q_i to state q_j with exactly m steps (*i.e.*, with m inputs) and 0 otherwise. Also, the $i-j$ entry of B_m is 1 if it is possible to proceed from state q_i to state q_j with at most m inputs, and 0 otherwise.

We wish to show that for some m every entry in the first row of B_m is a 1, *i.e.*, $b_{0j}^{(m)}=1$; $0 \leq j \leq \alpha - 1$. Thus, with a suitable input sequence of at most m bits, you can go from state q_0 (the initial state) to every other state.

Upon computing the matrix A_m, we find that for $0 \leq k \leq m\alpha - 1$

$$a_{ij}^{(m)} = 1 \text{ when } \begin{cases} i = k \pmod{\alpha} \\ j = \left[\dfrac{k}{2^m}\right] \end{cases}$$

and

$$a_{ij}^{(m)} = 0 \text{ otherwise.}$$

fig. 5 shows A_2 and A_3 for $\alpha = 6$.

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

(a) (b)

Fig. 5—Transition matrices for $\alpha = 6$. (a) A_2. (b) A_3.

Observe that the number of 1's in each column of A_m is 2^m for $2^m \leq \alpha$, and that for $2^m \geq \alpha$ the matrices A_m and B_m have all their entries equal to 1. Hence the number of input bits required to go from any state to any other state is no larger than the least integer greater than or equal to $\log_2 \alpha$. Thus, starting in state q_0, each of the α states can be reached. This shows that α states are both necessary and sufficient for a Turing machine of our form to perform multiplication by α.

JOEL M. WINETT
M.I.T. Lincoln Lab.
Lexington, Mass.

Improved Memory Matrices for the *n*-Tuple Pattern Recognition Method*

Summary

It is shown that a previous version of the *n*-tuple pattern recognition method can be made more effective by making certain changes in the learning phase. A means of further increasing readability through judicious choice of *n*-tuples is described.

Introduction

Two phases, "*learning*," and "*reading*," are used in the *n*-tuple pattern recognition method.[1] The *memory matrix* stores information obtained from sample patterns during the learning phase, and this stored information is used in the reading phase to identify patterns not previously presented. Obviously, the quality of information stored and the methods' success are directly related, although other factors must also be taken into account.

A storage address for each possible state $(0, 1, 2, \cdots, 2^n - 1)$ of each *n*-tuple for each character type (*e.g.*, *A*, *B*, *C*, $\cdots$) is provided in the memory matrix. Only 0's and 1's were stored in these addresses in the original version.[2] A 1 was stored in a given address if the *n*-tuple in question was placed in the specified state by *any one* of the given character images presented during learning. Even in this original paper it was suggested that the memory matrix might be used more effectively if frequencies, rather than merely 0's and 1's were stored.[3] This suggestion, however, was not exploited at that time. This paper seeks to demonstrate that readability with the *n*-tuple method is much improved if memory matrices are chosen which are more nearly optimum than are 0, 1 matrices. Variations resulting from value changes of the parameter n are also given.

The Frequency Memory Matrix

A different learning mode for the memory matrix is given in each row of the first column of Table I. The original *n*-tuple procedure with its 0, 1 matrix is represented in row 1.

Row 2 represents a memory matrix obtained by *adding* 1's to the appropriate addresses during learning, and then dividing final sums by the number of sets learned (40 in all cases reported here). This *frequency memory matrix* was used as a first step in obtaining the matrices represented in rows 3, 4, 5 and 6 of Table I.

Three Normalizations of Frequency Memory Matrix

Three normalizations of the frequency memory matrix are given in rows 3, 4 and 5 of Table I. For row 3, the matrix was normalized "within state," that is, each entry was divided by the sum of all entries referring to the same state of the same *n*-tuple.

Highleyman's and Kamentsky's normalization procedure[5] was used to obtain the results shown in row 4, Table I. They used the method quite successfully, on this same data, for the case of $n=1$.[6] As reported in row 4, the matrix entries were normalized "within character class." Each entry was divided by the square root of the sum of the squares of entries referring to the particular character class. The matrix is thus reduced to a set of unit vectors.

A variation of the Highleyman-Kamentsky normalization technique, in which the zero states of the *n*-tuples are suppressed, is represented in row 5 of Table I. This corresponds, in the case of $n=1$, to handling "black" image areas only.

TABLE I
PER CENT RECOGNIZED CORRECTLY
COMPARISON OF METHODS

Row	Method	40 Sets Read (Same as learned) $n=1$	2	6	10 Sets Read (Different than learned) $n=1$	2	6
		%	%	%	%	%	%
1	0, 1 Matrix		13.0	62.0		2	24
2	Probability Matrix	45.0	48.5		25	25	
3	Normalized Within State	70.0	70.0		47	47	
4	Highleyman-Kamentsky Method	77.5	78.5	81.5	58	58	53
5	Highleyman-Kamentsky Method, Zero State Suppressed	76.7	77.2	80.7	59	62	51
6	Maximum Likelihood Method	79.7	83.2	91.7	61	63	61

Source of Data

The images used to obtain data given in Table I consisted of 50 sets of handwritten numbers, 0 through 9, for a total of 500 separate images. These numerals were written by 50 different persons at Bell Telephone Laboratories.[4] Subsequently, the images were individually digitized on a 12×12 grid. In all trials reported here, 40 sets were learned; following which, all 50 sets were read. Before learning or reading, each image was centered by placing its center of gravity at a predetermined point on the "retina."

Percentage of characters successfully recognized under a variety of conditions is summarized in Table I. The various methods used were listed in the first column. The second column shows percentage successfully read via each method, using the 40 alphabets already learned. The third column shows percentage successfully read of the 10 alphabets not previously learned. (Ties in all cases are counted as failures.) In each case, results are shown for $n=1$, $n=2$, and $n=6$.

The Maximum Likelihood Technique

A procedure[7] based on a maximum likelihood technique was employed to obtain the results given in row 6 of Table I. This procedure, for $n \geq 2$, is a generalization of the technique described in detail by Minsky in his "Steps Toward Artificial Intelligence."[8] Here each entry in the frequency memory matrix is replaced by its logarithm,

* Received July 2, 1961.

[1] W. W. Bledsoe and I. Browning, "Pattern recognition and reading by machine," *Proc. Eastern Joint Computer Conf.*, Boston, Mass., pp. 225–232; 1959.

[2] Bledsoe and Browning, *ibid.*, pp. 225–227.

[3] Bledsoe and Browning, *op. cit.*, see "Probability," p. 231.

[4] Provided to the authors by W. H. Highleyman and referred to by Highleyman and L. A. Kamentsky in "Comments on a character recognition method of Bledsoe and Browning," IRE TRANS. ON ELECTRONIC COMPUTERS (*Correspondence*), vol. EC-9, p. 263; June, 1960.

[5] W. H. Highleyman, "An analog method for character recognition," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-10, pp. 502–512; September, 1961. (See also Highleyman and Kamentsky.[4])

[6] Other centering techniques which further increase readability were also introduced by Highleyman and Kamentsky.

[7] W. Harkness, D. T. Laird, and L. L. Pryor, Pennsylvania State University (private communication).

[8] M. L. Minsky, "Steps toward artificial intelligence," PROC. IRE, vol. 49, pp. 14–15; January, 1961.

with zero replaced by some pre-selected negative number. As Table I shows, "per cent read" improved dramatically when this approach is used, increasing from a rate of 13 per cent to a rate of 83 per cent in the case of $n=2$. (Ties are counted as incorrect.) The maximum likelihood learning procedure is the most efficient yet reported for the n-tuple method.

In regard to all the above methods, in so far as they utilize conditional probabilities, decision functions, and correlation techniques, reference should be made to the work of Chow.[9]

Some Observations

Note that $n=2$ definitely provides better readability than $n=1$, although not by a large amount. It can also be seen that $n=6$ is more impressive for the 40 learned image sets than for the 10 unlearned sets. Readability for $n=6$ on unlearned images probably would improve if learning experience was substantially increased, say to 1000 sets. Generally speaking readability can be expected to increase with n but, at the same time, more learning experience will be required.

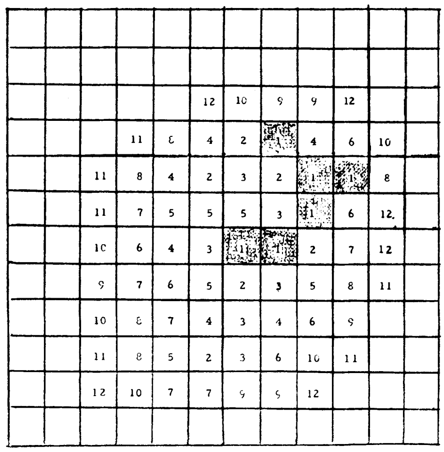

Fig. 1—The 12×12 retina, showing the elements of each of the twelve specially chosen 6-tuples which were used for the results given in Table II. For example, the elements of the first 6-tuple are shown as dark squares.

TABLE II
Comparison of Per Cent Read Correctly for Random n-Tuple and the Specially Chosen 6-Tuples of Fig. 1

Row	Method	40 Sets Read (Same As Learned) $n=1$	2	6	10 Sets Read (Different Than Learned) $n=1$	2	6
		%	%	%	%	%	%
1	0, 1 Matrix Method Using Random n-Tuples (From Table I)		13.0	62.0		2	24
2	0, 1 Matrix Method Using the 12 6-Tuples Shown in Fig. 1			98.0			50
3	Maximum Likelihood Method Using Random n-Tuples (From Table I)	79.7	83.2	91.7	61	63	61
4	Maximum Likelihood Method Using the 12 6-Tuples Shown in Fig. 1			99.75			67

Even though the maximum likelihood method scored better than the other methods tried in this small study, it would be a mistake to claim that such a result should have been expected beforehand. In fact the method is based on the assumption that the n-tuples are *independent*. But for most problems in character recognition they are very *dependent* (especially in the case $n=1$).[10]

Results like those reported in Table I inevitably raise an important question: How much more readability can be expected with further improvements in the matrix? Clearly, an *optimum* set of stored matrix values exists for any given pattern set. Indeed, learning can be described precisely as the attempt to obtain the optimum matrix for specified sets, some methods being superior to others in any given case.

If "number of images correctly read" is accepted as the definition of "readability," readability for a given pattern set can be regarded as a function of many variables, the variables being values recorded in the matrix. In this light, optimization techniques can be employed to seek the optimum matrix for a given set. The result, when found, could then be recorded as the last or "best" row entry in Table I. The actual matrix which provides optimum performance will depend naturally on the particular n-tuples selected, as well as on other system parameters, including n itself. Optimization, therefore, must at least include: finding the "best" n, finding the "best" set of n-tuples, and finding the "best" corresponding memory matrix.

An Attempt to Optimize n-Tuples

In the calculations represented by Table I, no effort was made to discover optimum n-tuples. In fact, the n-tuples used were deliberately selected on a random basis. Subsequently, one attempt was made to find a more nearly optimum n-tuple set, as follows:

The probability of a given element being touched by any image presented was established for each retinal element. These elements were then rank-ordered, from most to least probable and placed in groups of six. The first twelve 6-tuples chosen by this method are shown in Fig. 1.

This rather small but apparently powerful set was used in conjunction with the maximum likelihood learning procedure. The result of this attempt is shown in Table II, where it is compared with the corresponding result for the randomly chosen n-tuples. Even better results would be predicted if a larger and stil more nearly optimum n-tuple set were to be applied. For example, the methods of information theory might be applied to choose more efficient n-tuples, with each prospective n-tuple treated as a separate information channel.[11]

Discovery of the "best" n-tuple sets and the "best" weights for them (matrix entries) seems closely related to the "demon" problem of Selfridge,[12] to the operator problem of Uhr and Vossler,[13] and to the work of Doyle[14] and others.

Other References

[1] W. W., Bledsoe, "Further results on the n-tuple pattern recognition method," IRE Trans. on Electronic Computers (*Correspondence*), vol. EC-10, p. 96; March, 1961.
[2] L. Uhr, "A possibly misleading conclusion as to the inferiority of one method for pattern recognition to a second method to which it is guaranteed to be superior," IRE Trans. on Electronic Computers (*Correspondence*), vol. EC-10, p. 96; March, 1961.

W. W. Bledsoe
C. L. Bisson
Panoramic Research, Inc.
Palo Alto, Calif.

[9] C. K. Chow, "An optimum character recognition system using decision functions," IRE Trans. on Electronic Computers, vol. EC-6, pp. 247–254; December, 1957.
[10] See, Minsky, *op. cit.*, p. 15, last paragraph.
[11] G. P. Steck, "Statistical model for the Browning-Bledsoe pattern recognition scheme," IRE Trans. on Electronic Computers, vol. EC-11, pp. 274–282; April, 1962.
[12] O. G. Selfridge, "Pandemonium: A paradigm for learning," *in* "Mechanization of Thought Process," HMSO, London, pp. 511–535; 1959.
[13] L. Uhr and C. Vossler, "A pattern recognition program that generates, evaluates, and adjusts its own operators," *Proc. Western Joint Computer Conf.*, Los Angeles, Calif., pp. 550–570; 1961.
[14] W. Doyle, "Recognition of sloppy, handwritten characters," *Proc. Western Computer Conf.*, San Francisco, Calif., pp. 133–142; 1960.

Teaching Aid for "Games That Teach the Fundamentals of Computer Operation"*

In the above mentioned article[1] Englebart outlined a method which gives an audience an insight into the mysteries of a digital computer. This method is unique in that a part of the audience is divided into groups where each person acts as a binary element and, with the proper instructions, the operation of a "human-element" computer is demonstrated. In the demonstration the humans are "wired" into a network similar to an actual computer network. In preliminary testing with student groups, the method was very successful. We have added a simple computer to extend the human-simulation "games" and to give the students a transition to the understanding of the physical realization of these basic operations. The simple computer uses relays for all logic operations, where each relay represents one person.

During the demonstration we first proceed to a point where the human element computer is working satisfactorily in a

* Received February 27, 1962.
[1] D. C. Engelbart, IRE Trans. on Electronic Computers, vol. EC-10, pp. 31–41; March, 1961.