure 4.3. Frank Rosenblatt (left) working (with Charles Wrightman) on a prototype A-unit. (Courtesy of the Division of Rare and Manuscript Collections, Cornell University Library.)

Continuing during the early 1960s as a professor at Cornell University in Ithaca, New York, he experimented with a number of different kinds of perceptrons. His work, more than that of Clark and Farley and of the other neural network pioneers, was responsible for initiating one of the principal alternatives to symbol-processing methods in AI, namely, neural networks.

Rosenblatt's perceptrons consisted of McCulloch–Pitts-style neural elements, like the one shown in Fig. 4.4. Each element had inputs (coming in from the left in the figure), "weights" (shown by bulges on the input lines), and one output (going out to the right). The inputs had values of either 1 or 0, and each input was multiplied by its associated weight value. The neural element computed the sum of these weighted values. So, for example, if all of the inputs to the neural element in Fig. 4.4 were equal to 1, the sum would be 13. If the sum were greater than (or just equal to) a "threshold value," say 7, associated with the element, then the output of the neural element would be 1, which it would be in this example. Otherwise the output would be 0.

A perceptron consists of a network of these neural elements, in which the outputs of one element are inputs to others. (There is an analogy here with Selfridge's

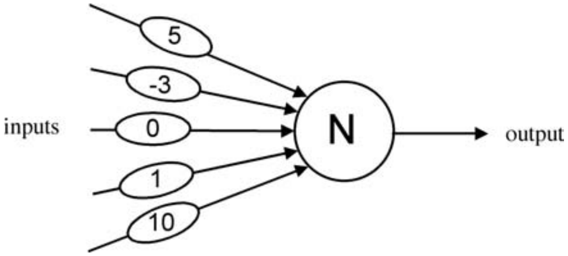


Figure 4.4. Rosenblatt's neural element with weights.

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

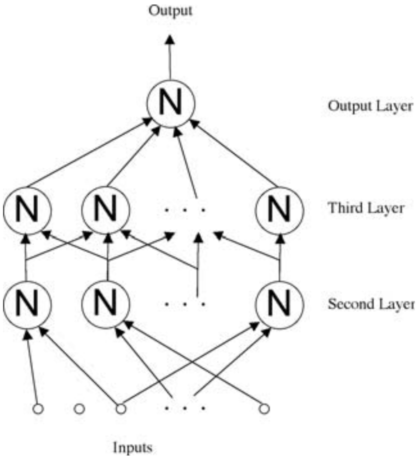


Figure 4.5. A perceptron.

Pandemonium in which mid-level demons receive “shouts” from lower level demons. The weights on a neural element’s input lines can be thought of as analogous to the strength-enhancing or strength-diminishing “volume controls” in Pandemonium.) A sample perceptron is illustrated in Fig. 4.5. [Rosenblatt drew his perceptron diagrams in a horizontal format (the electrical engineering style), with inputs to the left and output to the right. Here I use the vertical style generally preferred by computer scientists for hierarchies, with the lowest level at the bottom and the highest at the top. To simplify the diagram, weight bulges are not shown.] Although the perceptron illustrated, with only one output unit, is capable of only two different outputs (1 or 0), multiple outputs (sets of 1’s and 0’s) could be achieved by arranging for several output units.

The input layer, shown at the bottom of Fig. 4.5, was typically a rectangular array of 1’s and 0’s corresponding to cells called “pixels” of a black-and-white image. One of the applications Rosenblatt was interested in was, like Selfridge, character recognition.

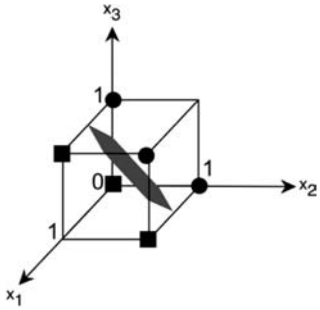
I’ll use some simple algebra and geometry to show how the neural elements in perceptron networks can be “trained” to produce desired outputs. Let’s consider, for example, a single neural element whose inputs are the values x_1, x_2, and x_3 and whose associated weight values are w_1, w_2, and w_3. When the sum computed by this element is exactly equal to its threshold value, say t, we have the equation

$$w_1x_1 + w_2x_2 + w_3x_3 = t.$$

In algebra, such an equation is called a “linear equation.” It defines a linear boundary, that is, a plane, in a three-dimensional space. The plane separates those input values that would cause the neural element to have an output of 1 from those that would cause it to have an output of 0. I show a typical planar boundary in Fig. 4.6.

An input to the neural element can be depicted as a point (that is, a vector) in this three-dimensional space. Its coordinates are the values of x_1, x_2, and x_3, each of which can be either 1 or 0. The figure shows six such points, three of them (the small circles, say) causing the element to have an output of 1 and three (the small squares,

Figure 4.6. A separating plane in a three-dimensional space.



say) causing it to have an output of 0. Changing the value of the threshold causes the plane to move sideways in a direction parallel to itself. Changing the values of the weights causes the plane to rotate. Thus, by changing the weight values, points that used to be on one side of the plane might end up on the other side. "Training" takes place by performing such changes. I'll have more to say about training procedures presently.

In dimensions higher than three (which is usually the case), a linear boundary is called a "hyperplane." Although it is not possible to visualize what is going on in spaces of high dimensions, mathematicians still speak of input points in these spaces and rotations and movements of hyperplanes in response to changes in the values of weights and thresholds.

Rosenblatt defined several types of perceptrons. He called the one shown in the diagram a "series-coupled, four-layer perceptron." (Rosenblatt counted the inputs as the first layer.) It was termed "series-coupled" because the output of each neural element fed forward to neural elements in a subsequent layer. In more recent terminology, the phrase "feed-forward" is used instead of "series-coupled." In contrast, a "cross-coupled" perceptron could have the outputs of neural elements in one layer be inputs to neural elements in the same layer. A "back-coupled" perceptron could have the outputs of neural elements in one layer be inputs to neural elements in lower numbered layers.

Rosenblatt thought of his perceptrons as being models of the wiring of parts of the brain. For this reason, he called the neural elements in all layers but the output layer "association units" ("A-units") because he intended them to model associations performed by networks of neurons in the brain.

Of particular interest in Rosenblatt's research was what he called an "alpha-perceptron." It consisted of a three-layer, feed-forward network with an input layer, an association layer, and one or more output units. In most of his experiments, the inputs had values of 0 or 1, corresponding to black or white pixels in a visual image presented on what he called a "retina." Each A-unit received inputs (which were not multiplied by weight values) from some randomly selected subset of the pixels and sent its output, through sets of adjustable weights, to the final output units, whose binary values could be interpreted as a code for the category of the input image.

Various "training procedures" were tried for adjusting the weights of the output units of an alpha-perceptron. In the most successful of these (for pattern-recognition

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

purposes), the weights leading in to the output units were adjusted only when those units made an error in classifying an input. The adjustments were such as to force the output to make the correct classification for that particular input. This technique, which soon became a standard, was called the "error-correction procedure." Rosenblatt used it successfully in a number of experiments for training perceptrons to classify visual inputs, such as alphanumeric characters, or acoustic inputs, such as speech sounds. Professor H. David Block, a Cornell mathematician working with Rosenblatt, was able to prove that the error-correction procedure was guaranteed to find a hyperplane that perfectly separated a set of training inputs when such a hyperplane existed.[8] (Other mathematicians, such as Albert B. Novikoff at SRI, later developed more elegant proofs.[9] I give a version of this proof in my book *Learning Machines.*[10])

Although some feasibility and design work was done using computer simulations, Rosenblatt preferred building hardware versions of his perceptrons. (Simulations were slow on early computers, thus explaining the interest in building special-purpose perceptron hardware.) The MARK I was an alpha-perceptron built at the Cornell Aeronautical Laboratory under the sponsorship of the Information Systems Branch of the Office of Naval Research and the Rome Air Development Center. It was first publicly demonstrated on 23 June 1960. The MARK I used volume controls (called "potentiometers" by electrical engineers) for weights. These had small motors attached to them for making adjustments to increase or decrease the weight values.

In 1959, Frank Rosenblatt moved his perceptron work from the Cornell Aeronautical Laboratory in Buffalo, New York, to Cornell University, where he became a professor of psychology. Together with Block and several students, Rosenblatt continued experimental and theoretical work on perceptrons. His book *Principles of Neurodynamics* provides a detailed treatment of his theoretical ideas and experimental results.[11] Rosenblatt's last system, called Tobermory, was built as a speech-recognition device.[12] [Tobermory was the name of a cat that learned to speak in *The Chronicles of Clovis*, a group of short stories by Saki (H. H. Munro).] Several Ph.D. students, including George Nagy, Carl Kessler, R. D. Joseph, and others, completed perceptron projects under Rosenblatt at Cornell.

In his last years at Cornell, Rosenblatt moved on to study chemical memory transfer in flatworms and other animals – a topic quite removed from his perceptron work. Tragically, Rosenblatt perished in a sailing accident in Chesapeake Bay in 1969.

Around the same time as Rosenblatt's alpha-perceptron, Woodrow W. (Woody) Bledsoe (1921–1995) and Iben Browning (1918–1991), two mathematicians at Sandia Laboratories in Albuquerque, New Mexico, were also pursuing research on character recognition that used random samplings of input images. They experimented with a system that projected images of alphanumeric characters on a 10 × 15 mosaic of photocells and sampled the states of 75 randomly chosen pairs of photocells. Pointing out that the idea could be extended to sampling larger groups of pixels, say *N* of them, they called their method the "*N*-tuple" method. They used the results of this sampling to make a decision about the category of an input letter.[13]

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

4.2.2 *ADALINES and MADALINES*

Independently of Rosenblatt, a group headed by Stanford Electrical Engineering Professor Bernard Widrow (1929–) was also working on neural-network systems during the late 1950s and early 1960s. Widrow had recently joined Stanford after completing a Ph.D. in control theory at MIT. He wanted to use neural-net systems for what he called "adaptive control." One of the devices Widrow built was called an "ADALINE" (for adaptive linear network). It was a single neural element whose adjustable weights were implemented by switchable (thus adjustable) circuits of resistors. Widrow and one of his students, Marcian E. "Ted" Hoff Jr. (who later invented the first microprocessor at Intel), developed an adjustable weight they called a "memistor." It consisted of a graphite rod on which a layer of copper could be plated and unplated – thus varying its electrical resistance. Widrow and Hoff developed a training procedure for their ADALINE neural element that came to be called the Widrow–Hoff least-mean-squares adaptive algorithm. Most of Widrow's experimental work was done using simulations on an IBM1620 computer. Their most complex network design was called a "MADALINE" (for many ADALINEs). A training procedure was developed for it by Stanford Ph.D. student William Ridgway.[14]

4.2.3 *The MINOS Systems at SRI*

Rosenblatt's success with perceptrons on pattern-recognition problems led to a flurry of research efforts by others to duplicate and extend his results. During the 1960s, perhaps the most significant pattern-recognition work using neural networks was done at the Stanford Research Institute in Menlo Park, California. There, Charles A. Rosen (1917–2002) headed a laboratory that was attempting to etch microscopic vacuum tubes onto a solid-state substrate. Rosen speculated that circuits containing these tubes might ultimately be "wired-up" to perform useful tasks using some of the training procedures being explored by Frank Rosenblatt. SRI employed Rosenblatt as a consultant to help in the design of an exploratory neural network.

When I interviewed for a position at SRI in 1960, a team in Rosen's lab, under the leadership of Alfred E. (Ted) Brain (1923–2004), had just about completed the construction of a small neural network called MINOS (Fig. 4.7). (In Greek mythology, Minos was a king of Crete and the son of Zeus and Europa. After his death, Minos was one of the three judges in the underworld.) Brain felt that computer simulations of neural networks were too slow for practical applications, thus leading to his decision to build rather than to program. (The IBM 1620 computer being used at the same time by Widrow's group at Stanford for simulating neural networks had a basic machine cycle of 21 microseconds and a maximum of 60,000 "digits" of random-access memory.) For adjustable weights, MINOS used magnetic devices designed by Brain. Rosenblatt stayed in close contact with SRI because he was interested in using these magnetic devices as replacements for his motor-driven potentiometers.

Rosen's enthusiasm and optimism about the potential for neural networks helped convince me to join SRI. Upon my arrival in July 1961, I was given a draft of

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

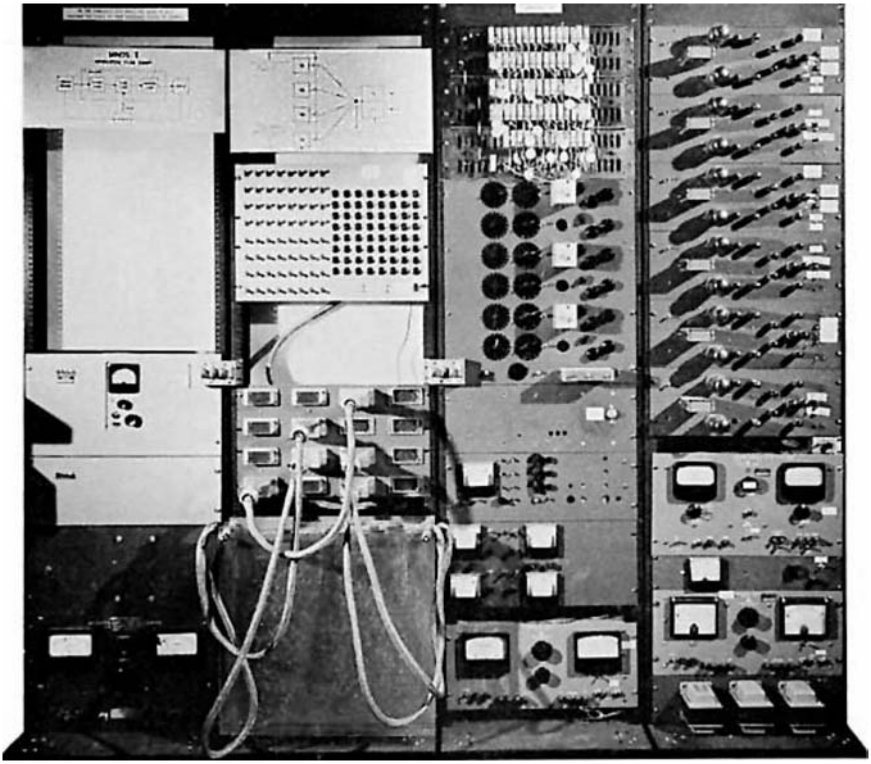


Figure 4.7. MINOS. Note the input switches and corresponding indicator lights in the second-from-the-left rack of equipment. The magnetic weights are at the top of the third rack. (Photograph used with permission of SRI International.)

Rosenblatt's book to read. Brain's team was just beginning work on the construction of a large neural network, called MINOS II, a follow-on system to the smaller MINOS. (See Fig. 4.8.)

Work on the MINOS systems was supported primarily by the U.S. Army Signal Corps during the period 1958 to 1967. The objective of the MINOS work was "to conduct a research study and experimental investigation of techniques and equipment characteristics suitable for practical application to graphical data processing for military requirements." The main focus of the project was the automatic recognition of symbols on military maps. Other applications – such as the recognition of military vehicles, such as tanks, on aerial photographs and the recognition of hand-printed characters – were also attempted.[15]

In the first stage of processing by MINOS II, the input image was replicated 100 times by a 10 × 10 array of plastic lenses. Each of these identical images was then sent through its own optical feature-detecting mask, and the light through the mask was detected by a photocell and compared with a threshold. The result was a set of 100 binary (off–on) values. These values were the inputs to a set of 63 neural elements ("A-units" in Rosenblatt's terminology), each with 100 variable magnetic weights.

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

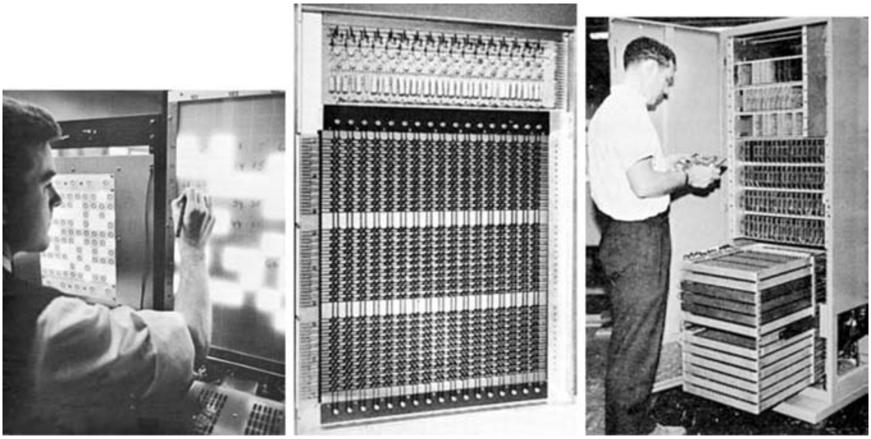


Figure 4.8. MINOS II: operator's display board (left), an individual weight frame (middle), and weight frames with logic circuitry (right). (Photographs used with permission of SRI International.)

The 63 binary outputs from these neural elements were then translated into one of 64 decisions about the category of the original input image. (We constructed 64 equally distant "points" in the sixty-three-dimensional space and trained the neural network so that each input image produced a point closer to its own prototype point than to any other. Each of these prototype points was one of the 64 "maximal-length shift-register sequences" of 63 dimensions.)[16]

During the 1960s, the SRI neural network group, by then called the Learning Machines Group, explored many different network organizations and training procedures. As computers became both more available and more powerful, we increasingly used simulations (at various computer centers) on the Burroughs 220 and 5000 and on the IBM 709 and 7090. In the mid-1960s, we obtained our own dedicated computer, an SDS 910. (The SDS 910, developed at Scientific Data Systems, was the first computer to use silicon transistors.) We used that computer in conjunction with the latest version of our neural network hardware (now using an array of 1,024 preprocessing lenses), a combination we called MINOS III.

One of the most successful results with the MINOS III system was the automatic recognition of hand-printed characters on FORTRAN coding sheets. (In the 1960s, computer programs were typically written by hand and then converted to punched cards by key-punch operators.) This work was led by John Munson (1939–1972; Fig. 4.9), Peter Hart (1941– ; Fig. 4.9), and Richard Duda (1936– ; Fig. 4.9). The neural net part of MINOS III was used to produce a ranking of the possible classifications for each character with a confidence measure for each. For example, the first character encountered in a string of characters might be recognized by the neural net as a "D" with a confidence of 90 and as an "O" with a confidence of 10. But accepting the most confident decision for each character might not result in a string that is a legal statement in the FORTRAN language – indicating that one or more of the decisions was erroneous (where it is assumed that whoever wrote statements on the coding

Figure 4.9. John Munson (left), Peter Hart (middle), and Richard Duda (right). (Photographs courtesy of Faith Munson, of Peter Hart, and of Richard Duda.)

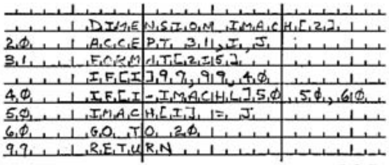
sheet wrote legal statements). Accepting the second or third most confident choices for some of the characters might be required to produce a legal string.

The overall confidence of a complete string of characters was calculated by adding the confidences of the individual characters in the string. Then, what was needed was a way of ranking these overall confidence numbers for each of the possible strings resulting from all of the different choices for each character. Among this ranking of all possible strings, the system then selected the most confident *legal* string.

As Richard Duda wrote, however, "The problem of finding the 1st, 2nd, 3rd, ... most confident string of characters is by no means a trivial problem." The key to computing the ranking in an efficient manner was to use a method called *dynamic programming*.[17] (In a later chapter, we'll see dynamic programming used again in speech recognition systems.)

An illustration of a sample of the original source and the final output is shown in Fig. 4.10.

After the neural net part of the system was trained, the overall system (which decided on the most confident legal string) was able to achieve a recognition accuracy of just over 98% on a large sample of material that was not part of what the system



```
           DIMENSION IMACH[2]
2Ø         ACCEPT 31,I,J
31         FORMAT[2I5]
           IF[I]99,99,4Ø
4Ø         IF[I-IMACHL]5Ø,5Ø,6Ø
5Ø         IMACH[I] = J
6Ø         GO TO 2Ø
99         RETURN
```

```
        DIMENSION IMACM[2]
20      ACCEPT 31,I,J
31      FORMAT[215]
        IF[I]79,99,40
40      IF[I-IMACHL]50,50,60
50      IMACH[I]=J
60      GO TO 20
99      RETURN
```

Figure 4.10. Recognition of FORTRAN characters. Input is above and output (with only two errors) is below. (Illustration used with permission of SRI International.)

was trained on. Recognizing handwritten characters with this level of accuracy was a significant achievement in the 1960s.[18]

Expanding its interests beyond neural networks, the Learning Machines Group ultimately became the SRI Artificial Intelligence Center, which continues today as a leading AI research enterprise.

4.3 Statistical Methods

During the 1950s and 1960s there were several applications of statistical methods to pattern-recognition problems. Many of these methods bore a close resemblance to some of the neural network techniques. Recall that earlier I explained how to decide which of two tones was present in a noisy radio signal. A similar technique could be used for pattern recognition. For classifying images (or other perceptual inputs), it was usual to represent the input by a list of distinguishing "features," such as those used by Selfridge and his colleagues. In alphanumeric character recognition for example, one first extracted features from the image of the character to be classified. Usually the features had numerical values, such as the number of times lines of different angles intersected the character or the length of the perimeter of the smallest circle that completely enclosed the character. Selecting appropriate features was often more art than science, but it was critical to good performance.

We'll need a bit of elementary mathematical notation to help describe these statistically oriented pattern-recognition methods. Suppose the list of features extracted from a character is $\{f_1, f_2, \ldots, f_i, \ldots, f_N\}$. I'll abbreviate this list by the boldface symbol **X**. Suppose there are k categories, $C_1, C_2, \ldots, C_i, \ldots, C_k$ to which the character described by **X** might belong. Using Bayes's rule in a manner similar to that described earlier, the decision rule is the following:

> Decide in favor of that category for which $p(\mathbf{X} \mid C_i)p(C_i)$ is largest, where $p(C_i)$ is the *a priori* probability of category C_i and $p(\mathbf{X} \mid C_i)$ is the likelihood of **X** given C_i. These likelihoods can be inferred by collecting statistical data from a large sample of characters.

As I mentioned earlier, researchers in pattern recognition often describe the decision process in terms of geometry. They imagine that the values of the features obtained from an image sample can be represented as a point in a multidimensional space. If we have several samples for each of, say, two known categories of data, we can represent these samples as scatterings of points in the space. In character recognition, scattering can occur not only because the image of the character might be noisy but also because characters in the same category might be drawn slightly differently. I show a two-dimensional example, with features f_1 and f_2, in Fig. 4.11. From the scattering of points in each category we can compute an estimate of the probabilities needed for computing likelihoods. Then, we can use the likelihoods and the prior probabilities to make decisions.

I show in this figure the boundary, computed from the likelihoods and the prior probabilities, that divides the space into two regions. In one region, we decide in favor of category 1; in the other, we decide in favor of category 2. I also show a new feature point, **X**, to be classified. In this case, the position of **X** relative to the boundary dictates that we classify **X** as a member of category 1.

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

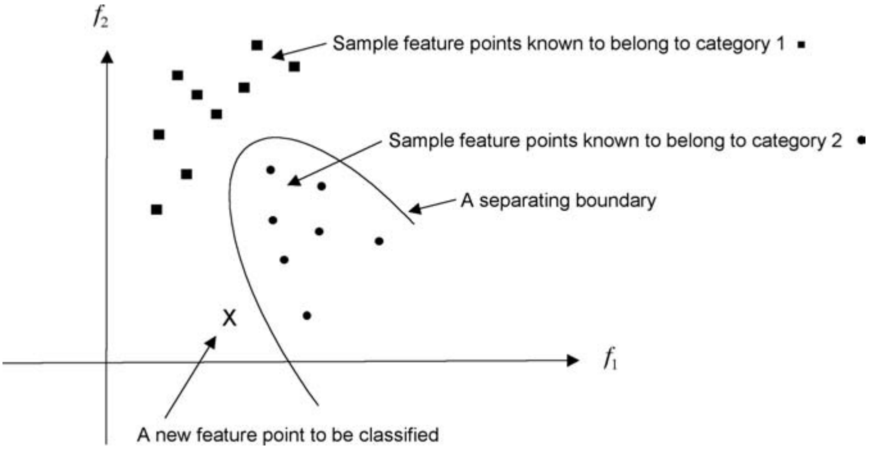

f_2
Sample feature points known to belong to category 1 ■
Sample feature points known to belong to category 2 ●
A separating boundary
X
f_1
A new feature point to be classified

Figure 4.11. A two-dimensional space of feature points and a separating boundary.

There are other methods also for classifying feature points. An interesting example is the "nearest-neighbor" method. In that scheme, invented by E. Fix and J. L. Hodges in 1951,[19] a new feature point is assigned to the same category as that sample feature point to which it is closest. In Fig. 4.11, the new point **X** would be classified as belonging to category 2 using the nearest-neighbor method.

An important elaboration of the nearest-neighbor method assigns a new point to the same category as the majority of the k closest points. Such a decision rule seems plausible (in the case in which there are many, many sample points of each category) because there being more sample points of category C_i closer to an unknown point, **X**, than sample points of category C_j is evidence that $p(\mathbf{X} \mid C_i)p(C_i)$ is greater than $p(\mathbf{X} \mid C_j)p(C_j)$. Expanding on that general observation, Thomas Cover and Peter Hart rigorously analyzed the performance of nearest-neighbor methods.[20]

Any technique for pattern recognition, even those using neural networks or nearest neighbors, can be thought of as constructing separating boundaries in a multidimensional space of features. Another method for constructing boundaries using "potential functions" was suggested by the Russian scientists M. A. Aizerman, E. M. Braverman, and L. I. Rozonoer in the 1960s.[21]

Some important early books on the use of statistical methods in pattern recognition are ones by George Sebestyen,[22] myself,[23] and Richard Duda and Peter Hart.[24] My book also describes some of the relationships between statistical methods and those based on neural networks. The technology of pattern recognition as of the late 1960s is nicely reviewed by George Nagy (who had earlier been one of Frank Rosenblatt's graduate students).[25]

4.4 Applications of Pattern Recognition to Aerial Reconnaissance

The neural network and statistical methods for pattern recognition attracted much attention in many aerospace and avionics companies during the late 1950s and early 1960s. These companies had ample research and development budgets stemming

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

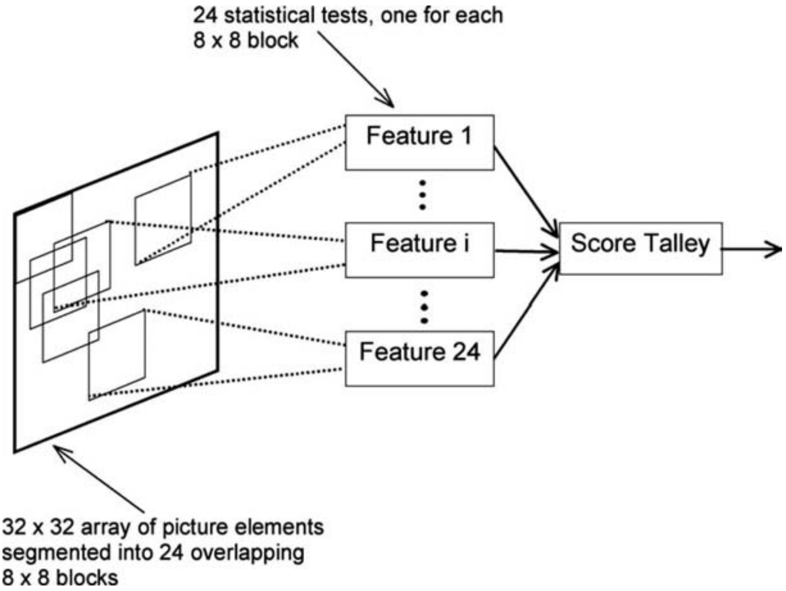

24 statistical tests, one for each
8 x 8 block
Feature 1
Feature i
Feature 24
Score Talley
32 x 32 array of picture elements
segmented into 24 overlapping
8 x 8 blocks

Figure 4.12. A Philco tank-recognition system. (Adapted from Laveen N. Kanal and Neal C. Randall, "Target Detection in Aerial Photography," paper 8.3, *Proceedings of the 1964 Western Electronics Show and Convention (WESCON)*, Los Angeles, CA, Institute of Radio Engineers (now IEEE), August 25–28, 1964.)

from their contracts with the U.S. Department of Defense. Many of them were particularly interested in the problem of aerial reconnaissance, that is, locating and identifying "targets" in aerial photographs. Among the companies having substantial research programs devoted to this and related problems were the Aeronutronic Division of the Ford Motor Co.,[26] Douglas Aircraft Company (as it was known at that time), General Dynamics, Lockheed Missiles and Space Division, and the Philco Corporation. (Philco was later acquired by Ford in late 1961.)

I'll mention some of the work at Philco as representative. There, Laveen N. Kanal (1931–), Neil C. Randall (1930–), and Thomas Harley (1929–) worked on both the theory and applications of statistical pattern-recognition methods. The systems they developed were for screening aerial photographs for interesting military targets such as tanks. A schematic illustration of one of their systems is shown in Fig. 4.12.[27]

Philco's apparatus scanned material from 9-inch film negatives gathered by a U2 reconnaissance airplane during U.S. Army tank maneuvers at Fort Drum, New York. A small section of the scanned photograph, possibly containing an M-48 tank (in standard position and size), was first processed to enhance edges, and the result was presented to the target detection system as an array of 1's and 0's. The first of their systems used a 22 × 12 array; later ones used a 32 × 32 array as shown in Fig. 4.12. The array was then segmented into 24 overlapping 8 × 8 "feature blocks." The data in each feature block were then subjected to a statistical test to decide whether or not the small area of the picture represented by this block contained part of a tank.

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

The statistical tests were based on a "training sample" of 50 images containing tanks and 50 samples of terrain not containing tanks. For each 8 × 8 feature block, statistical parameters were compiled from these samples to determine a (linear) boundary in the sixty-four-dimensional space that best discriminated the tank samples from the nontank samples.

Using these boundaries, the system was then tested on a different set of 50 images containing tanks and 50 images not containing tanks. For each test image, the number of feature blocks deciding "tank present" was tallied to produce a final numerical "score" (such as 21 out of the 24 blocks decided a tank was present). This score could then be used to decide whether or not the image contained a tank.

The authors stated that "the experimental performance of the statistical classification procedure exceeded all expectations." Almost half of the test samples had perfect scores (that is, all 24 feature blocks correctly discriminated between tank and nontank). Furthermore, all of the test samples containing tanks had a score greater than or equal to 11, and all of the test samples not containing tanks had a score less than or equal to 7.

An early tank-detecting system at Philco was built with analog circuitry – not programmed on a computer. As Thomas Harley, the project leader for this system, later elaborated,[28]

> It is important to remember the technological context of the era in which this work was done. The system we implemented had no built-in computational capabilities. The weights in the linear discriminant function were resistors that controlled the current coming from the (binary) voltage source in the shift register elements. Those currents were added together, and each feature was recognized or not depending whether on the sum of those currents exceeded a threshold value. Those binary feature decisions were then summed, again in an analog electrical circuit, not in a computer, and again a decision [tank or no tank] was made depending on whether the sum exceeded a threshold value.

In another system, the statistical classification was implemented by a program, called MULTINORM, running on the Philco 2000 computer.[29] In other experiments, Philco used additional statistical tests to weight some of the feature blocks more heavily than others in computing the final score. Kanal told me that these experiments with weighting the outputs of the feature blocks "anticipated the support vector machine (SVM) classification idea . . . [by] using the first layer to identify the training samples close to the boundary between tanks and non-tanks."[30] (I'll describe the important SVM method in a later chapter.)

Of course, these systems had a rather easy task. All of the tanks were in standard position and were already isolated in the photograph. (The authors mention, however, how the system could be adapted to deal with tanks occurring in any position or orientation in the image.) The photograph in Fig. 4.13 shows a typical tank image. (The nontank images are similar, except without the tank.)

I find the system interesting not only because of its performance but also because it is a layered system (similar to Pandemonium and to the alpha-perceptron) and because it is an example in which the original image is divided into overlapping subimages, each of which is independently processed. As I'll mention later,

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

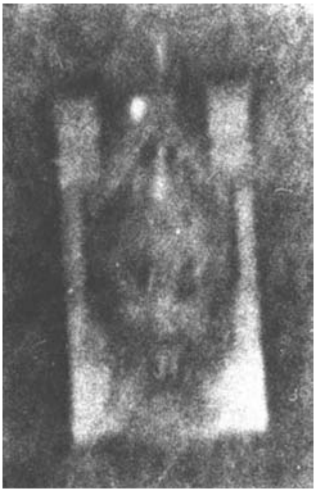


Figure 4.13. A typical tank image. (Photograph courtesy of Thomas Harley.)

overlapping subimages play a prominent role in some computational models of the neocortex.

Unfortunately, the Philco reports giving details of this work aren't readily available.[31] Furthermore, Philco and some of the other groups engaged in this work have disappeared. Here is what Tom Harley wrote me about the Philco reports and about Philco itself:[32]

> Most of the pattern recognition work done at Philco in the 1960s was sponsored by the DoD [Department of Defense], and the reports were not available for public distribution. Since then, the company itself has really vanished into thin air. Philco was bought by Ford Motor Company in 1961, and by 1966, they had eliminated the Philco research labs where Laveen [Kanal] and I were working. Ford tried to move our small pattern recognition group to Newport Beach, CA [the location of Ford's Aeronutronic Division, whose pattern recognition group folded later also], and when we all decided not to go, they transferred us to their Communications Division, and told us to close out our pattern recognition projects. Laveen eventually went off to the University of Maryland, and in 1975, I transferred to the Ford Aerospace Western Development Labs (WDL) in Palo Alto, where I worked on large systems for the intelligence community. In later years, what had been Philco was sold to Loral, and most of that was later sold to Lockheed Martin. I retired from Lockheed in 2001.

Approaches to AI problems involving neural networks and statistical techniques came to be called "nonsymbolic" to contrast them with the "symbol-processing" work being pursued by those interested in proving theorems, playing games, and problem solving. These nonsymbolic approaches found application mainly in pattern recognition, speech processing, and computer vision. Workshops and conferences devoted especially to those topics began to be held in the 1960s. A subgroup of the IEEE Computer Society (the Pattern Recognition Subcommittee of the Data Acquisition and Transformation Committee) organized the first "Pattern Recognition Workshop," which was held in Puerto Rico in October 1966.[33] A second one (which I attended) was held in Delft, The Netherlands, in August 1968. In 1966, this subgroup became the IEEE Computer Society Pattern Analysis and Machine

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

Intelligence (PAMI) Technical Committee, which continued to organize conferences and workshops.[34]

Meanwhile, during the late 1950s and early 1960s, the symbol-processing people did their work mainly at MIT, at Carnegie Mellon University, at IBM, and at Stanford University. I'll turn next to describing some of what they did.

Notes

1. See http://www.nist.gov/public_affairs/techbeat/tb2007_0524.htm. [62]
2. Russell A. Kirsch *et al.*, "Experiments in Processing Pictorial Information with a Digital Computer," *Proceedings of the Eastern Joint Computer Conference*, pp. 221–229, Institute of Radio Engineering and Association for Computing Machinery, December 1957. [62]
3. The proceedings of the conference were published in George L. Fischer Jr. *et al.*, *Optical Character Recognition*, Washington, DC: Spartan Books, 1962. [62]
4. From J. Rabinow, "Developments in Character Recognition Machines at Rabinow Engineering Company," in George L. Fischer Jr. *et al.*, *op. cit.*, p. 27. [63]
5. From http://www.sri.com/about/timeline/erma-micr.html. [63]
6. Oliver G. Selfridge and Ulrich Neisser, "Pattern Recognition by Machine," *Scientific American*, Vol. 203, pp. 60–68, 1960. (Reprinted in Edward A. Feigenbaum and Julian Feldman (eds.), *Computers and Thought*, pp. 237ff, New York: McGraw Hill, 1963.) [63]
7. An early reference is Frank Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain," *Psychological Review*, Vol. 65, pp. 386ff, 1958. [64]
8. H. David Block, "The Perceptron: A Model for Brain Functioning," *Reviews of Modern Physics*, Vol. 34, No. 1, pp. 123–135, January 1962. [68]
9. Albert B. J. Novikoff, "On Convergence Proofs for Perceptrons," in *Proceedings of the Symposium on Mathematical Theory of Automata*, pp. 615–622, Brooklyn, NY: Polytechnic Press of Polytechnic Inst. of Brooklyn, 1963. [68]
10. Nils J. Nilsson, *Learning Machines: Foundations of Trainable Pattern-Classifying Systems*, New York: McGraw-Hill Book Co., 1965; republished as *The Mathematical Foundations of Learning Machines*, San Francisco: Morgan Kaufmann Publishers, 1990. [68]
11. Frank Rosenblatt, *Principles of Neurodynamics*, Washington, DC: Spartan Books, 1962. [68]
12. Frank Rosenblatt, "A Description of the Tobermory Perceptron," *Collected Technical Papers*, Vol. 2, Cognitive Systems Research Program, Cornell University, 1963. [68]
13. Woodrow W. Bledsoe and Iben Browning, "Pattern Recognition and Reading by Machine," *Proceedings of the Eastern Joint Computer Conference*, pp. 225–232, New York: Association for Computing Machinery, 1959. [68]
14. William C. Ridgway, "An Adaptive Logic System with Generalizing Properties," *Stanford Electronics Laboratories Technical Report 1556-1*, Stanford University, Stanford, CA, 1962. [69]
15. For a description of MINOS II, see Alfred E. Brain, George Forsen, David Hall, and Charles Rosen, "A Large, Self-Contained Learning Machine," *Proceedings of the Western Electronic Show and Convention*, 1963. The paper was reprinted as Appendix C of an SRI proposal and is available online at http://www.ai.sri.com/pubs/files/rosen65-esu65-1tech.pdf. [70]
16. For a discussion of shift-register codes and other codes, see W. Peterson, *Error-Correcting Codes*, New York: John Wiley & Sons, 1961. Our technique was reported in A. E. Brain and N. J. Nilsson, "Graphical Data Processing Research Study and Experimental

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

Investigation," Quarterly Progress Report No. 8, p. 11, SRI Report, June 1962; available online at http://www.ai.sri.com/pubs/files/1329.pdf. [71]

17. Robert E. Larsen of SRI suggested using this method. The online encyclopedia Wikipedia has a clear description of dynamic programming. See http://en.wikipedia.org/wiki/Dynamic_programming. [72]
18. The technical details of the complete system are described in two papers: John Munson, "Experiments in the Recognition of Hand-Printed Text: Part I – Character Recognition," and Richard O. Duda and Peter E. Hart, "Experiments in the Recognition of Hand-Printed Text: Part II – Context Analysis," *AFIPS Conference Proceedings*, (of the 1968 Fall Joint Computer Conference), Vol. 33, pp. 1125–1149, Washington, DC: Thompson Book Co., 1968. Additional information can be found in SRI AI Center Technical reports, available online at http://www.ai.sri.com/pubs/files/1343.pdf and http://www.ai.sri.com/pubs/files/1344.pdf. [73]
19. E. Fix and J. L. Hodges Jr., "Discriminatory analysis, nonparametric discrimination," USAF School of Aviation Medicine, Randolph Field, Texas, Project 21-49-004, Report 4, Contract AF41(128)-31, February 1951. See also B. V. Dasarathy (ed.), *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*, Los Alamitos, CA: IEEE Computer Society Press, which is a reprint of 1951 unpublished work of Fix and Hodges. [74]
20. Thomas M. Cover and Peter E. Hart, "Nearest Neighbor Pattern Classification," *IEEE Transactions on Information Theory*, pp. 21–27, January 1967. Available online at http://ieeexplore.ieee.org/iel5/18/22633/01053964.pdf. [74]
21. See M. A. Aizerman, E. M. Braverman, and L. I. Rozonoer, "Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning," *Automation and Remote Control*, Vol. 25, pp. 917–936, 1964, and A. G. Arkadev and E. M. Braverman, *Computers and Pattern Recognition*, (translated from the Russian by W. Turski and J. D. Cowan), Washington, DC: Thompson Book Co., Inc., 1967. [74]
22. George S. Sebestyen, *Decision-Making Processes in Pattern Recognition*, Indianapolis, IN: Macmillan Publishing Co., Inc., 1962. [74]
23. Nils J. Nilsson, *op. cit.* [74]
24. Richard O. Duda and Peter E. Hart, *Pattern Classification and Scene Analysis*, New York: John Wiley & Sons, 1973; updated version: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*, 2nd Edition, New York: John Wiley & Sons, 2000. [74]
25. George Nagy, "State of the Art in Pattern Recognition," *Proceedings of the IEEE*, Vol. 56, No. 5, pp. 836–857, May 1968. [74]
26. See, for example, Joseph K. Hawkins and C. J. Munsey, "An Adaptive System with Direct Optical Input," *Proceedings of the IEEE*, Vol. 55, No. 6, pp. 1084–1085, June 1967. Available online for IEEE members at http://ieeexplore.ieee.org/iel5/5/31078/01446273.pdf?tp=&arnumber=1446273&isnumber=31078. [75]
27. Laveen N. Kanal and Neal C. Randall, "Target Detection in Aerial Photography," paper 8.3, *Proceedings of the 1964 Western Electronics Show and Convention (WESCON)*, Los Angeles, CA, Institute of Radio Engineers (now IEEE), August 25–28, 1964. (Several other papers on pattern recognition were presented at this conference and are contained in the proceedings.) [75]
28. Thomas Harley, personal e-mail communication, July 15, 2007. [76]
29. Laveen N. Kanal and Neal C. Randall, *op. cit.* [76]
30. Laveen Kanal, personal e-mail communication, July 13, 2007. [76]
31. Laveen N. Kanal, "Statistical Methods for Pattern Classification," Philco Report, 1963; originally appeared in T. Harley *et al.*, "Semi-Automatic Imagery Screening Research Study and Experimental Investigation," Philco Reports VO43-2 and VO43-3, Vol. I,

https://doi.org/10.1017/CBO9780511819346.006 Published online by Cambridge University Press

Sec. 6, and Appendix H, prepared for U.S. Army Electronics Research and Development Laboratory under Contract DA-36-039-SC-90742, March 29, 1963. [77]

32. Thomas Harley, personal e-mail communication, July 11, 2007. [77]
33. Laveen N. Kanal (ed.), *Pattern Recognition, Proceedings of the IEEE Workshop on Pattern Recognition*, held at Dorado, Puerto Rico, Washington, DC: Thompson Book Co., 1968. [77]
34. See the Web page at http://tab.computer.org/pamitc/. [78]