M. Henrion, "Uncertainty in Artificial Intelligence: Is Probability Epistemologically and Heuristically Adequate?" in J. Mumpower, ed., *Expert Judgment and Expert Systems,* Springer-Verlag, New York, 1987, pp. 105–130.

E. Horvitz, J. Breese, and M. Henrion, "Decision Theory in Expert Systems and Artificial Intelligence," *Int. J. Approx. Reason.* **2,** 247–302 (1988).

R. Howard and J. Matheson, "Influence Diagrams," in R. Howard and J. Matheson, eds., *Readings on the Principles and Applications of Decision Analysis,* Vol. II, Strategic Decisions Group, Menlo Park, Calif., 1981, pp. 721–762.

R. Johnson, "Independence and Bayesian Updating Methods," in Kanal and Lemmer, eds., 1986, pp. 197–202.

J. Pearl, "Fusion, Propagation, and Structuring in Belief Networks," *Artif. Intell.* **29,** 241–288 (1986).

J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference,* Morgan-Kaufmann, San Mateo, Calif., 1988.

W. Rousseau, *A Method for Computing Probabilities in Complex Situations,* Technical Report 6252-2, Center for Systems Research, Stanford University, Stanford, Calif., 1968.

R. Shachter and D. Heckerman, "Thinking Backward for Knowledge Acquisition," *AI Mag.* **8,** 55–63 (1987).

E. Shortliffe and B. Buchanan, "A Model of Inexact Reasoning in Medicine," *Math. Biosci.* **23,** 351–379 (1975).

A. Tversky and D. Kahneman, "Causal Schemata in Judgments under Uncertainty," in D. Kahneman, P. Slovic, and A. Tversky, eds., *Judgment under Uncertainty: Heuristics and Biases,* Cambridge University Press, New York, 1982.

S. Wright, "Correlation and Causation," *J. Agricult. Res.* **20,** 557–585 (1921).

DAVID HECKERMAN
University of Southern California

This work was supported by the National Cancer Institute under Grant RO1CA51729-01A1.

CHARACTER RECOGNITION

Character recognition, also known as optical character recognition (OCR), is concerned with the automatic conversion of an image of a character, or of characters in running text, into the corresponding symbolic form. The long history of research in this area, some commercial successes, and the continuing need for implementations to handle less restricted forms of text, makes character recognition the most important application area, to date, in machine perception. The ability of humans to read printed text effortlessly is far from matched by today's machines which makes this an important research topic in artificial intelligence.

The processing steps involved in most OCR systems of today are indicated in Figure 1. The image scanning resolution is a function of point size of characters to be recognized (eg, for 8–point characters, 300 dots per inch (dpi) scanning is sufficient). Since text is printed as dark points on light backgrounds (or vice versa), the image is almost always mapped into a binary image, corresponding to the figure-ground dichotomy. Next the layout has to be analyzed, in a process referred to as document analysis, and the words segmented before attempting recognition.

Technical challenges in character recognition arise from three sources:

1. symbols: the set of idealized shapes that can occur, often in a hierarchy where simple symbols are assembled into more complex ones, at several levels of organization.
2. deformation: the range of shape variations that each symbol is allowed to undergo, including geometric transformations (translation, rotation, scaling, stretching, etc) and more complex or time-dependent distortions (eg, due to the biomechanics of handwriting).
3. imaging defects: imperfections in the image due to printing, optics, scanning, spatial quantization, binarization, etc.

Character recognition methods are sometimes specialized to handle subcategories such as digits only. Handwriting and machine print demand somewhat different approaches. Handwriting, particularly cursive script, consists of elongated strokes, whereas machine print consists of regularly spaced blobs. The shapes of characters in handwritten words are often influenced by the context in which they appear. Handwriting recognition has distinct technologies for the on-line and off-line cases; in the on-line case, an electronic surface is used for writing. On-line recognition is simpler than off-line recognition since the temporal data can be translated into stroke information.

The problem of character recognition is a special case of the general problem of reading. While characters occasionally appear in isolation, they usually appear with other characters. Characters group together to form words, words form sentences, sentences form paragraphs, paragraphs form text blocks, and text blocks together with illustrations form document pages, etc. Even though a deformed or degraded character in isolation may be unrecognizable, the context in which the character appears can make the recognition problem simple. The utilization of *a priori* knowledge about the domain of discourse as well as constraints imposed by the surrounding orthography is the main challenge in the development of robust methods. In this article, the discussion of automatic character recognition methods will be divided into two parts: isolated character recognition and word recognition, which includes character recognition in context. Isolated character recognition methods begin with the assumption that an object has been extracted from the surrounding background and it is necessary to assign it into one of a small set of pattern classes such as upper and lower case characters, digits, special symbols, etc. Word recognition involves assigning a compound object representing several characters into a word class. The performance of a given character recognition method is measured not only in terms of correct recognition rate but also in terms of having a low error rate, with the balance being rejected.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

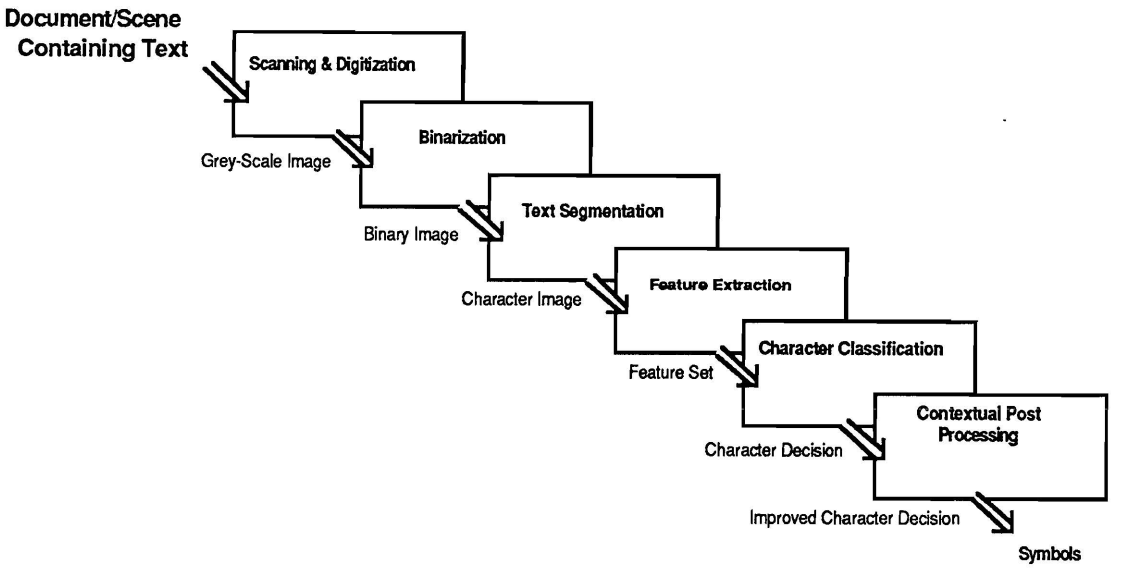


Figure 1. Typical character recognition process.

ISOLATED CHARACTER RECOGNITION

There are many techniques used to associate a symbolic identity with the image of a segmented, or isolated, character. The multitude of fonts in machine printing and the deformations encountered in handwriting make the problem of isolated character recognition a continuing challenge (Fig. 2). This section will describe several methods for the recognition of isolated characters, namely, template matching, discriminant function classifiers based on pixel arrays and structural feature vectors, rule-based analysis of contour feature strings, artificial neural networks, and combination classifiers.

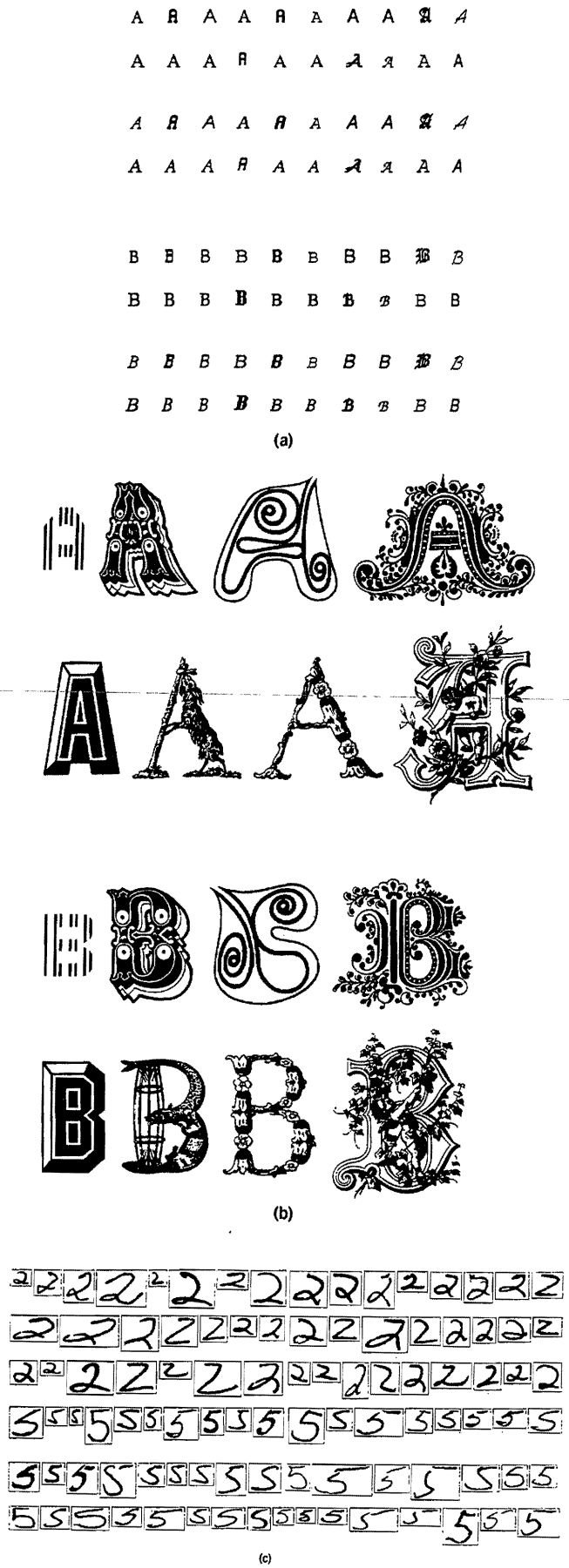
Template Matching

In the process of template matching (qv), the observed pattern is directly compared to templates, or prototypes, representing each class. The classification is according to the best match, or minimum mismatch. Mathematically, the comparison is achieved with a distance measure. The distance between pattern x and the prototype of class C_i is computed by a function $d(x,C_i)$ and x is assigned to the class that minimizes this function. A metric that is useful for patterns having binary-valued features is the Hamming distance, which is the number of features, (typically pixel values) in which the observed pattern differs from the prototype of class C_i. A character recognition example using three prototypes and the Hamming distance is shown in Figure 3.

There are many variations of the template matching concept. Defining a similarity measure instead of a distance measure several different methods are obtained. For example, if n_{ij} is the number of pixels having values i and j in the template and pattern, then $n_{11}/(n_{11}+n_{01})$ is the ratio of the number of correct matches of 1's to the number of 1's in the unknown target pattern. Thus, the procedure ignores matches of 0's and does not penalize incorrect matches. Several other similarity measures for binary matching are also appropriate for character template matching. Some of these weight individual matches and mismatches according to their statistical separability. Template matching is suitable for an application where a limited number of character types have to be recognized. When the number of prototypes is limited, it suffers from a lack of robustness because of a sensitivity to noise in the image and an inability to adapt to differences in character style. It is interesting that from an AI perspective, template matching has been ruled out as an explanation of human performance for similar reasons. However, when a large number of representative prototypes are available, template matching can yield surprisingly good results even with wide variability in patterns. For example, in the problem of handwritten digit recognition using several thousand training templates, correct recognition rates of over 90% have been reported. Such methods also suffer from having to perform a large number of computations, which may be a drawback on serial computers.

Discriminant Function Classifiers

These represent the character as a feature vector x, associate a function $f_i(x)$ with class C_i, and assign x to the class that has the maximum discriminant function value. Some discriminant functions are derived from a statistical (Bayesian) formulation of the problem. The parameters of the discriminant functions are typically estimated from a set of training samples. The polynomial method for character recognition has historically performed well enough to be incorporated into some commercial implementations for recognizing multi-font print. The binary character image is first mapped into a $n \times n$ binary array via character normalization. This image is then represented by a $n^2 = N$ element column vector $v = (v_1, v_2, \ldots, v_N)^t$. Using the components of v as linear terms and products of the components as polynomial terms, an M-element polynomial



feature vector x is constructed by a predefined mapping $\rho(v) = x$. Generally, the components of x are of degree two or less which results in a quadratic feature vector of the form,

$$x = (x_1, x_2, \ldots, x_M)^t = (1, v_1, v_2, \ldots, v_N, v_1{*}v_2, \ldots, v_{M-1}{*}v_M)^t.$$

Not all pixel pairs are typically used, and M tends to be far fewer than $1 + N{*}(N + 1)/2$. One successful implementation for recognizing handwritten digits utilizes $n = 16$, $N = 256$, and $M = 1240$, where the pairs are chosen to be within a small distance from each other (Fig. 4).

Given K classes to be discriminated, based on the polynomial feature vector x, a K-dimensional discriminant vector $d = (d_1, \ldots, d_K)$ is formed. Each of the K discriminant functions d_i is defined to be a linear expression in the components of x,

$$d_i = a_{i1}{*}x_1 + \cdots + a_{iM}{*}x_M, \qquad i = 1, \ldots, K$$

and thus a quadratic polynomial expression in the components of v. The discriminant vector $d = (d_1\, d_2 \ldots d_K)^t$ can therefore be written as $d = A^t{*}x$, where A is a $M \times K$ matrix, whose ith column, $i = 1, \ldots, K$, consists of the elements $a_{i1}, \ldots, a_{iM}$.

To obtain the coefficient matrix A, which gives the confidences to enable classification, the least mean square approach is chosen. Under the assumption that C training characters are available, a $C \times K$ objective matrix Y is defined as $Y = (y_1 \ldots y_K)^t$, where y_i is a binary vector indicating class membership of training character i. Similarly, let X be a $C \times M$ matrix such that row vector i equals $\rho(v)$, where v is the vector representation of the ith training image. The training proceeds by minimizing the mean–square deviation between the actual class membership matrix Y and the estimated class membership matrix given by XA. The minimization of $E\{|Y - XA|\}$ leads to the requirement that $E\{XX^t\}A = E\{XY^t\}$; it follows as a necessary condition for minimizing the norm. Consequently, the coefficient matrix A can be probabilistically approximated by $A = (XX^t)^{-1}XY^t$. After the coefficient matrix A is computed, the polynomial discriminant function can be evaluated for a test image, v. This evaluation consists of calculating the discriminant vector d and choosing the class corresponding to the largest component in this vector.

Structural Feature Vectors

Another approach is to extract structural features and represent them as a feature vector and use statistically determined discriminant functions. When asked to de-

Figure 2. The presence of many different fonts and deformations in machine-printed and handwritten characters makes the character recognition task challenging: (*a*) examples of character shapes commonly used; (*b*) decorative fonts used in books [from Haab and Haettenschweiler, 1972]; and (*c*) handwritten numerals extracted from ZIP Codes on envelopes.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

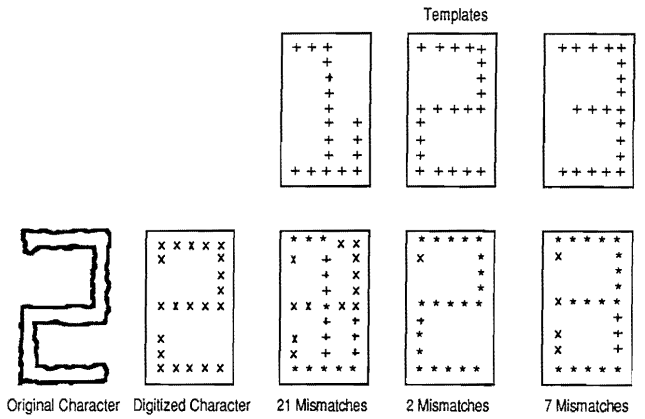


Figure 3. Template matching using the Hamming distance. All points of the digitized character (ie, features) are compared with the corresponding points in each template. If the two are not the same (ie, both 0 or both 1), a mismatch, or distance 1, is counted. Here, the second template is selected as a result of minimum mismatch.

scribe an alphanumeric character, a person will most likely use a structural description. For example, an uppercase letter A has the following description: two straight lines meeting with a sharp point at the top, and a third line crossing the two at approximately their midpoint. The basis of any structural technique is the representation of the character icon, ie, figure, with a set of feature primitives. These features must be able to describe each figure encountered as well as discriminate between them. The feature set can include descriptions for endpoints, line segments, arcs, curves, and crossings. The image could be normalized, smoothed, filtered, thinned, or any number of other operations applied to reduce noise and simplify feature extraction.

One such character recognition system consists of three stages: feature extraction, feature parameterization and statistical feature classification. The feature extractor looks for five types of features in a character: stroke, hole, concavity, cross point and end point (Fig. 5). The presence of these features can be detected by first determining a line adjacency graph (LAG). A LAG is obtained from a run-length representation of a binary image; its nodes correspond to runs of object pixels and its edges correspond to adjacent runs.

Feature parameterization is used to map the detected features into a binary feature vector. The parameter spaces may have different dimensions depending on the amount of information obtained from a feature. For example, the stroke parameter space is 4–D representing a 4–tuple $\langle x, y, r, \theta \rangle$ where $\langle x, y \rangle$ is the center of the stroke, r is the length, and θ is the angle formed with the positive x–axis. The cross point parameter space is two-dimensional, representing its location $\langle x, y \rangle$. Each dimension of a parameter space is divided into five equal-sized intervals; therefore, the stroke parameter space is partitioned into 625 hypercubes of equal size. By using a similar partition scheme, the cross point parameter space is partitioned into 25 squares. Each of these hypercubes and squares has a corresponding feature in the feature vector, ie, 625 features for strokes and 25 features for cross points. If the space of a hypercube has a stroke mapping, its corresponding feature in the vector will be set to one. This partition-

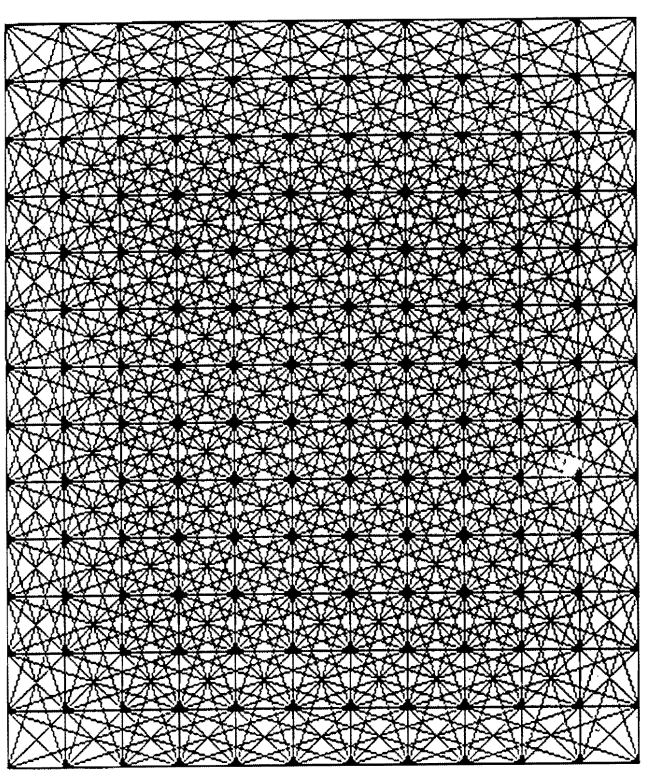


Figure 4. Pixel pairs used in polynomial discriminant classification. Each pair of pixels used in the feature vector v is indicated by a line segment in the 16×16 matrix.

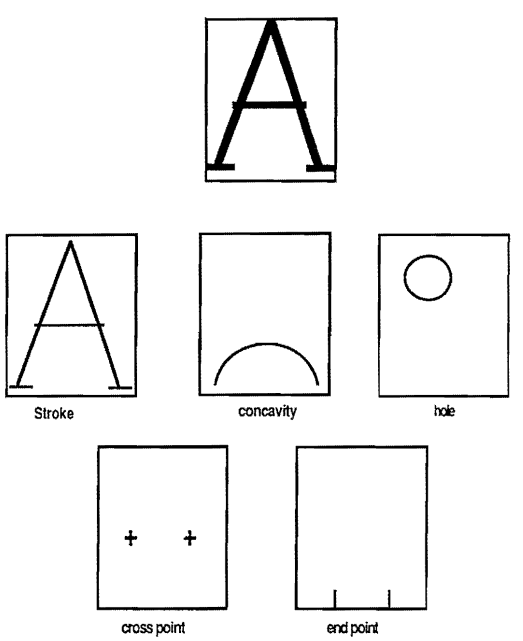


Figure 5. Structured features in characters. The five features are extracted using a line adjacency graph.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

ing method is feasible based on the assumption that characters in the same class should have features which tend to map closely together in the parameter spaces.

Classification of the binary feature vector into character classes is done using a Bayes classifier which requires estimating the class-conditional statistical distributions of the binary feature vectors. Using the simplifying assumption that the feature vector elements are statistically independent, the method and its variations have been demonstrated to achieve very high recognition rates for multifont machine printed characters when the font mixture consists of those most often used in printed documents.

Stroke Analysis

A character represented by strokes can be assigned to a class by using one of several AI methodologies. Strokes are somewhat analogous to the strokes made by a person when a letter is drawn. Rule-based systems (qv) or semantic networks (qv) can be used not only to encode knowledge about strokes but also to direct the analysis. One such approach uses a network made up of many types of arcs and nodes, only a small portion of which is described here.

The subset arc *s* states that the node at its tail has the property at its head. Terminal nodes represent a primitive theoretic property about the image. A node with outgoing *s* and *p* arcs represents the largest subset of the set at the head of the *s* arc with the property at the head of the *p* arc. A node with more than one outgoing *s* arc represents the intersection of the sets at the heads of the *s* arcs. The example description of an upper-case 'F,' which consists of a long vertical stroke and two short horizontal strokes (Fig. 6) illustrates these concepts. Node 2 represents the subset of all the input with a major vertical line on the left. Note that this includes many letters such as B, D, E, F, H, and so on. Nodes 4 and 5 represent the strokes near the top and middle of the major vertical line, and node 6 represents the concept that there is no other stroke near its bottom. Nodes 7 and 8 represent the concept that the horizontal line near the top of the major vertical line is on its top and to its right. Nodes 9 and 10 represent a similar concept for the horizontal line near the middle of the character. Finally, node 11 represents F as the intersection of the sets represented by nodes 6, 7, 8, 9, and 10.

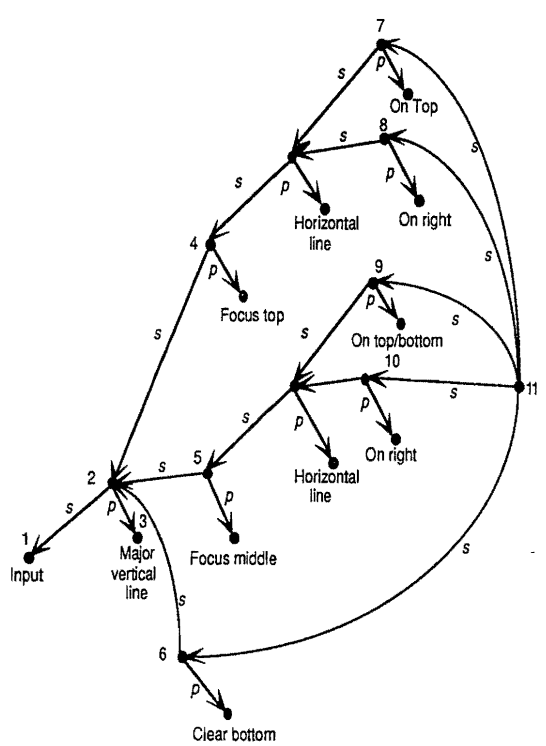


Figure 6. Representation of 'F' in a semantic network.

This is not only a description of F but also a plan to follow for its recognition. The terminal node input reads a character image and begins recognition. The major vertical line is then tested for, and if it is located, additional tests are carried out to locate the appropriately oriented strokes near the top and middle of the image. If any of these tests fail, backtraining takes place and the presence of primitives from other characters is determined. For example, in the complete system, if the major vertical line cannot be located, a loop, such as occurs in an O, is tested for next. Advantages of this approach include its use of a more flexible control structure than most traditional methods. Disadvantages include its application to a limited alphabet of only 20 uppercase letters. Although many cases of distorted input were recognized correctly, the robustness of this technique remains unclear.

Contour Analysis

Another structural approach is to analyze the contour of a character. One such system for recognizing handwritten digits uses structural features based on the curvatures around the inner and outer contours of the figure. The primitive feature set consists of eight features: five concave features (three simple arc-like structures of varying curvature, and two endpoints), and three convex features of varying curvature. Associated with each feature is a direction quantized to eight compass points, and a location quantized to a 4 × 4 Cartesian grid with the origin in the upper left (Fig. 7).

The contour of the figure is first represented in the form of a chain-code; the chain-code is an eight direction code following the contour such that a change of one unit in the positive direction of the chain-code represents a 45–degree turn in the positive direction, likewise a negative change of one chain-code unit represents a negative 45–degree turn. The chain-code contour trace is converted to a curvature trace around the figure. The relative degree of curvature for each point along the original image is calculated. Local variations and noise are filtered by looking at the preceding and following points when calculating the curvature at the current point. Points along the image contour where the degree of curvature changes are the places where features are defined. A rule base is used to classify the extracted feature string. The rule base is designed as a decision tree, where each successive branch narrows down the possible candidates that can match the feature string. The rules are generalized to have a one-to-many relationship. Each class can be fully covered by only a few rules. In the complete system there are 130 rules for all ten classes of digits.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

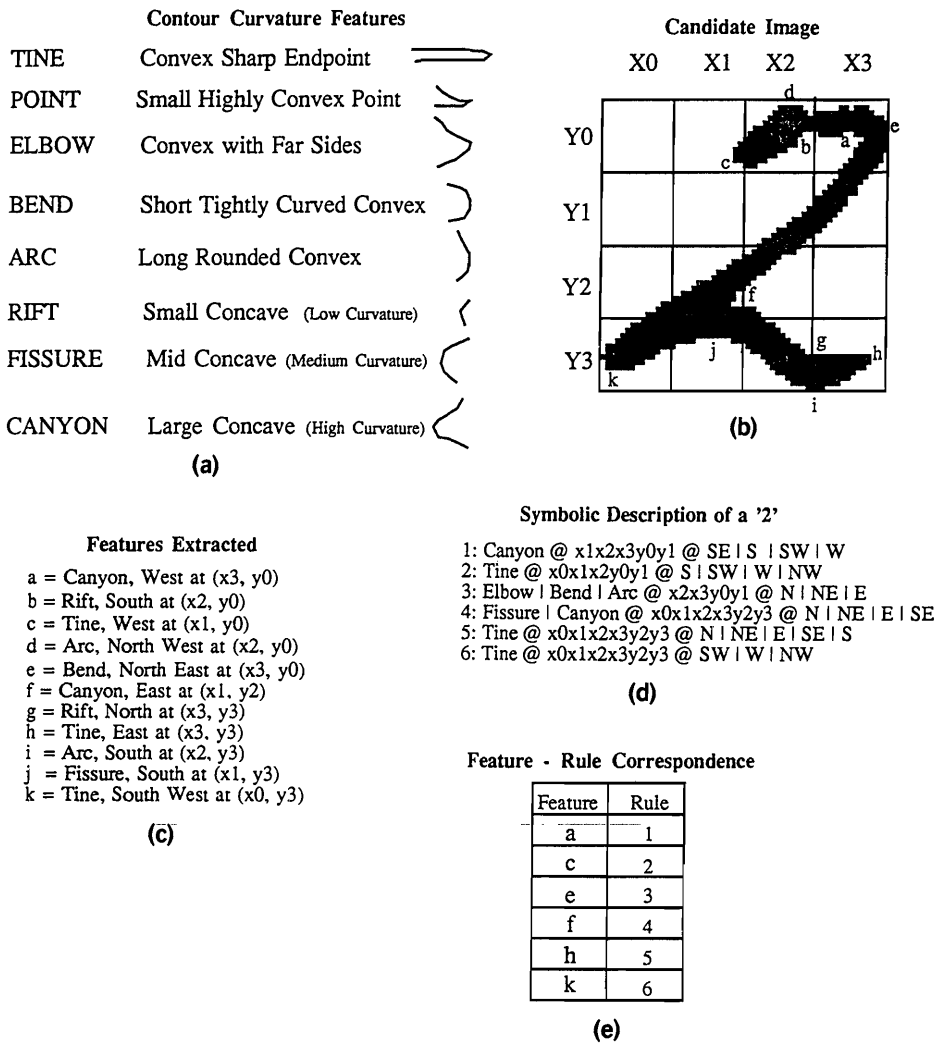


Figure 7. Contour feature analysis: (*a*) contour features; (*b*) image in a 4×4 bounding box; (*c*) features extracted from (*b*); (*d*) rules in database that need to be satisfied for a digit 2; and (*e*) extracted features that match the six rules.

Phenomenological Attributes

One approach is to develop a character description scheme based on human experiments. The idea is that if features people use to recognize letters are properly described and used in a character recognition algorithm, the algorithm should perform as well as a human.

Three levels of human description are distinguished: functional, skeletal and physical. The abstract or functional level defines the essential meaning of letters in terms of a set of features or functional attributes. These are determined by a procedure that includes the presentation of ambiguous characters to human subjects and the use of their responses to determine the functional attribute at the pivot of the ambiguity. An intermediate skeletal level provides a description that distinguishes characters from everything else as well as from characters in other families of type fonts. This level of description is implemented as a set of graphs, one for each character in each font family. The lowest physical level in this hierarchy is where actual character images are placed.

This representational system can be used for recognition in several ways. Functional descriptions can be used directly if procedures are developed to detect the features they specify. This would be appropriate if it were known *a priori* that only character images (not graphics, halftones, etc) will be presented to a recognition system since functional descriptions can only distinguish one character from another. Otherwise, a skeletal representation would be a better choice since it can discriminate characters from everything else. This corresponds more closely to the way people read letters; however, its font-specific nature loses some robustness. The main advantage of this line of research is its acknowledgement of the complexity of the character recognition task and the necessity to incorporate knowledge about human character recognition in algorithms.

Neural Networks

Artificial neural networks provide an alternative methodology to the classification of patterns represented as feature vectors. The backpropagation model has been used most often in character recognition applications. Experiments with binary pixel arrays and feature vectors represented as binary feature vectors with a three-layer network have shown performance level comparable to first-order Bayesian linear discriminant functions for handwritten digits and characters.

Combination of Classifiers

It is found in character recognition practice that each of several approaches to character recognition perform well with different writing styles. For instance, a polynomial classifier, which is correlated with template matching, does well on broken characters that a structural approach fails on. On the other hand, the structural approach is tolerant to wide variations in strokes. Thus the approach of combining several different approaches with a decision tree can yield performance higher than using any individual approach.

WORD RECOGNITION

Several approaches to utilize context in recognizing characters are known. Contextual information is usually in the form of a lexicon of acceptable words, n-grams (legal letter combinations), or letter transition probabilities. Three distinct approaches to the use of context in character recognition can be identified. The first approach, referred to as *contextual postprocessing*, is a three-step approach. First, the word image is segmented into character images. Second, the segmented characters are recognized by using an isolated character recognition technique. Third, the resulting word is corrected, eg, by comparing to each word in a lexicon to determine a match. If none is found then a distance measure between the two words, eg, a Levenshtein metric, which measures the number of weighted editing operations, such as substitution, insertion and deletion, necessary to transform one word to the other, is used to find the closest word.

The second approach, referred to as *character recognition in context*, is a two-step approach, where contextual information is used in the process of recognizing individual characters. As with the previous approach, first the word image is segmented into character images. In the second step, features are extracted for each character image and the classification into a word is done by examining the entire (compound) feature set. A simplification of the second step is to weight the choices for a given character image by their frequencies of occurrence in the text and to eliminate the unlikely choices based on the neighboring character images and associated class decisions.

The third approach, *word-shape analysis*, is a one-step approach that bypasses segmentation. Features are extracted from the entire word and classification is attempted using a lexicon organized by word features. A simple set of features is used in a first level analysis to select a neighborhood of words and a more detailed analysis discriminates between a small subset of character classes. When the lexicon is small, there exists a strong top-down constraint on the word recognition problem. In such a case it is only necessary to compute those (bottom-up) features that discriminate between words. Thus the process becomes one of hypothesis-testing, or verification, instead of recognition.

Contextual Postprocessing

These techniques utilize knowledge at the word level to correct errors in character recognition. The methods use information about other characters that have been recognized in a word as well as knowledge about the text in which the word occurs to carry out this task. Typically, the knowledge about the text takes the form of a dictionary, a list of words that occur in the text. For example, a character recognition algorithm may not be able to reliably distinguish between a 'u' and a 'v' in the second position of *q*ote*. A contextual postprocessing technique would determine that 'u' is correct since it is very unlikely that *qvote* would be in an English language dictionary.

Methods of contextual postprocessing differ in their manner of representing the lexicon. Some methods use an approximation to a dictionary that often takes the form of probabilities of letter transitions. Other approaches use an exact representation such as a serial representation, a hash table, or a graph structure.

Binary n-Grams. The method of binary n-grams is one approach that uses an approximate representation. In this method a set of n-dimensional binary arrays represents a dictionary. Each of the dimensions can take on one of m values, where m is the number of letters in the alphabet, and the binary data in the matrix indicates whether the letter combination that specifies its location occurs in the dictionary. A 1 (logical true value) indicates the occurrence of the letter combination, and a 0 (logical false value) indicates its nonoccurrence. Typically, n values (position indices) are associated with each array. These tell the positions in which the letter combinations occur within each dictionary word. This method can be used to detect as well as correct errors in the output of a character recognition algorithm. Many error types can be handled by this approach; however, only the substitution of one character for another is described here since this is the most common error in character recognition. A word is considered correct only if the intersection of all its appropriate n-gram entries is nonzero. Otherwise, it must contain an error. The position of the error is determined by intersecting the sets of position indices that returned zero in the detection phase. If there is only a single position in this intersection, it contains the error. Vectors from all the arrays that involve that position, given that the other positions are correct, are then intersected. If there is only a single letter in that intersection, it can be substituted in the error position to produce a word that is acceptable to the n-gram arrays.

An example illustrates these points. Figure 8 shows a dictionary of the three-letter words {cat, cot, tot}. The three binary digram ($n = 2$) arrays for this dictionary are also shown.

If a character recognition technique outputs the string 'coo,' detection of the error would be done by

dictionary: {*cat, cot, tot*}:

$d_{1,2}$

	a	c	o	t
a	0	0	0	0
c	1	0	1	0
o	0	0	0	0
t	0	0	1	0

$d_{1,3}$

	a	c	o	t
a	0	0	0	1
c	0	0	0	0
o	0	0	0	1
t	0	0	0	0

$d_{2,3}$

	a	c	o	t
a	0	0	0	0
c	0	0	0	1
o	0	0	0	0
t	0	0	0	1

Figure 8. Example dictionary and its representation by binary digram arrays.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

$$d_{1,2}(c, 0) \cap d_{1,3}(c, 0) \cap d_{2,3}(0, 0).$$

This would return 0 from both $d_{1,3}$ and $d_{2,3}$. Since the intersection of {1, 3} and {2, 3} yields {3}, correction is done by intersecting the vectors:

$$d_{1,3}(c, *) \cap d_{2,3}(0, *).$$

The resulting vector has only one nonzero element, corresponding to a '*t*.' Therefore, coo is corrected to cot. This short example illustrates several of the advantages and disadvantages of this method. The computations to locate and correct errors are relatively simple and involve only binary comparisons. Hence they can be economically implemented. However, the potential storage costs are also apparent by observation of the sparseness of the arrays. This is a major weakness of this method.

Character Recognition in Context

The problem of assigning a set of character images to a symbol string is addressed in the area of pattern recognition known as compound decision theory. The problem is formulated as follows:

The observed sequence of patterns, or vectors with feature elements, is

$$X = x_1, \ldots, x_m.$$

Each pattern x_i is to be assigned to one symbol, or character class, in the set

$$L = \{L_1, L_2, \ldots, L_r\}.$$

Since there are r possible choices for each pattern, there are r^m possible assignments for X. The goal is to choose that assignment

$$W_j = w_{j1}, \ldots, w_{jm}, \qquad w_{ij} \in L$$

which has the maximum probability over all possible assignments: $j = 1, \ldots, r^m$. Estimating all the joint probabilities to perform the exact probability computation is impractical. One simplifying assumption is to assume that a character icon string arises from a Markov source. Assuming a first-order Markov source, the task of determining the joint probability of a given word reduces to a product involving first-order transitional probabilities between letters and the class-conditional, or confusion, probabilities associated with each pattern. The word with the highest probability is efficiently computed by a method known as the Viterbi algorithm; it involves $(m-1) \times r^2$ computations instead of r^m computations.

The Viterbi algorithm yields the most likely letter combination based on the probabilities assumed but can yield a string that is absent in the lexicon. The Dictionary Viterbi Algorithm (DVA) is a technique that brings in a dictionary to play a role in the search; it can be used for either contextual postprocessing or for character recognition in context. It uses an exact representation for a dictionary. A graph of letter alternatives (a *trellis*) produced by a segmentation of the word into characters is first set up. An example of such a graph in the contextual post-processing mode is shown in Figure 9. The string at the top of the graph is assumed to be input from a character recognition algorithm and {a, c, o, t} is the alphabet of the source text. Each node is labeled with a letter of the alphabet and has a cost associated with it that is the probability that the letter on the node is confused with the corresponding letter of the input word. Each arc in the graph also has a cost associated with it that is the probability that the letter at its head follows the letter at its tail in the source text. A path is traced through this graph in a left-to-right manner one column at a time. The costs of all the ways of reaching a node from nodes in the previous column are computed, and only the partial path with the best cost is retained. Each time the cost of an arc is evaluated, the presence in the dictionary of the substring composed of the letters on the path from the beginning of the graph to the node at the head of the arc is determined. If it does not occur in the dictionary, this partial path is discarded from future consideration. This evaluation process is performed once for every node in the graph of alternatives. The letters on the best path from the first node to the last node are output.

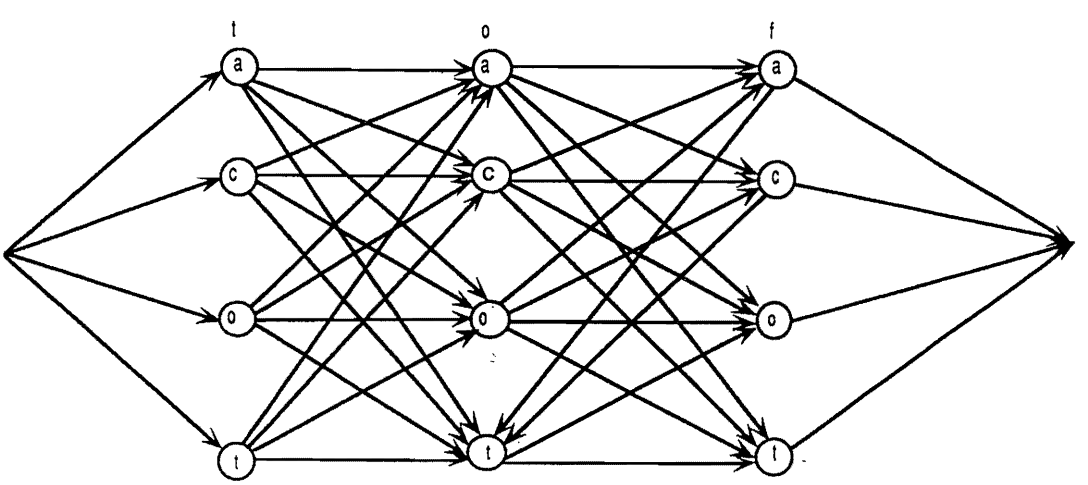


Figure 9. Example graph of alternatives for DVA.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

The simultaneous searching of the graph of alternatives and the dictionary is done with a data structure for the dictionary known as a *trie*. An example for the dictionary {cat, cot, tot} is shown in Figure 10. If the graph of alternatives shown in the previous figure was evaluated with this trie, only the *c* and *t* nodes in the first column would be considered since these are the only two letters at the first level of the trie. At the next step only one path to each of the *a* and *o* nodes in the second column would be retained. These partial paths would most likely be *ca* and *to*. At the next step, only *cat* and *tot* would be considered because of the absence of any other paths in the trie. Most probably *tot* would be output because it is most like the input. The DVA, as other techniques that use an exact representation for a dictionary, is more accurate than methods that use an approximate representation. However, methods based on exact representations incur additional processing costs. The acceptability of these costs should be determined by the application and the need for improved performance.

Interactive Activation Model. A cognitive theory of how information in human memory could affect, top-down, the course of perceptual recognition is useful as a computational model of character recognition in context. This cognitive theory, known as the interactive activation model (IAM), posits three levels of representation arranged in a hierarchy: features, letters, and words. As illustrated in Figure 11, each level consists of a number of nodes at various states of activation for the entities relevant to that level. Each node is connected to a large number of other nodes from which it can receive either excitatory inputs (designated by an arrow at the end of the connection in Fig. 11), which raise its activation level, or inhibitory inputs (designated by a small disk in Fig. 11), which lower its activation level. Each node, in turn, transmits its activation as excitatory or inhibitory inputs to other nodes.

The presentation of a letter (actually, the letter's features) causes excitation of the nodes consistent with that letter's features and inhibition of the nodes for those features that are inconsistent with that letter. The nodes whose activity has been increased transmit their excitation by increasing the activation of letter nodes that contain those features. Similarly, the activation of the letter nodes results in excitation of word nodes that contain those letters and inhibition of word nodes that do not contain those letters. At all levels there is strong intralevel

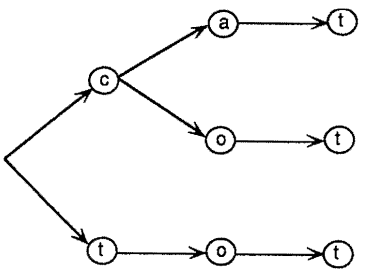


Figure 10. Trie representation for {cat, cot, tot}.

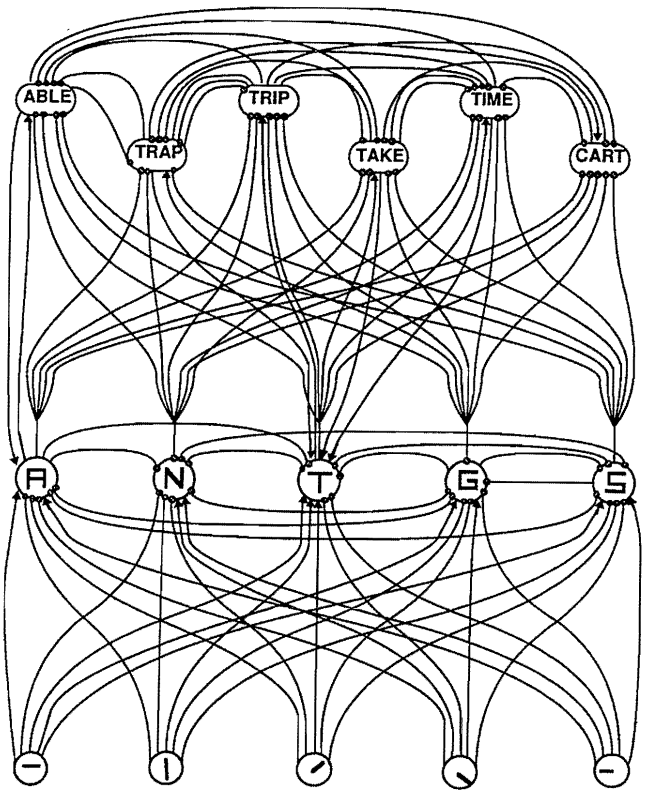


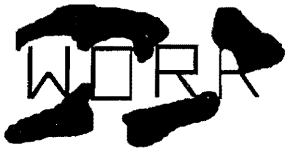
Figure 11. The three levels of the interactive activation model for the word superiority effect.

inhibition. Each node at a given level inhibits the other nodes at that level. There is also top-down excitation. Activity at the word level can excite or inhibit activity at the letter level.

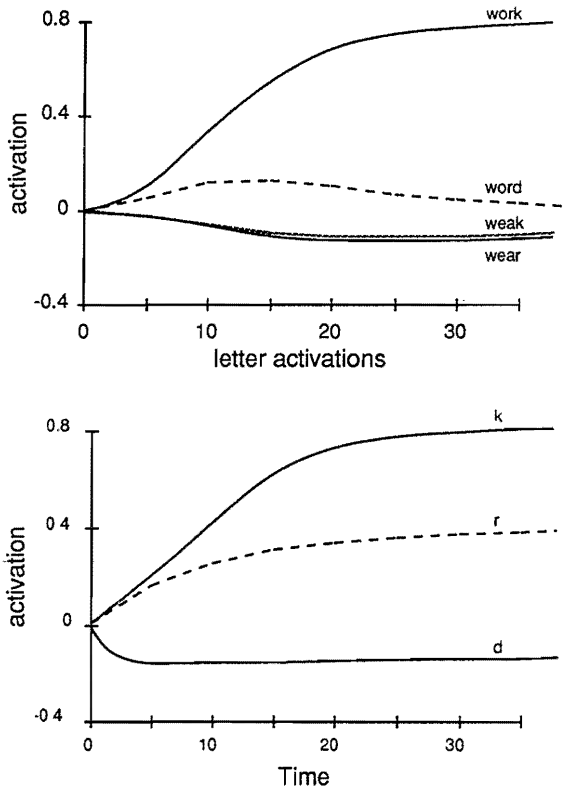
We can now trace the time course of activation of a given node as a word, WORK for example, is presented. Assume that the subject successfully detected the first three letters, WOR, but detected only some of the features of the fourth letter, as indicated in Figure 12. Initially there is an increase in the activation level of those letters consistent with the features actually detected; we are only considering the letter nodes corresponding to the fourth position in the word. These would be R and K. These nodes transmit their excitation to the word level. Although WORK can benefit from activation of the K node, there is no word WORR to receive activation from R. As the activation of WORK increases, it starts to excite, top-down, the K node and inhibit the R node. R starts to weaken and the activation level of K grows until it clearly exceeds the activation level of R.

We can now see how IAM handles the major phenomenon of the word superiority effect as well as the advantage of a letter within a word over the letter itself. A nonword would not have a node at the word level. Consequently, there would be less chance for a letter in that string to benefit from top-down activation. It is possible, however, that words sharing letters with the nonword might generate, through their partial activation, some top-down facilitation from the word level.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.



(a)



(b)

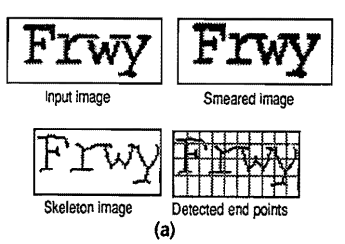
Figure 12. The interactive activation model's characterization of the activation of the letter 'K' in the fourth position, given that WORK was presented and that the features extracted were as shown in (*a*).

Word Shape Analysis

This is an approach for word recognition that is independent of character segmentation. This method is applicable to degraded images of machine-printed words where characters may be broken or touching. Word shape, such as the pattern of ascenders, descenders, and normal-height characters in a lower-case word, is a visual cue that has been known for many years to be useful for word recognition by humans. Several alternative representations for word shape have been explored, and a representation has been found that produces a small search space in a large dictionary. This representation is based on features that can be reliably extracted from word images and does not require the segmentation of words into characters. This avoids a pitfall of current reading algorithms and more closely reflects the way visual information is used in the early stages of word recognition by humans. The process receives an image of a word as input as well as a lexicon that is assumed to contain the word. A set of global and local shape features are extracted from the image, and registered with reference to a global coordinate frame. A set of highly specialized classifiers are used to match different subsets of the extracted shape features, and produce different rankings of the words in the input lexicon. A combination strategy is applied to produce a consensus ranking. The words with the highest ranks are output as a neighborhood containing the word in the image.

One such method utilizes 41 local feature sets. Some feature sets contain only one feature whereas others involve several. The specific feature sets were: stroke distribution (1), edges (9), end points of skeleton images (6), and letter shape features (19). The letter shape features include: vertical strokes, ascender, descender and short strokes; horizontal strokes, single horizontal strokes, two aligned horizontal strokes and two aligned vertical strokes; diagonal strokes, with positive and negative slopes; curves, left bent, right bent arcs; and topological features, holes, dots, bridges between strokes. An example involving computing six feature sets, that of end points, and the neighborhoods of words generated are shown in Figure 12. Six different classifiers (C14–C18) generate six different neighborhoods from a dictionary.

This method is an algorithmic model for a theory of human visual processing. The theory proposes that many features, extracted in parallel across an image, are selectively and dynamically combined to form higher level entities, which contribute to recognition of the object in the image. These events may be attended by cognitive processes. The method has been successfully applied to a da-



(a)

$allumulululuddu$		[endpoint feature symbolic string]
a	0	[location: ascender region endpoints]
d	78	[location: descender region endpoints]
m	1	[location: middle region endpoints]
u	1235679	[location: upper middle region endpoints]
l	012456	[location: lower middle region endpoints]

(b)

C14 endpt sym	C15 endpt a	C16 endpt d	C17 endpt m	C18 endpt u	C19 endpt l	Combined
Lazy	Acres	Emily	EMILY	Calais	Encore	*Frwy*
Pkway	Ave	Freewy	Emily	Fwy	*Frwy*	Emily
AVENU	Aven	Frway	Encore	Ways	PATITO	Pkwy
ROADS	Avenu	*Frwy*	FAR	CALAIS	Pkwy	Boul
FWY	Avenue	Fwy	FERN	FRWAY	WAYS	Prky
Freewy	Avn	Lazy	FLOYD	*Frwy*	AVENU	Postal
Frway	Avnue	Lbj	FRWAY	PKWAY	Alpha	Point
Drive	Driv	Parkwy	FRWY	PKWYS	Frway	Pkway
Frwy	Drive	Pkway	FWY	AVNUE	PALDAO	Drive
AVEN	Drives	Pkwy	Far	EMILY	PRADO	Fwy

(c)

Figure 13. Word-shape feature computation: (*a*) word image with preprocessing operations; (*b*) six end-point feature strings; (*c*) neighborhood of words (top ten) generated by end-point classifiers, and neighborhood of words generated by overall classifier.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

tabase of postal word images scanned on a postal OCR. The method has been proposed to be integrated with an OCR based word recognizer to achieve a robust word recognition system. The resulting system is to recognize words in various positions in an address block image so that contextual address information can be utilized to improve the currently achievable level of sort.

APPLICATIONS

Applications of automatic character recognition with economic importance include the reading of bank checks and forms, serial numbers (eg, of machine parts, automobiles), postal addresses on envelopes, and engineering drawings and maps, in addition to reading machines for the blind.

Small desktop character readers that can typically recognize up to six fonts and medium-sized character readers that can recognize a wide range of character fonts are now commercially available. The United States and other countries have installed large postal address-reading machines that must meet more stringent performance requirements than most other character readers. The performance of all these machines is controlled by many constraints. Deviations from these constraints can cause a large deterioration in performance. In most cases individual characters must not touch one another, and text must be clearly printed in dark ink on a lightly colored background. In some units the location of individual characters must fall within prespecified limits. Even the most versatile machines require that characters be unsmudged and that adjacent characters not touch one another. Furthermore, multifont capability is often achieved by requiring an operator to train the machine on new fonts. This constraint frequently causes the machine to misrecognize text printed in a font that it has not previously seen.

The mere presence of such constraints in even the most sophisticated reading machines illustrates that the ability to read text automatically with the same fluency as a human remains an unachieved goal. This is further evidenced by the performance of postal address-reading machines that have been the subject of much research and development and are designed to read relatively unconstrained text. These machines can correctly read over 90% of the addresses that appear on machine-printed first-class mail. However, they can only read about 34% of the addresses on mail from collection boxes. Overall, 62% of the addresses on mail processed by postal reading machines are correctly recognized. These percentages are based on mail samples that were readable by a human operator. This shows that even the most expensive commercial equipment is not nearly as fluent as a human reader. Obviously much work is needed if a program is to reach levels of human competence.

CONCLUSIONS

A high degree of success can be achieved in printed character recognition, particularly with high quality printed documents that do not use too many fonts. However, generalizations to handle a large number of fonts, decorative fonts, handwriting (hand-printing as well as cursive writing), and degraded text (such as that which is touching or has broken characters), continues to pose difficult problems.

BIBLIOGRAPHIC REMARKS

Character recognition is a topic that has been investigated for more than 30 years. An early influential paper was due to Bledsoe and Browning (1959). Functional components of optical scanners are discussed by Nagy (1983). The effect of sampling rate on character recognition is discussed by Pavlidis (1983). An overview of document analysis is given by Srihari and Zack (1986). The use of image defect models in algorithm development for character recognition was proposed by Baird (1990).

For an introduction to Bayes decision theory and binary feature vector classification, the reader is referred to the text by Duda and Hart (1973). A recent survey of binary template matching is given by Tubbs (1989). The polynomial discriminant method has been shown to be highly successful for postal address reading by Schurmann and co-workers (1978). The structural feature extraction method, due to Baird (1988) and Lee and co-workers (1990), are based on the line adjacency graph analysis of Pavlidis (1986). The contour-based method has its origins in the work of D'Amato and co-workers (1982). The semantic network approach to represent knowledge of character shapes was given by Krumme (1979). The use of multiple classifiers in character recognition is discussed by Hull and co-workers (1990) and Srihari and co-workers (1989).

There exists a large number of papers on neural network approaches to character recognition. An empirical evaluation of different neural network models for handwritten digit recognition was given by Pawlicki and co-workers (1988), and a performance comparison of back-propagation and first-order Bayesian methods were given by Lee and co-workers (1990). A high-performance neural network chip for handwritten digit recognition is described by Graf, Jackel, and Hubbard (1988).

The Viterbi algorithm (Viterbi, 1967) was adapted to text recognition by Neuhoff (1975) and later by Shinghal and Toussaint (1979a). The binary n-gram approach to lexical representation was explored by Hanson, Riseman, and Fisher (1976), which was experimentally compared to the Viterbi approach by Hull and Srihari (1982). Other representations of lexical knowledge include: serial representation (Shinghal and Toussaint, 1979b), a hash table (Doster, 1977) and a graph structure (Hull and co-workers, 1983). The contextual post-processing approach was rejected as a theory of human performance by Cattell (1886). The DVA was proposed by Srihari, Hull, and Choudhari (1983).

The recognition of off-line cursive handwritten words using probabilistic models and a trie were explored by Bozinovic and Srihari (1989). Some interesting similarities exist between the relaxation-based word recognition system of Hayes (1979) and the connectionist theory (interactive activation model) of word perception proposed by

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

McClelland and Rumelhart (1981). A preliminary study of word perception by humans was carried out by Brady (1981). The influence of syntax and semantics in reading handprinted FORTRAN coding sheets was explored by Brady and Wielinga (1978). The word shape approach, which has a cognitive basis (Haber and Haber, 1981), was developed by Hull (1988) and explored further in the context of reading postal address words by Ho, Hull, and Srihari (1990).

BIBLIOGRAPHY

H. S. Baird, "Feature Identification for Hybrid Structural/Statistical Pattern Classification," *Computer Vision, Graphics and Image Processing* **42,** 318–333 (1988).

H. S. Baird, "Document Image Defect Models," *Proceedings of the Workshop on Syntactic and Structural Pattern Recognition,* Murray Hill, N.J., 1990, pp. 38–46.

W. W. Bledsoe and I. Browning, "Pattern Recognition and Reading by Machine," *Proc. Eastern J. Comput. Conf.* **16,** 225–232 (1959).

R. Bozinovic and S. N. Srihari, "Off-line Cursive Script Word Recognition," *IEEE Trans. Patt. Anal. Mach. Intel.* **PAMI-11,** 68–83 (1989).

J. M. Brady and B. J. Wielinga, "Reading the Writing on the Wall," in A. R. Hanson and E. Riseman, eds., *Computer Vision Systems,* Academic Press, New York, 1978, pp. 283–299.

M. Brady, "Toward a Computational Theory of Early Visual Processing in Reading," *Visible Lang.* **XV**(2), 183–215 (Spring 1981).

J. K. Cattel, "The Time it Takes to See and Name Objects," *Mind* **11,** 63–65 (1886).

C. H. Cox, III, P. Coueignoux, B. Blesser, and M. Eden, "Skeletons: A Link Between Theoretical and Physical Letter Descriptions," *Patt. Recog.* **15**(1), 11–22 (1982).

D. D'Amato, L. Pintsov, H. Koay, D. Stone, J. Tan, K. Tuttle, and D. Buck, "High Speed Pattern Recognition System for Alphanumeric Handprinted Characters," *Proceedings of the IEEE-CS Conference on Pattern Recognition and Image Processing,* Las Vegas, Nev., July 1982, pp. 165–170.

W. Doster, "Contextual Postprocessing System for Coorporation With A Multiple-choice Character Recognition System," *IEEE Trans. Comput.* **C–26**(11) (November 1977).

R. O Duda and P. E. Hart, *Pattern Classification and Scene Analysis,* Wiley-Interscience, 1973.

H. P. Graf, L. D. Jackel, and W. E. Hubbard, "VLSI Implementation of a Neural Network Model," *IEEE Computer,* 41–49 (March 1988).

A. Haab and W. Haettenschweiler, *Lettera 4,* Hastings House Publisher, New York, 1972.

R. N. Haber and L. R. Haber, "Visual Components of the Reading Process," *Visible Lang.* **XV**(2), 147–181 (1981).

A. R. Hanson, E. Riseman, and E. G. Fisher, "Context in Word Recognition," *Patt. Recog.* **8,** 35–45 (1976).

K. C. Hayes, Jr., "Reading Handwritten Words Using Hierarchical Relaxation," Ph.D. dissertation, TR–783, University of Maryland, College Park, 1979.

T. K. Ho, J. J. Hull, and S. N. Srihari, "Combination of Structural Classifiers," *Proceedings of the Workshop on Syntactic and Structural Pattern Recognition,* Murray Hill, N.J., 1990, pp. 123–136.

J. J. Hull, "A Computational Theory of Visual Word Recognition," Ph.D. dissertation, SUNY at Buffalo, NY, 1988.

J. J. Hull and S. N. Srihari, "Experiments in Text Recognition With Binary n-Gram and Viterbi Algorithms," *IEEE Trans. Patt. Anal. Mach. Intell.* **PAMI-4**(5), 520–530 (September 1982).

J. J. Hull, S. N. Srihari, and R. Choudhari, "An Integrated Algorithm for Text Recognition: Comparison with a Cascaded Algorithm," *IEEE Trans. Patt. Anal. Mach. Intell.* **PAMI-5**(4), 384–395 (July 1983).

J. J. Hull, A. Commike, and T. K. Ho, "Multiple Algorithms for Handwritten Character Recognition," in C. Y. Suen, ed., *Frontiers in Handwriting Recognition,* Concordia University, Montreal, Canada, 1990.

D. W. Krumme, "Theory and Implementation of a Network Representation of Knowledge: Application to Character Recognition," Ph.D. dissertation, University of California, Berkeley, 1979.

D. S. Lee, S. W. Lam, and S. N. Srihari, "A Structural Approach to Recognize Hand-printed and Degraded Machine-printed Characters," *Proceedings of the Workshop on Syntactic and Structural Pattern Recognition,* Murray Hill, N.J., 1990, pp. 256–272.

D. S. Lee, S. N. Srihari, and R. Gaborski, "Bayesian and Neural Network Pattern Recognition: A Theoretical Connection and Empirical Comparison and Handwritten Digits," in I. Sethi and A. Jain, eds., *Artificial Neural Networks & Statistical Pattern Recognition,* in press.

J. L. McClelland and D. E. Rumelhart, "An Interactive Activation Model of Context Effects in Letter Perception: Part 1, An Account of Basic Findings," *Psychol. Rev.* **88**(5), 375–407 (September 1981).

G. Nagy, "Optical Scanning Digitizers," *IEEE Computer,* 13–24 (May 1983).

D. L. Neuhoff, "The Viterbi Algorithm as an Aid in Text Recognition," *IEEE Trans. Inform. Theory* **IT–21,** 222–228 (1975).

T. Pavlidis, "Effects of Distortions on the Recognition Rate of a Structural OCR System," *Proceedings of the IEEE-CS Conference on Computer Vision Pattern Recognition,* 1983, pp. 303–309.

T. Pavlidis, "A Vectorizer and Feature Extractor for Document Recognition," *Computer Vision, Graphics and Image Processing* **35,** 111–127 (1986).

T. F. Pawlicki, D. S. Lee, J. J. Hull, and S. N. Srihari, "Neural Network Models and Their Application to Handwritten Digit Recognition," *IEEE International Conference on Neural Networks,* San Diego, Calif., 1988, pp. 63–70.

J. Schurmann, "A Multifont Word Recognition System for Postal Address Reading," *IEEE Trans. Computers* **C-27**(8), 721–732 (April 1978).

R. J. Shillman, "Character Recognition Based on Phenomenological Attributes: Theory and Methods," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, Mass., 1974.

R. Shinghal and G. T. Toussaint, "Experiments in Text Recognition with the Modified Viterbi Algorithm," *IEEE Trans. Patt. Anal. Mach. Intell.* **PAMI-1**(2), 184–192 (1979a).

R. Shinghal and G. T. Toussaint, "A Bottom-up and Top-down Approach to Using Context in Text Recognition," *Int. J. Man-Mach. Stud.* **11,** 201–212 (1979b).

S. N. Srihari and G. Zack, "Document Image Analysis," *Proceedings of the Sixth International Conference on Pattern Recognition,* Paris, 1986, pp. 434–436.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

S. N. Srihari, J. J. Hull, and R. Choudhari, "Integrating Diverse Knowledge Sources in Text Recognition," *ACM Trans. Office Information Systems* **1**(1), 68–87 (1983).

S. N. Srihari, E. Cohen, J. J. Hull, and L. Kuan, "A System to Locate and Recognize ZIP Codes in Handwritten Addresses," *Int. J. Res. Eng.* **1**(1), 37–45 (1989).

J. D. Tubbs, "A Note on Binary Template Matching," *Patt. Recog.* **22**(4), 359–364 (1989).

A. J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Trans. on Inform. Theory* **IT–13**(2), 260–269 (April 1967).

Surveys and Tutorials

G. Nagy, Optical Character Recognition: Theory and Practice, in P. R. Krishnaiah and L. N. Kanal, eds., *Handbook of Statistics*, Vol. 2, pp. 621–649, 1982, is a survey of statistical feature analysis techniques for character recognition.

E. Reuhkala, "Recognition of Strings of Discrete Symbols with Special Application to Isolated Word Recognition," *Acta Polytech. Scand. Ma.* **38**, 1–92 (1983) contains a brief survey of methods for contextual postprocessing and an exhaustive bibliography.

S. N. Srihari, *Computer Text Recognition and Error Correction*, IEEE Computer Society Press, Silver Spring, Md., 1984, is a tutorial on the reading of text by computer. Twenty basic papers and an extensive bibliography are given.

C. Y. Suen, M. Berthod, and S. Mori, "Automatic Recognition of Handprinted Characters—The State of the Art," *Proc. IEEE* **68**(4), 469–487 (April 1980) is a survey of techniques developed for the recognition of isolated handprinted characters.

I. Taylor and M. M. Taylor, *The Psychology of Reading*, Academic Press, Orlando, FL, 1983, is an overview of research about human reading. Contains a comprehensive bibliography.

J. R. Ullmann, Advances in Character Recognition, in K. S. Fu, ed., *Applications of Pattern Recognition*, CRC Press, Boca Raton, Fla., pp. 197–236, 1982, is a general overview of character recognition techniques oriented toward practical applications. A comprehensive bibliography including many U.S. and U.K. patents is given.

SARGUR N. SRIHARI
JONATHAN J. HULL
SUNY, Buffalo

CHECKERS-PLAYING PROGRAMS

Programming computers to play games is one of the earliest areas of AI research (Jackson, 1974; Barr and Feigenbaum, 1981). As it did in the past, it continues today to attract workers for a number of reasons. The first and most obvious of these is that the ability to play complex games appears to be the province of the human intellect. It is therefore challenging to write programs that match or surpass the skills humans have in planning (qv), reasoning, and choosing among several options in order to reach their goal. Another motivation for this research is that the techniques developed while programming computers to play games may be used to solve other complex problems in real life, for which games serve as models. Finally, games provide researchers in AI in particular and computer science in general with a medium for testing their theories on various topics ranging from knowledge representation (qv) and the process of learning (qv) to searching algorithms (see SEARCH) and parallel processing. The game of checkers was one of the first for which a program was written. This entry describes the early and important work of Samuel (1963, 1967) as well as more recent efforts by Griffith (1974) and Akl and Doran (1983) (see GAME PLAYING).

THE GAME OF CHECKERS

Checkers is an old board game believed to have originated in ancient Egypt (Ryan, 1978). It is played by two persons and involves no element of chance. The presence of clear rules and goals makes it a game of strategy. Also, the game is one of perfect information in the sense that at any given time both players have complete knowledge of all the previous moves and the current board situation. Finally, the outcome of a game is either a win for one of the two players and a loss for the other or a draw. Checkers is therefore a zero-sum game.

Like most other game-playing programs, all known programs for playing checkers search a game tree (qv), an example of which is shown in Figure 1. In such a tree nodes correspond to board positions and branches correspond to moves. The root node represents the board position from which the player whose turn it is to play is required to make a move. A node is at ply (or depth) k if it is at a distance of k branches from the root. A node at ply k, which has branches leaving it and entering nodes at ply $k + 1$, is called a nonterminal node; otherwise the node is terminal. A nonterminal node at ply k is connected by branches to its offspring at ply $k + 1$. Thus, the offspring of the root represent positions reached by moves from the initial board; offspring of these represent positions reached by the opponent's replies; offspring of these represent positions reached by replies to the replies, and so on. The number of branches leaving a nonterminal node is the

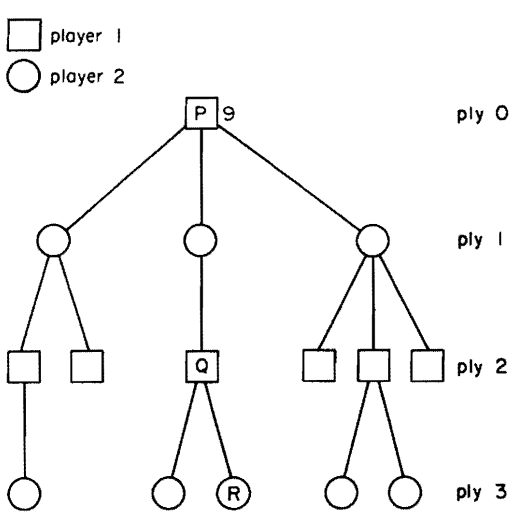


Figure 1. A game tree: P, Q, and R are board positions. Number 9 is the value of the alpha-beta search of position P.

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.

ENCYCLOPEDIA OF ARTIFICIAL INTELLIGENCE SECOND EDITION

VOLUME 1

Stuart C. Shapiro, *Editor-in-chief*

A Wiley-Interscience Publication
John Wiley & Sons, Inc.
New York / Chichester / Brisbane / Toronto / Singapore

Encyclopedia of Artificial Intelligence, Second Edition, Wiley, New York, 1991, 138-150.