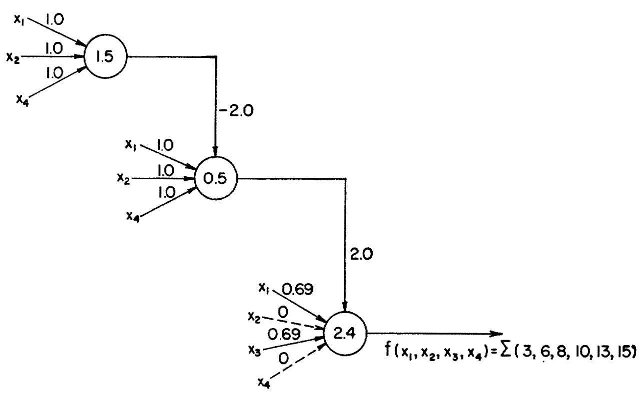

Fig. 3. A threshold-logic realization of the threshold-product function $f(x_1, x_2, x_3, x_4) = \Sigma(3, 6, 8, 10, 13, 15)$.

from (61a), (61b), (61c), and (61f). These are all the restrictions on the $|b_i|$ specified by (61a) through (61h). Hence, if we pick $|b_0| = \frac{2}{3}$, we can have $|b_2| = |b_4| = 1$ from (62) and (63), $|b_1| = 2$ from (64), and $|b_3| = 2$ from (65). Therefore, $(-\frac{2}{3}, -2, -1, 2, -1)$ is a constrained solution for (57). Using the realization method described in Section IV, we need three threshold-logic elements for realizing (57). The realization based on the above constrained solution is shown in Fig. 3.

VI. Conclusions

In this note, a class of switching functions, called threshold-product functions, has been studied in detail. We have shown that both threshold-sum functions (threshold functions) and parity functions are special cases of threshold-product functions. A simple threshold-logic realization technique requiring $2 + \lfloor \log_2 p \rfloor$ threshold-logic elements has been found for a threshold-product function with a solution of index p. In order to reduce the number of threshold-logic elements in the realization, a constrained solution is desired. A systematic method for finding a constrained solution, when a switching function is a threshold-product function, has been established. This method can be employed for testing whether a switching function is a threshold-product function as well. When the number of variables in a switching function is not large, say no more than 6, a simpler method for the above purposes has been found. Furthermore, when a threshold-product function has a constrained solution of index 2, a minimal threshold-logic realization method has been obtained. Further study along this line is to find a minimal threshold-logic realization technique for any threshold-product function. If we could find an efficient way to decompose any switching function into a combination of a minimal number of threshold-product functions, then an economical threshold-logic realization method for an arbitrary switching function would be obtained.

References

[1] M. C. Paull and E. J. McCluskey, "Boolean functions realizable with single threshold devices," *Proc. IRE*, vol. 48, pp. 1335–1337, July 1960.

[2] S. Muroga, I. Toda, and S. Takasu, "Theory of majority decision elements," *J. Franklin Inst.*, vol. 271, pp. 376–418, May 1961.

[3] C. K. Chow, "On the characterization of threshold functions," *Proc. AIEE 2nd Ann. Symp. on Switching Circuit Theory and Logical Design*, pp. 34–38, September 1961.

[4] C. C. Elgot, "Truth functions realizable by single threshold organs," *Proc. AIEE 2nd Ann. Symp. on Switching Circuit Theory and Logical Design*, pp. 225–245, September 1961.

[5] S. Muroga, "Functional forms of majority functions and a necessary and sufficient condition for their realizability," *Proc. AIEE 2nd Ann. Symp. on Switching Circuit Theory and Logical Design*, pp. 39–46, September 1961.

[6] R. O. Winder, "Single stage threshold logic," *Proc. AIEE 2nd Ann. Symp. on Switching Circuit Theory and Logical Design*, pp. 321–332, September 1961.

[7] I. J. Gabelman, "The synthesis of Boolean functions using a single threshold element," *IRE Trans. Electronic Computers*, vol. EC-11, pp. 639–642, October 1962.

[8] C. L. Sheng, "A method for testing and realization of threshold functions," *IEEE Trans. Electronic Computers*, vol. EC-13, pp. 232–239, June 1964.

[9] M. L. Dertouzos, "An approach to single-threshold-element synthesis," *IEEE Trans. Electronic Computers*, vol. EC-13, pp. 519–528, October 1964.

[10] I. J. Gabelman, "Properties and transformations of single threshold element functions," *IEEE Trans. Electronic Computers*, vol. EC-13, pp. 680–684, December 1964.

[11] Y. C. Ho and R. L. Kashyap, "An algorithm for linear inequalities and its applications," *IEEE Trans. Electronic Computers*, vol. EC-14, pp. 683–688, October 1965.

[12] H. C. Torng, "An approach for the realization of linearly-separable switching unctions," *IEEE Trans. Electronic Computers*, vol. EC-15, pp. 14–20, February 1966.

[13] R. C. Minnick, "Linear input logic," *IRE Trans. Electronic Computers*, vol. EC-10, pp. 6–16, March 1961.

[14] R. O. Winder, "More about threshold logic," *Proc. AIEE 2nd Ann. Symp. on Switching Circuit Theory and Logical Design*, pp. 55–64, September 1961.

[15] P. M. Lewis and C. L. Coates, "Realization of logical functions by a network of threshold components with specified sensitivity," *IEEE Trans. Electronic Computers*, vol. EC-12, pp. 443–461, October 1963.

[16] S. H. Cameron, "The generation of minimal threshold nets by an integer program," *IEEE Trans. Electronic Computers (Correspondence)*, vol. EC-13, pp. 299–302, June 1964.

[17] J. E. Hopcroft and R. L. Mattson, "Synthesis of minimal threshold logic networks," *IEEE Trans. Electronic Computers*, vol. EC-14, pp. 552–560, August 1965.

[18] E. P. Stabler, "Threshold gate network synthesis," *1965 IEEE Conf. Rec., 6th Ann. Symp. on Switching Circuit Theory and Logical Design* (Ann Arbor, Mich.: October 6–8, 1965), pp. 5–11.

[19] M. A. Breuer, "Implementation of threshold nets by integer linear programming," *IEEE Trans. Electronic Computers (Short Notes)*, vol. EC-14, pp. 950–952, December 1965.

[20] C. L. Sheng, "Compound synthesis of threshold-logic networks for the realization of general Boolean functions," *IEEE Trans. Electronic Computers*, vol. EC-14, pp. 798–814, December 1965.

[21] W. H. Kautz, "The realization of symmetric switching functions with linear-input logical elements," *IRE Trans. Electronic Computers*, vol. EC-10, pp. 371–378, September 1961.

[22] W. W. Peterson, *Error-Correcting Codes.* Cambridge, Mass.: M.I.T. Press, 1961, Chapter 3.

[23] S. H. Caldwell, *Switching Circuits and Logical Design.* New York: Wiley, 1960, ch. 5.

[24] W. O. Winder, "Enumeration of seven-argument threshold functions," *IEEE Trans. Electronic Computers*, vol. EC-14, pp. 315–325, June 1965.

Experiments with Highleyman's Data

JOHN H. MUNSON, RICHARD O. DUDA, MEMBER, IEEE, AND PETER E. HART, MEMBER, IEEE

***Abstract*—The results of three experiments with Highleyman's hand-printed characters are reported. Nearest-neighbor classification is used to explain the high error rates (42 to 60 percent) obtained by general statistical procedures. An error rate of 32 percent is obtained by preceding piecewise-linear classification by edge-detecting preprocessing. The minimum human error rate is estimated, and suggested as a performance standard.**

***Index Terms*—Character recognition, classification, feature extraction, human performance, nearest-neighbor classification, pattern recognition, preprocessing.**

Introduction

The problem of recognizing hand-printed characters has attracted the attention of researchers for more than a decade. Among the many experiments that have been reported, the only ones that can be directly compared are those that used a set of data collected, quantized, and encoded by Highleyman.[1]–[6]

In 1963, in response to several requests for the use of his data, Highleyman offered to make the set available as an ". . . unintended, incomplete, yet interesting, available, and temporary standard."[7] The data consisted of 50 alphabets of hand-printed characters. Each alphabet consisted of the 10 numerals and 26 upper-case letters printed by a particular individual, and each character was quantized and represented as a 12×12 binary (black-white) array.

The great amount of variability encountered in the data has tended to rule out the simpler approaches, such as the use of decision trees, and the methods used have been more or less statistical in spirit. One common characteristic of these methods has been the use of some or all of the patterns to fix the values of free parameters in the classifier. In those cases where the first 40 alphabets (called the training data) were used to determine parameters and the last 10 alphabets (called the testing data) were used to provide an independent test, the performance on the test data was always much worse than the performance on the training data. For example, Chow[4] obtained a 2.1 percent error rate on the training data, but a 41.7

Manuscript received October 20, 1967. The research reported here was supported by the U. S. Army Electronics Command under Contract DA 28-043-AMC-01901(E).

The authors are with the Stanford Research Institute, Menlo Park, Calif.

percent error rate on independent test, and this represents the best performance reported to date.

Similar discrepancies have been noted by other investigators,[2],[4]–[6] and have usually been attributed to the small number of samples available for characters having so much variability. There is no doubt that a larger number of samples would reduce the size of this discrepancy, for in the case of infinite training and testing sets, the error rates should be the same. It is not clear, however, how much the test error rate would be reduced, or how many samples would be needed to estimate the best achievable performance.

The purpose of this note is to describe the results of three different experiments with Highleyman's data. The first used a nonparametric classification procedure that exchanges the need for assumptions about the pattern distributions for the need for a large number of patterns. The second used edge-detecting preprocessing prior to classification to remove some of the variability in the characters and to exploit simple a priori knowledge about the data. In the third experiment, the ability of people to recognize the test data was measured to provide an objective performance standard.

Nearest-Neighbor Classification

The nearest-neighbor decision rule (NN rule) is a nonparametric decision rule that assigns an unclassified pattern to the class of the nearest pattern of a set of correctly classified reference patterns.[8] When the set of reference patterns is large, the error rate of the NN rule is less than twice the minimum possible error rate. Specifically, if

P_0 = Bayes probability of error,
P = Large-sample NN probability of error,
N = Number of classes,

then, under very weak regularity conditions,

$$P_0 \leq P \leq 2P_0 - \frac{N}{N-1} P_0^2,$$

and these bounds can be shown to be the tightest possibe.[8],[9]

When the NN rule was applied to Highleyman's data, the training patterns were used as the reference patterns for the classification of the testing data. Each pattern was viewed as a 144-component binary vector. A test pattern was classified by measuring the Hamming distance between it and each of the 1440 training patterns, and by assigning it to the class of the nearest pattern; ties with patterns in different classes were broken arbitrarily.

The error rate resulting from applying this procedure to the testing data was 47.5 percent. If the training set were large enough for the large sample results to hold, this would mean that the minimum error rate would lie somewhere between 27.6 and 47.5 percent. We shall see that the minimum error rate is probably less than 11.4 percent, and, hence, that the training data is not a sufficiently large sample in the nearest-neighbor sense.

Preprocessing and Piecewise-Linear Classification

The purpose of preprocessing is to simplify the classification problem by extracting from the input data only that information which is needed for classification. In designing a preprocessor, the designer in effect tries to give the classifier the benefit of his knowledge of the problem. His goal is to find features which discriminate between characters in different classes, and are relatively insensitive to normal variations among characters in the same class.

We have investigated several preprocessing techniques in the course of a project to recognize hand-printed text.[10] Among these techniques are the use of feature templates, which look for the presence of such features as edges, corners, line segments, etc., and the extraction of descriptions of topological and geometrical characteristics such as enclosures, concavities, stroke tips, etc. While these latter are among the most useful features, it is quite difficult to extract them from characters that are as frequently broken and fragmented as are Highleyman's characters. Thus, the only preprocessing that we attempted was the use of simple edge-detecting templates.

The preprocessing/classification method we used has been described in detail by Munson.[11],[12] The only operation we had to perform to use existing computer programs with Highleyman's data was to expand the 12×12 figures to 24×24 figures to match our standard format. This was done merely by copying each row and column twice. Edge detection was accomplished by the use of edge-detecting mask pairs, or templates. Each mask pair consisted of two 2×8 rectangles of points, adjacent to each other along their long edges. One of the masks was given positive weight, the other, negative, and a threshold was set such that if the positive mask encountered six more figure points than the negative one, the binary response of the mask pair was ON.

To provide a limited degree of translation invariance, the responses of five such mask pairs were ORed together to give a single binary component of the output feature vector. The five mask pairs in a group had the same orientation and were in the same region of the 24×24 field. Nine regions were allotted to each of the four major compass directions, and six regions were allotted to each of the eight secondary directions. Thus, the complete feature vector consisted of 84 binary components, and the significance of a typical component was "An edge oriented north-of-west has been detected in the left central region of the field."

These 84-bit feature vectors, augmented by an 85th threshold bit, formed the input to the classifier, which was a piecewise-linear machine.[6] This classifier formed 72 dot products of the feature vector with 72 stored weight vectors, 2 for each of the 36 classes. It classified a pattern by assigning it to the class corresponding to the largest dot product. The weight values were determined by fixed-increment error-correction training.[6] The training margin was set to 85 so that a correction was made whenever the dot product for the correct class failed to exceed all dot products in other classes by this amount; no margin was used during testing.

During training, the training patterns were viewed in any of nine different positions, a nominal position in which the character was centered, and eight other positions obtained by displacing the figure by two elements vertically and/or horizontally. After 18 training iterations (by which time all views of all of the training patterns had been encountered twice), testing was performed. All nine views of each test pattern were presented, and the class appearing most often among the nine individual responses was selected for the pattern. The resulting error rate for all 36 classes was 31.7 percent. Repetition of this experiment using the ten numerals alone yielded an error rate of 12.0 percent. Both of these results are significantly better than previously reported results, but this performance still falls short of human performance.

Human Performance

In 1960, Neisser and Weene reported an average error rate of 4.1 percent made by a group of nine people in recognizing hand-printed upper-case letters and numerals, and indicated that 3.2 percent was probably a good estimate of the minimum possible error rate for their data.[13] These results apply to a 34-category alphabet, since confusions between I and 1 or between 0 and Ø were not counted as errors. More importantly, the characters used were reproduced photographically with high resolution and apparently with good gray scale, whereas Highleyman's data are low-resolution figures with two-level gray scale; thus, these numbers do not apply to Highleyman's data.

To estimate human error rates on Highleyman's data, we performed a simple, computer-controlled experiment involving ten people, who, though aware of the existence of Highleyman's data, had not seen the test data before. The experimental procedure had two phases, a training phase in which the subjects familiarized themselves with both the equipment and the data by viewing the training data under test conditions, and a testing phase in which performance was recorded. In both phases, the characters were selected randomly without replacement from ten alphabets printed by ten different writers; the training phase used the first ten alphabets, while the testing phase used the last ten.

The characters were displayed as a 12×12 array of points (bright points for the figure) occupying a 0.3-inch square centered in a 3×4.5-inch oscilloscope screen. Each subject was free to take as long as he wished in making up his mind, and when a decision was reached he recorded it by striking the corresponding typewriter key. This caused the subject's decision to be recorded, the correct character to be typed out if a mistake had been made, and the next character to be displayed. We chose to maintain the error response during the testing phase due to its very noticeable ability to sustain the subject's attention and to induce him to perform well.

Most subjects were satisfied with the training phase after they had seen 75 to 100 characters, and volunteered to move on to the testing phase. On the test data, their error rates ranged from 13.6 percent to 18.3 percent, with an average error rate of 15.7 percent. Assuming a normal distribution of scores, this indicates that, with 95 percent confidence, the true mean error rate is 15.7 percent $\pm$ .9 percent.

These numbers include a fair proportion of errors due to confusions between I and 1 and 0 and Ø. If these errors are not counted, the mean error rate drops to 11.5 percent, which is still considerably greater than the 4.1 percent reported by Neisser and Weene for their unquantized characters. If the I-1 and 0-Ø distinctions are retained, but if a plurality vote of the ten separate responses is used to classify the characters (ties being broken arbitrarily), then an error rate of 11.4 percent results. We believe that this value is close to the minimum error rate that can be achieved with Highleyman's data, and that the performance of other methods on the 36-character test data should be viewed relative to this standard.

Discussion

If 11.4 percent is the minimum achievable error rate for Highleyman's data, then the 47.5 percent error rate obtained by nearest-neighbor classification indicates that the amount of training data is much too small for such a general nonparametric technique. We suspect that any statistical technique that makes use of little a priori knowledge of the distributions will experience this same difficulty, and that this explains, at least in part, the discrepancies reported elsewhere between training and testing performance.[2],[4]–[6]

If it is not practical to obtain enough training data to allow the use of such general techniques, then some preprocessing must be done to exploit the investigator's a priori knowledge of the problem. By using edge-detecting preprocessing followed by nine-view classification by a piecewise-linear machine, we obtained an error rate of 31.7 percent. While this represents a significant improvement over previously reported results, it is still far too high to be practical.

While the development of more effective preprocessing and classification techniques for Highleyman's data may be a challenging problem in itself, we feel that larger and higher-quality data sets are needed for work aimed at achieving useful results. Such data sets may contain hundreds, or even thousands, of samples in each class. We know, for example, that investigators at SRI and IBM have used data sets containing over ten thousand samples, and we expect that even larger data sets will be collected.

Experience with such data suggests that an array size of at least 20×20 is needed, with an optimum size of perhaps 30×30. Multilevel gray scale or adaptive two-level quantization may be valuable. In any case, the data (whether in its original or quantized form) should be recognizable to humans with no more than a few percent errors. No machine recognizer should be expected to exceed human performance on the original characters viewed out of context, and progress beyond this point will have to depend on the effective exploitation of contextual relations.

References

[1] W. H. Highleyman and L. A. Kamentsky, "Comments on a character recognition method of Bledsoe and Browning," *IRE Trans. Electronic Computers (Correspondence)*, vol. EC-9, p. 263, June 1960.

[2] W. W. Bledsoe, "Further results on the N-tuple pattern recognition method," *IRE Trans. Electronic Computers (Correspondence)*, vol. EC-10, p. 96, March 1961.

[3] W. H. Highleyman, "An analog method for character recognition," *IRE Trans. Electronic Computers*, vol. EC-10, pp. 502–512, September 1961.

[4] C. K. Chow, "A recognition method using neighbor dependence," *IRE Trans. Electronic Computers*, vol. EC-11, pp. 683–690, October 1962.

[5] W. H. Highleyman, "Linear decision functions, with application to pattern recognition," *Proc. IRE*, vol. 50, pp. 1501–1514, June 1962.

[6] R. O. Duda and H. Fossum, "Pattern classification by iteratively determined linear and piecewise linear discriminant functions," *IEEE Trans. Electronic Computers*, vol. EC-15, pp. 220–232, April 1966.

[7] W. H. Highleyman, "Data for character recognition studies," *IEEE Trans. Electronic Computers (Correspondence)*, vol. EC-12, pp. 135–136, April 1963.

[8] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Information Theory*, vol. IT-13, pp. 21–27, January 1967.

[9] P. E. Hart, "An asymptotic analysis of the nearest-neighbor decision rule," Stanford Electronics Labs., Stanford, Calif., Tech. Rept. 1828-2 (SEL-66-016), May 1966.

[10] J. H. Munson, "The recognition of hand-printed text," *Proc. IEEE Pattern Recognition Workshop* (to be published).

[11] A. E. Brain and J. H. Munson, "Graphical-data-processing research study and experimental investigation," Stanford Research Institute, Menlo Park, Calif., Tech. Rept. 22, Contract DA 36-039 AMC-03247(E), SRI Project 4565, April 1966.

[12] R. O. Duda and J. H. Munson, "Graphical-data-processing research study and experimental investigation," Stanford Research Institute, Menlo Park, Calif., Tech. Rept. ECOM-01901-24, Contract DA 28-043 AMC-01901(E), SRI Project 5864, August 1966.

[13] U. Neisser and P. Weene, "A note on human recognition of hand-printed characters," *Information and Control*, vol. 3, pp. 191–196, 1960.