We have refrained from value judgments such as "good" or "better than" in discussing both the classifiers and the evaluators. The actual characteristics which make a classifier suitable for a particular application may depend on the application itself, as, for example, when the penalty associated with misclassification of a class member is not the same as for misclassification of a nonmember.[24] Although the percent correct prediction should probably be abandoned as a measure of the performance of binary classifiers, the other measures discussed can be useful in developing a total picture of relative and absolute classifier performance.

Acknowledgment. Support of this research under Grant MPS-74-01249 from the National Science Foundation is gratefully acknowledged. We also gratefully acknowledge the University of Nebraska Research Council, which provided partial support for purchase of the computer-readable mass spectra data set.

References and Notes

(1) L. J. Soltzberg is a visiting Associate Professor during the 1975–1976 academic year from Simmons College, Boston, Mass.
(2) B. R. Kowalski and C. F. Bender, *J. Am. Chem. Soc.*, **94**, 5632 (1972).
(3) B. R. Kowalski and C. F. Bender, *J. Am. Chem. Soc.*, **95**, 686 (1973).
(4) P. C. Jurs, B. R. Kowalski, T. L. Isenhour, and C. N. Reilley, *Anal. Chem.*, **41**, 690 (1969).
(5) B. R. Kowalski, P. C. Jurs, T. L. Isenhour, and C. N. Reilley, *Anal. Chem.*, **41**, 695 (1969).
(6) L. Pietrantonio and P. C. Jurs, *Pattern Recognition*, **4**, 391 1972).
(7) B. R. Kowalski and C. F. Bender, *Anal. Chem.*, **45**, 2334 (1973).
(8) G. L. Ritter, S. R. Lowry, C. L. Wilkins, and T. L. Isenhour, *Anal. Chem.*, **47**, 1951 (1975).
(9) T. J. Stonham, I. Aleksander, M. Camp, W. T. Pike, and M. A. Shaw, *Anal. Chem.*, **47**, 1817 (1975).
(10) T. J. Stonham and M. A. Shaw, *Pattern Recognition*, **7**, 235 (1975).
(11) B. R. Kowalski, P. C. Jurs, T. L. Isenhour, and C. N. Reilley, *Anal. Chem.*, **41**, 1949 (1969).
(12) S. R. Lowry, H. B. Woodruff, G. L. Ritter, and T. L. Isenhour, *Anal. Chem.*, **47**, 1126 (1975).
(13) H. B. Woodruff, S. R. Lowry, and T. L. Isenhour, *Appl. Spectrosc.*, **29**, 226 (1975).
(14) R. W. Liddell and P. C. Jurs, *Appl. Spectrosc.*, **27**, 371 (1973).
(15) D. R. Preuss and P. C. Jurs, *Anal. Chem.*, **46**, 520 (1974).
(16) R. W. Liddell and P. C. Jurs, *Anal. Chem.*, **46**, 2126 (1974).
(17) B. R. Kowalski and C. A. Reilly, *J. Phys. Chem.*, **75**, 1402 (1971).
(18) B. R. Kowalski and C. F. Bender, *Anal. Chem.*, **44**, 1405 (1972).
(19) C. L. Wilkins, R. C. Williams, T. R. Brunner, and P. C. McCombie, *J. Am. Chem. Soc.*, **96**, 4182 (1974).
(20) T. R. Brunner, R. C. Williams, C. L. Wilkins, and P. J. McCombie, *Anal. Chem.*, **46**, 1798 (1974).
(21) T. R. Brunner, C. L. Wilkins, R. C. Williams, and P. J. McCombie, *Anal. Chem.*, **47**, 662 (1975).
(22) C. L. Wilkins and T. L. Isenhour, *Anal. Chem.*, **47**, 1849 (1975).
(23) L. Uhr, "Pattern Recognition, Learning, and Thought", Prentice-Hall, Englewood Cliffs, N.J., 1973, p 26.
(24) H. Rotter and K. Varmuza, *Org. Mass Spectrom.*, **10**, 874 (1975).
(25) E. Stenhagen, S. Abrahamssen, and F. W. McLafferty, "The Registry of Mass Spectral Data", Wiley-Interscience, New York, N.Y., 1974.
(26) C. H. Chen, "Statistical Pattern Recognition", Hayden Book Co, Inc., Rochelle Park, N.J., 1973, p 54.
(27) The information gain I(A,B) could, of course, be expressed directly in terms of N, N^{pred}, N^{corr}, and N_{total}, but the equation is cumbersome. In computing I(A,B) from $p(i)$, $p(k)$, and $p(i,k)$, the situation can arise where $p(i,k) = 0$. The corresponding term $p(i,k) \log_2 p(i,k)/p(i)p(k)$ in the expression for I(A,B) is then indeterminate. However, by L Hospital's Rule:

$$\lim_{x \to 0} x \log cx = \lim_{x \to 0} \frac{c' \ln cx}{1/x} = \lim_{x \to 0} \frac{\frac{d}{dx} c' \ln cx}{\frac{d}{dx} \frac{1}{x}} = \lim_{x \to 0} c''x = 0$$

Thus, such terms contribute nothing to the summation.
(28) T. F. Lam, C. L. Wilkins, T. R. Brunner, L. J. Soltzberg, and S. L. Kaberline, *Anal. Chem.*, in press.
(29) L. J. Soltzberg, C. L. Wilkins, S. L. Kaberline, T. F. Lam, and T. R. Brunner, *J. Am. Chem. Soc.*, following paper in this issue.
(30) N. J. Nilsson, "Learning Machines", McGraw-Hill, New York, N.Y., 1965.
(31) W. L. Felty and P. C. Jurs, *Anal. Chem.*, **45**, 885 (1973).
(32) DLN classifier 11 (Table V) is exceptional because of its perfect performance on nonmembers of the class and thus has a high figure of merit in spite of its low performance $p(j|1)$ on class members.

Evaluation and Comparison of Pattern Classifiers for Chemical Applications: Adaptive Digital Learning Networks and Linear Discriminants

Leonard J. Soltzberg,[1] Charles L. Wilkins,* Steven L. Kaberline, T. F. Lam, and T. L. Brunner

Contribution from the Department of Chemistry, University of Nebraska-Lincoln, Lincoln, Nebraska 68588. *Received April 9, 1976*

Abstract: Recent research on the use of adaptive networks of digital learning elements for chemical pattern recognition has stressed the high performance of such classifiers and their applicability to linearly inseparable data. In the present work, we apply a new performance measure, the figure of merit, and a large set of test data in a rigorous evaluation of the performance of digital learning networks. The results herein reported show that, when confronted with a large data set selected without particular consideration of the peculiarities of the network, the digital learning network continues to give good performance, although this performance is substantially below the levels previously reported. A comparison of the performance of the digital learning network classifiers with that of a set of linear discriminant functions indicates similar levels of performance for the two types of classifier.

The formal problem of pattern recognition can be approached from the viewpoint of various paradigms.[2] Those models based on the establishment of templates of various sorts resemble what one might expect in a biological pattern recognition apparatus. At the same time, abstract or ad hoc algorithms based on purely mathematical notions of pattern resemblance can also function quite successfully.

Pattern classifiers applied to chemical data have generally been of the abstract or ad hoc type. Specifically, most chemical applications have employed linear discriminant functions computed by error correction feedback[3] or by sequential simplex methods.[4] Some use has been made of the k nearest neighbor algorithm[5] as well as certain other abstract statistical methods.[6]

Recently, Stonham et al.[7,8] have described the machine recognition of chemical classes from a limited group of mass

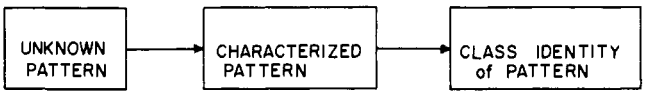


Figure 1. Stages in the pattern recognition process.

spectra using a method which is less abstract than those referred to above. Their approach, based on the adaptive digital learning network, stems from the work of Bledsoe and Browning on *n*-tuple sampling for pattern recognition.[9] The initial reports[7,8] indicated that this approach might be significantly superior to the use of pattern classifiers of the kind previously applied to chemical data.

As Uhr points out,[10] the two most basic processes in pattern recognition may be thought of as *characterization* of the unknown pattern and *classification* of the unknown pattern (see Figure 1). In the linear discriminant and *k* nearest neighbor methods, a pattern is characterized simply as a geometric point in $n + 1$-dimensional space (where n is the number of features comprising the pattern). In contrast, *n*-tuple sampling characterizes a pattern by analyzing it into subpattern units of n features each (where n is the *n*-tuple parameter, which may range from one to the total number of features, but has typically been in the range of 2 to 10). Each of these n feature subpatterns is associated with one portion of a template which is used in the classification process. Uhr has observed that modifications of this approach can be suggestive of a variety of plausible biological pattern recognition processes.[11]

In the classification stage, the linear discriminant function method depends on the existence of an n-dimensional hyperplane, which divides the $n + 1$-dimensional pattern space into two regions. A pattern is classified according to whether the point which characterizes it lies on one or the other side of this decision hyperplane. Since the hyperplane effects a strictly binary classification, a multicategory classifier constructed in this way will employ several hyperplanes. The adaptive digital network, in contrast, employs a template for each category. The pattern, characterized by its collection of *n*-tuples, is compared against each template and is assigned to the category whose template it matches most closely.

"Learning" or "training" (depending on one's educational philosophy) is the process by which the hyperplanes or the templates are developed from collections of patterns of known identity. Although numerous methods for developing hyperplanes have been advanced,[12] the simplest is the error correction feedback method,[3] which has been employed in the majority of chemical pattern recognition experiments. In this method, one begins with an arbitrary hyperplane and attempts to classify known patterns with it. Each time the hyperplane incorrectly classifies a pattern, the hyperplane is moved in its space so as to classify that pattern correctly. Application of linear programming techniques can give an improved hyperplane for any particular set of inseparable training data.[17] Training a template in the digital learning network method consists of submitting known patterns to an initially blank template. Those portions of the template corresponding to *n*-tuple subpatterns present in the training patterns are, in effect, "colored in" (see Appendix A).

Previous Work with Digital Learning Networks (DLNs)

Attention has been drawn to the need for objective criteria in evaluating the performance of pattern classifiers.[13,14] At the very least, a clear distinction must be made between the ability of a classifier to correctly classify patterns which were used in training (recognition) and the ability to classify patterns which were not present in the training set (prediction). For binary classifiers, where the classification is "class member" vs. "nonmember", it has been shown[14] that the commonly used "percent correct classifications" (number of patterns correctly classified divided by the total number of patterns in the test set) is inadequate as the single measure of performance, since this quantity depends on the distribution of patterns between class members and nonmembers. For a multicategory classifier, however, with which each pattern is assigned a definite class membership, the percent correct classification is a meaningful measure of overall performance, as long as recognition and prediction performances are kept separate.

Table I. The 440-Compound Data Set of Stonham and Coworkers[7,8]

Category no.	Compd class	No. of spectra
1	Methyl esters	29
2	Methyl ketones	11
3	Carboxylic acids	11
4	Ethyl esters	12
5	Higher esters, $n = 4, 5, 6$	13
6	Normal alcohols	33
7	Aldehydes	8
8	Higher ketones	10
9	Secondary alcohols	29
10	Substituted alcohols	14
11	Diesters	14
12	Substituted keto acids	8
13	1-Phenyl alkanes	10
14	Terpenes	18
15	*n*-Phenyl alkanes, $n \neq 1$	12
16	Aliphatic amines	22
17	Mercaptans	13
18	Sulfides	12
19	Straight-chain alkenes	14
20	Alkanes	34
21	Nitriles	7
22	Alkynes	24
23	Substituted pyrazines	6
24	Substituted phenols	19
25	Furans	8
26	Pyrroles	9
27	Thiophenes	27
28	Aromatic esters	13

Using the percent correct classification as a measure, we may examine the performance of digital learning network classifiers in their most challenging chemical test prior to this work. Stonham et al. assembled a collection of 440 mass spectra of organic compounds, which they divided into 28 rather specific chemical classes.[8] These classes are listed in Table I. The spectra (360) were used to train the 28 templates by means of the "optimum training sequence" procedure described by those authors.[7,8] In this procedure, those compounds least well recognized by a template are successively singled out for inclusion in the training set; this approach attempts to insure that the training set is representative of the particular compound class in question, but does not include unnecessary (and possibly misleading) information. In their experiment, the percent correct recognition of the 360 training compounds was 100% and the percent correct prediction of the 80 compounds not present in the training set was 97.5%.

The Data Set

As part of our program to develop a generally applicable on-line pattern recognition system for inclusion with computer-controlled mass spectral or nuclear magnetic resonance installations, we have explored the adaptive digital network as an alternative to the linear discriminant functions which have been our primary focus. In attempting to make our testing representative of the problem environment which would be encountered in an analytical laboratory, it has been our policy to utilize large data sets and to avoid careful preselection of data for pattern recognition experiments. Thus, for the present

Table II. The 1252-Compound Data Set

Category no.	Compd class	No. of spectra
1	Arenes	249
2	Aldehydes and ketones	96
3	Ethers	103
4	Aliphatic alcohols	185
5	Phenols	84
6	Carboxylic acids	51
7	Thiols	135
8	Esters	125
9	Amines	131
10	Amides	56
11	Nitriles	37

study we assembled a data set of 1252 mass spectra, comprising 11 rather broad chemical classes, by picking spectra sequentially from our file of 18 806 spectra.[15] Polyfunctional compounds were not avoided, although compounds which would have belonged to two or more of the 11 classes were discarded.[16] The 11 chemical classes are shown in Table II.

Implementation of the DLN Classifiers

Although the adaptive linear network concept lends itself to implementation in semiconductor hardware, as emphasized by Stonham et al.,[7,8] we have employed a software simulation of the network in our experiments. Our program was written in Fortran IV and was run on an IBM 360/65 computer. A flow chart of our procedure is given in Appendix B and a description of the training and classification algorithms is given in Appendix A. Particular importance attaches to the array which is used to represent the memory of the digital learning network. In our experiments, this array was in some cases declared to be LOGICAL, constraining the contents to be "0" 's and "1" 's; this configuration mirrors the character of the hardware digital learning net. In other cases, the memory array was declared to be INTEGER, so that repeated "hits" in a particular region of the template would accumulate, rather than producing no additional effect as with the LOGICAL array. Although this modification precludes the straightforward hardware implementation of the digital learning network originally envisioned, the INTEGER array version has led to some insights regarding the characteristics and limitations of the DLN as a chemical pattern recognizer.

Training a DLN template differs significantly from training a hyperplane discriminant function. The set of training compounds selected for training a hyperplane is generally chosen on the basis of availability. Some attention may be given to selecting compounds likely to be representative of the particular category of interest, but, in general, the more training compounds used, the better the classifier will perform in prediction tests. The training set must contain both compounds belonging to the category in question and a roughly similar number of compounds not belonging to that category. Also, at least a 3/1 ratio of patterns to descriptors/pattern is required to ensure meaningful experiments.

In contrast, the training set for a DLN template contains only compounds belonging to the category to be represented by the template.

Evaluation of DLN Classifiers

Tables III and IV summarize the percent correct prediction values for the various DLN classifiers developed in this study. These values are meaningful for comparisons among multicategory classifiers developed under various conditions, although they will not be suitable for subsequent comparisons with binary hyperplane classifiers.[14] The results in Table III illustrate the dilemma which is encountered in training a DLN based on a LOGICAL memory array. For small numbers of training compounds, the recognition ability of the classifier, not surprisingly, is very good. The predictive performance is moderately good; indeed, the ability to achieve nontrivial prediction performance with minimal training seems to be a characteristic of the DLN method.

However, as increasing numbers of training compounds are employed in order to incorporate more information in the classifier, the recognition performance falls off rapidly. This effect results from "blurring" of the templates by spurious peaks in the spectra and from "overgeneralization" as described by Stonham et al. Since the LOGICAL or binary memory array contains only "0" 's and "1" 's and there is no mechanism for erasing "1" 's, spurious or infrequent peaks in the training compounds will contribute as much to the character of the template as will highly characteristic peaks. Thus, as more training is incorporated, a DLN classifier begins to give high responses for compounds which are not in fact members of the class represented by that particular template.

In the task of prediction, as opposed to recognition, extra training improves the DLN performance up to a point. Presumably, this effect arises because the templates have been exposed to a more diverse population in training. Here too, however, overtraining finally begins to produce difficulty in discriminating categories and the performance falls off.

Stonham et al. have suggested that the problem of overtraining can be dealt with by carefully optimizing the training procedure, selecting for training only enough compounds to give a desired level of recognition performance, and by choosing these compounds to be those which are least well recognized, presumably ensuring that the templates will have as diverse as possible training experience.[7,8] However, our tests with a larger, more general mass spectral data set show that the "optimum training sequence" strategy does not necessarily have the desired effect. Included in Table III are the performances of learning networks trained using the "optimum training sequence" strategy.[7,8] In order to train the individual classifiers to a maximum response of 128, 467 training patterns were required. Evidently, the capability of the classifiers for identifying the remaining 785 compounds was not optimized by this procedure, since training with a randomly chosen group of 467 compounds (distributed among the 11 categories in proportion to the actual numbers of compounds in those categories) produced substantially better predictive performance. Examination of the errors made by the classifiers trained in this manner showed evidence of overtraining: numerous patterns achieved maximum response (=128) in more than one category. Several patterns scored 128 in seven or more categories. In order to reduce the overtraining problem, another set of templates was trained using the "optimum training sequence", with training to a response of 120 rather than 128. This level of training required only 144 training patterns, in contrast with the 467 training patterns required to achieve responses of 128; it is not surprising that the extensive extra training required to achieve maximum response would also tend to overgeneralize the templates and thereby reduce their discrimination. The reduced training did indeed improve the performance of the classifiers relative to the 467-pattern "optimum" training set, as seen in Table III. However, the performance was still not as good as that achieved with a comparable number of randomly selected training compounds.

As an alternative approach to the problem of overtraining, we have experimented with an INTEGER memory array, as mentioned earlier. With the INTEGER memory array, a *n*-tuple subpattern appearing frequently among the training patterns

Table III. Overall DLN Performance: LOGICAL Memory Array[a]

Patterns in training set	Randomly selected training set		"Optimum training sequence"			
			Response = 128		Response = 120	
	Recognition	Prediction[b]	Recognition	Prediction	Recognition	Prediction
64	100%	59%				
128	98%	64%				
144	95%	63%			99%	56%
302	89%	62%				
467	82%	59%	84%	48%		
604	73%	54%				
927	65%	55%				

[a] Values given are percent of patterns classified correctly. [b] For prediction, the number of patterns in the test set is 1252 minus the number in the training set.

Table IV. Overall DLN Performance: INTEGER Memory Array[a]

Patterns in training set	Randomly selected training set		"Optimum training sequence"			
			Response = 128		Response = 120	
	Recognition	Prediction[b]	Recognition	Prediction	Recognition	Prediction
64	80%	41%				
128	76%	57%				
144	77%	56%			76%	38%
302	76%	65%				
467	76%	67%	64%	58%		
604	69%	66%				
927	71%	65%				

[a] Values given are percent of patterns classified correctly. [b] For prediction, the number of patterns in the test set is 1252 minus the number in the training set.

is expected to have a greater effect ("to make more of an impression") upon the template than an infrequently occurring subpattern. Thus, one would hope to minimize the undesirable effect of spurious or highly individual features in the training process.

Table IV summarizes the performance of DLNs employing INTEGER memory arrays. These trials are based on the same training sets used in testing the LOGICAL array-based classifiers. As anticipated, the INTEGER memory array seems less susceptible to overtraining than the LOGICAL array. Additional training of DLNs employing INTEGER memory arrays produced improved predictive performance until about 500 compounds had been employed in training. For DLNs based on LOGICAL memory arrays, performance began to suffer after about one-fourth as much training. However, it is also apparent that the LOGICAL array-based DLNs performed nearly as well after relatively little training (128 training patterns) as did the INTEGER array-based DLNs after much more extensive training (467 training patterns). This observation further supports the idea that the ability to perform well with little training is a characteristic of DLN classifiers. It must be noted that, with both types of memory array, the randomly chosen training set produced a better multicategory classifier than did the "optimum training sequence" strategy.

The multicategory classifier under consideration here consists of a set of templates, one template for each functional group category. Since it is not necessarily the case that all templates are equally effective, it is of interest to consider the classifier's performance on individual categories. This analysis is conveniently performed by reference to the "figure of merit", M, based on the information gain[13,14] contributed by a classifier. For this purpose, we consider the multicategory classifier to be an adjustable binary classifier and examine its performance on individual pattern categories.[14]

The performances on individual categories are summarized in Tables V and VI. Examination of these data reveals variations among the categories not evident from the overall performance data in Tables III and IV. With a LOGICAL memory array as the classifier template medium (Table III), performance on category 1 patterns improves up to about 144 training compounds and then begins to fall off. Although this same tendency to display a performance peak is more or less evident for categories 1, 2, 3, 5, 7, 9, and 11, this is not the case, for example, for category 4, which shows decreased performance for training beyond 64 patterns, nor for category 6, for which performance appears virtually unaffected by various amounts of training. Performance on categories 8 and 10 fluctuates erratically as training is increased, indicating a dependence more on the identities of the training compounds than on their numbers. Furthermore, those categories showing peaks of performance do not all peak at the same level of training. When performance on individual categories is examined, it is seen that the "optimum training sequence" strategy does indeed produce superior results for some categories (5, 6, 7, 10, and 11), but not for others.

A similar picture emerges from the figure of merit data on INTEGER array-based DLN classifiers (Table VI). Significant variation in behavior among the categories is apparent. Performance on categories 2, 4, 5, 8, and 11 improves as originally postulated for the INTEGER memory array, increasing performance with increasing training without the impairment of overtraining and reaching a higher level of performance than the corresponding LOGICAL array classifiers. For categories 1, 3, and 7, however, performance of the INTEGER array-based classifiers is worse than that of the LOGICAL array-based classifiers. Fluctuations in performance on classes 8 and 11 again indicate dependence on the identity of the training patterns. With INTEGER array classifiers, the "optimum training sequence" gives better performance than a random training sequence for classes 3, 5, 6, 7, 10, and 11.

Table V. DLN Performance by Functional Group Category: LOGICAL Memory Array[a]

	Number of patterns in training set						
Category	64	128	144	302	467	604	927
1	0.67	0.69	0.69 [0.54]	0.63	0.59 [0.52]	0.57	0.50
2	0.12	0.31	0.30 [0.24]	0.23	0.18 [0.10]	0.16	*b*
3	0.05	0.11	0.08 [0.08]	0.18	0.20 [0.14]	0.17	0.05
4	0.28	0.24	0.23 [0.11]	0.25	0.12 [0.09]	0.13	0.08
5	0.04	0.29	0.30 [0.42]	0.39	0.36 [0.50]	0.32	0.29
6	0.06	0.04	0.04 [0.01]	0.06	0.02 [0.07]	0.04	0.01
7	0.62	0.77	0.79 [0.87]	0.74	0.75 [0.77]	0.67	0.67
8	0.21	0.14	0.13 [0.24]	0.18	0.25 [0.15]	0.15	0.25
9	0.20	0.28	0.29 [0.07]	0.23	0.25 [0.05]	0.08	0.07
10	0.14	0.23	0.17 [0.15]	0.12	0.12 [0.29]	0.13	0.25
11	0.31	0.50	0.51 [0.68]	0.46	0.29 [0.87]	0.47	0.45

[a] Values given are figures of merit for random training sequence; values in brackets are for "optimum training sequence". [b] No patterns in this category were left in the test set after training with 927 patterns.

Table VI. DLN Performance by Functional Group Category: INTEGER Memory Array[a]

Category	64	128	144	302	467	604	927
1	0.10	0.32	0.31 [0.24]	0.50	0.61 [0.54]	0.68	0.66
2	0.004	0.18	0.08 [0.001]	0.42	0.39 [0.11]	0.43	*b*
3	0.13	0.16	0.17 [0.02]	0.07	0.14 [0.16]	0.08	0.05
4	0.18	0.25	0.26 [0.01]	0.30	0.32 [0.17]	0.28	0.34
5	0.20	0.37	0.38 [0.25]	0.46	0.51 [0.55]	0.75	0.70
6	0.02	0.08	0.09 [0.05]	0.13	0.05 [0.06]	0.04	0.00
7	0.24	0.36	0.68 [0.65]	0.76	0.67 [0.87]	0.72	0.72
8	0.10	0.12	0.08 [0.09]	0.24	0.30 [0.09]	0.20	0.19
9	0.42	0.43	0.39 [0.21]	0.34	0.34 [0.33]	0.26	0.22
10	0.21	0.25	0.20 [0.13]	0.22	0.22 [0.30]	0.15	0.19
11	0.41	0.55	0.57 [0.56]	0.61	0.58 [0.82]	0.52	0.76

[a] Values given are figures of merit for random training sequence; values in brackets are for "optimum training sequence". [b] No patterns in this category were left in the test set after training with 927 patterns.

Performance Comparison: DLN and Linear Discriminant Classifiers

The question of the comparative performance of these DLN classifiers and the more familiar linear discriminant classifiers naturally arises. The DLN classifier operates as a multicategory classifier, assigning each pattern to one of the 11 categories, while a linear discriminant function simply effects a binary classification (class member or nonmember). Nonetheless, we can compare the two classifier types by examining the performance of a DLN classifier on individual categories and comparing the performance with that of linear discriminant functions trained for discrimination of the same categories.

In anticipation of this comparison, we have employed the same 1252 spectrum data set for the training and testing both of the DLN classifiers described here and of several sets of linear discriminant classifiers (weight vectors), the development of which is described elsewhere.[17] These linear discriminant classifiers were developed using error correction feedback with various numbers of features by simplex optimization of weight vectors derived from error correction feedback and by 60-feature simplex pattern recognition. Since

Table VII. Comparative Performance by Functional Group Category of Best Linear Discriminant Classifiers with Best DLN Classifiers

	Best linear discriminant			Best DLN (LOGICAL)			Best DLN (INTEGER)		
Category	$p(j\|1)$	$p(n\|2)$	M	$p(j\|1)$	$p(n\|2)$	M	$p(j\|1)$	$p(n\|2)$	M
1	0.92	0.95	0.63 (20-SIM)[a]	0.98	0.93	0.69	0.84	0.99	0.68
2	0.87	0.96	0.50 (60-LLM)	0.55	0.98	0.31	0.67	0.98	0.43
3	0.77	0.85	0.21 (60-LLM)	0.32	0.99	0.20	0.52	0.93	0.17
4	0.82	0.89	0.34 (60-LLM)	0.83	0.82	0.28	0.92	0.78	0.34
5	0.92	0.95	0.53 (20-SIM)	0.94	0.93	0.50	0.89	0.99	0.75
6	0.71	0.93	0.25 (60-SIM)	0.16	0.99	0.07	0.17	1.00	0.13
7	0.93	0.95	0.55 (20-SIM)	0.97	0.99	0.87	0.95	0.997	0.87
8	0.76	0.87	0.23 (25-SIM)	0.76	0.86	0.25	0.46	0.99	0.30
9	0.95	0.94	0.57 (60-LLM)	0.61	0.95	0.29	0.74	0.96	0.43
10	0.73	0.95	0.32 (60-LLM)	1.00	0.73	0.29	0.54	0.98	0.30
11	0.88	0.97	0.49 (60-LLM)	1.00	0.99	0.87	1.00	0.99	0.82
			$M_{ave} = 0.42$			$M_{ave} = 0.42$			$M_{ave} = 0.47$

[a] The number of features and method used to develop the discriminant. SIM is simplex and LLM is linear learning machine (error correction feedback) method.

there is no reason why all the classifiers in an integrated pattern recognition system need all have been developed by the same method, we have chosen to compare, for each functional group category, the best linear discriminant classifier (from among those reported in ref 17) with the best LOGICAL array-based and INTEGER array-based DLN classifiers. This comparison is shown in Table VII. The two types of DLN classifier must be reported separately because LOGICAL memory array templates cannot be combined with INTEGER array templates in a single classifier network. For completeness, both the figures of merit, M, and the class conditional probabilities (predictive abilities on class members and nonmembers) are included in Table VII.

An interesting comparison results if we consider the linear discriminant and DLN classifiers to have "comparable" performance if their figures of merit differ by no more than 0.05. With this somewhat arbitrary definition, we see from Table VII that the LOGICAL array-based digital learning network gives performance comparable to the linear discriminant classifier in four cases (categories 3, 5, 8, and 10), superior performance to the linear discriminant in three cases (categories 1, 7, and 11), and worse performance in four cases (categories 2, 4, 6, and 9). Comparison of the INTEGER array-based DLN classifiers with the linear discriminant classifiers shows that the DLN classifier gives performance comparable to the linear discriminant in four cases (categories 1, 3, 4, and 10), superior performance to the linear discriminant in four cases (categories 5, 7, 8, and 11), and worse performance in three cases (categories 2, 6, and 9). There is thus no clear-cut overall superiority of one classifier type over the other. For certain categories, such as 7 and 11, the DLN classifier seems to have a distinct advantage. For other categories, such as 2 and 9, the linear discriminant gives decidedly superior results. It is interesting to note that the average figures of merit for the three classifier types are very nearly the same (M_{ave} = 0.42 for the linear discriminant classifiers; M_{ave} = 0.42 for the LOGICAL array-based DLN classifiers; and M_{ave} = 0.47 for the INTEGER array-based DLN classifiers).

It is concluded that classifiers based on adaptive digital learning networks as we have described and implemented them give performance which is, in comparison with binary linear discriminant functions developed by error correction feedback and simplex optimization, neither better nor worse on the average. It appears that, for certain categories of functional groups or for certain situations (as, for example, when few training patterns are available), the digital learning network classifiers may offer advantages.

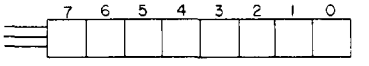


Figure 2. 8-Bit digital learning element.

Summary

Experiments with a large and general data set using objective performance measures have indicated that the training of an adaptive digital learning network classifier cannot be undertaken in a cavalier fashion. The subsequent performance of such classifiers was very sensitive to the makeup of the training set with regard both to the identities of the training patterns and the number of training patterns employed. Particularly, the problem of overtraining was severe and neither the "optimum training sequence" strategy nor the use of an INTEGER memory array could be depended upon to yield an optimum classifier; these results do not, however, preclude the possibility of inventing an a priori strategy which would produce superior DLN classifiers. At best, without resorting to reducing the size of the data set or modifying the category definitions, the performance we were able to achieve (measured in terms of percent correct prediction) was far below that which was reported by Stonham and co-workers on a smaller and probably more carefully selected data set. Nonetheless, digital learning network classifiers gave performance comparable with that of linear discriminant functions and may offer advantages in certain situations.

Acknowledgment. Support of this research under Grant MPS-74-01249 from the National Science Foundation is gratefully acknowledged. We also gratefully acknowledge the University of Nebraska Research Council, which provided partial support for the purchase of the computer-readable mass spectra data set.

Appendix A

The Digital Learning Network Algorithm. The basic unit of the n-tuple learning machine as described by Stonham and co-workers[7,8] is the "digital learning element", which may be thought of as a storage register (Figure 2). This storage register is associated with a randomly chosen group of n pattern elements from the pattern being presented to the learning machine for training or for recognition/prediction; this group is an n-tuple subpattern. The number of storage locations in the storage register is 2^n, where n is the number of pattern elements in the n-tuple; thus, 3-tuple sampling requires a $2^3 = 8$ bit register.

The patterns used with this method are binary patterns. Thus, in any n-tuple subpattern there are 2^n possible configurations; for a 3-tuple, these configurations are (000), (001), (010), ... , (111). The configuration or bit pattern of the n-tuple subpattern is used to address one of the 2^n locations in the digital learning element or storage register. In the training stage, a "1" is written into the bit position addressed by the n-tuple subpattern. In the recognition/prediction stage, the bit position addressed is read rather than written.

A "digital learning network" consists of a group of digital learning elements. The number of learning elements used in the network will depend on the n-tuple value (n), the number of pattern features being examined, and the sampling redundancy desired. For example, if 90 pattern features are to be subjected to 3-tuple sampling with no feature being sampled twice, then 30 eight-bit learning elements will be required to make a learning network. One learning network is prepared for each category of patterns to be classified.

To train a digital learning network for a particular category, a pattern belonging to that category is presented to the network, which has been initialized to contain all "0" 's. Each learning element in the learning network is associated with its own n-tuple subpattern in the pattern being studied and the bit pattern in that n-tuple subpattern addresses just one of the 2^n bit positions in the learning element. In each location so addressed, a "1" is now written. Now, there will be just one "1" in each learning element in the network. Further training proceeds by presenting to the network more patterns from the given category.

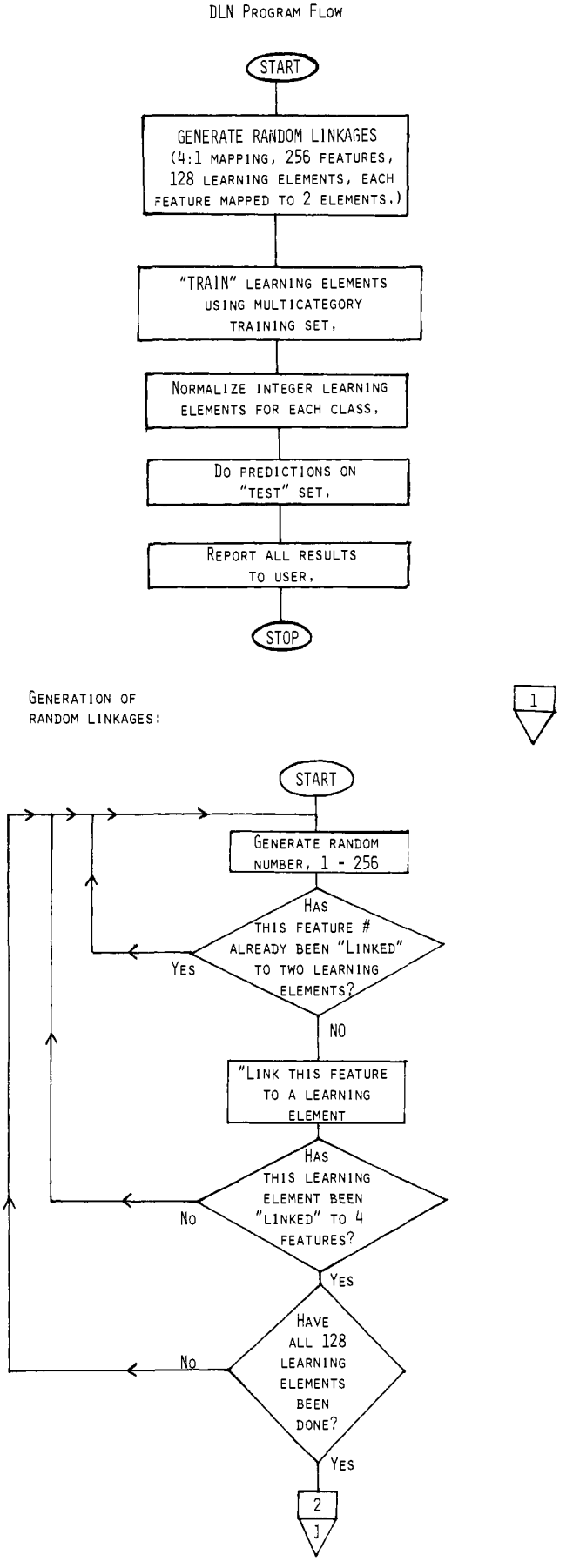
When recognition of a pattern is desired, the pattern is presented to the network just as for training: the n-tuple subpatterns are used to address locations in the learning elements comprising the learning network. However, instead of *writing* into the memory array, we now *read* the contents of the locations being addressed and add up these values. Any n-tuple subpattern which was present in a training pattern will thus address a location which had been written with a "1" during training; n-tuple subpatterns which were not present in any training pattern will address locations which are unchanged from their initial "0" value.

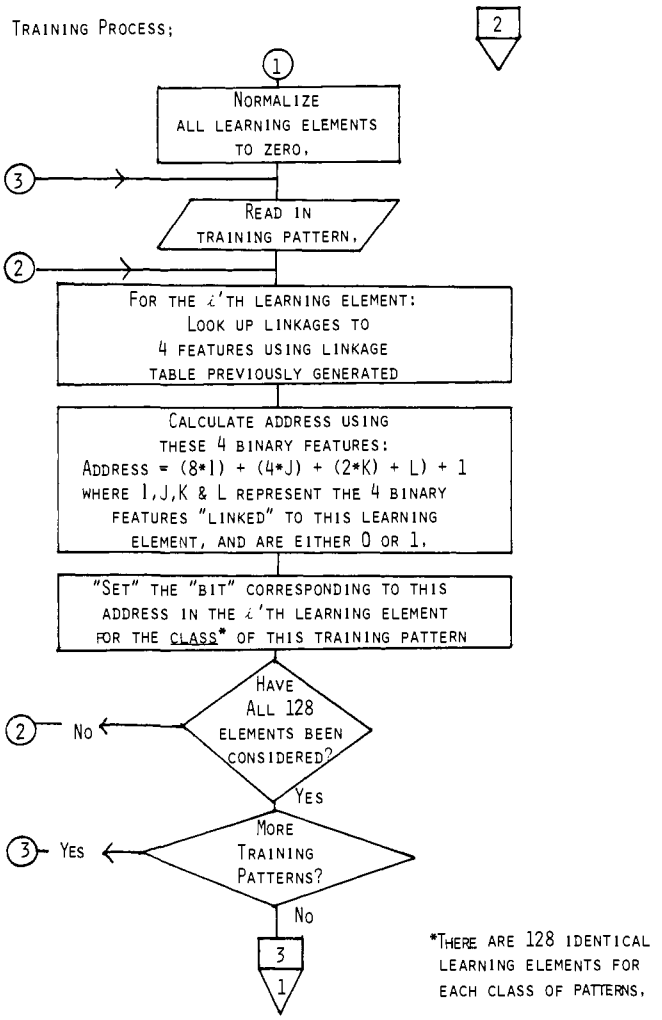
Thus, if a pattern which was used for training is presented for recognition, all the memory locations addressed by its n-tuple subpatterns will have been previously set to "1" 's during training; and, when the addressed contents are summed, the sum will be exactly equal to the number of learning elements in the network. This is the maximum "score" or response that a pattern can achieve. A pattern containing some n-tuple subpatterns not encountered during the training of the network will have a response less than the maximum.

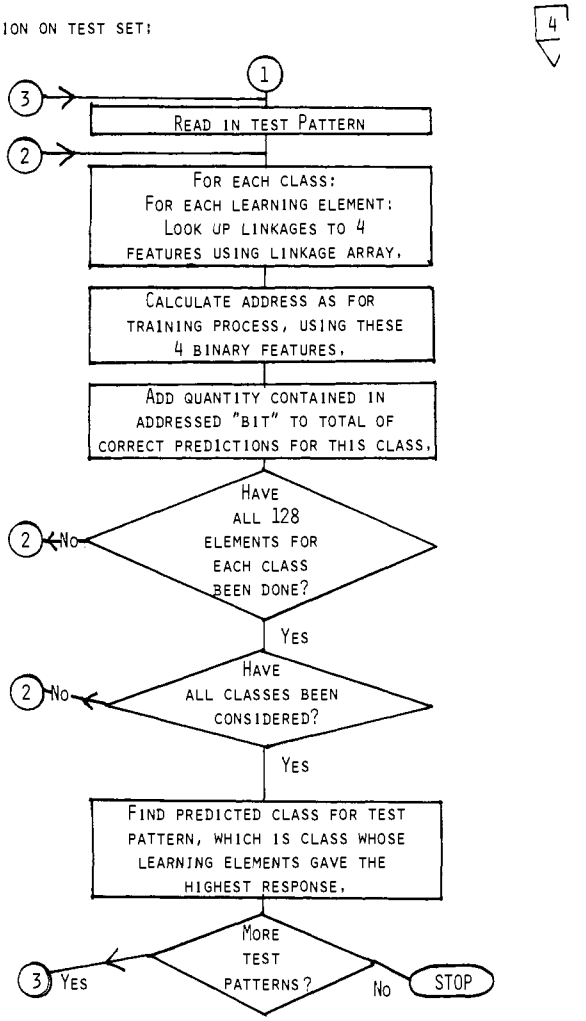
Multicategory classification is achieved by submitting an unknown pattern to various trained learning networks, each network representing one category. The network with which the pattern achieves the highest response gives the class identity of the pattern.

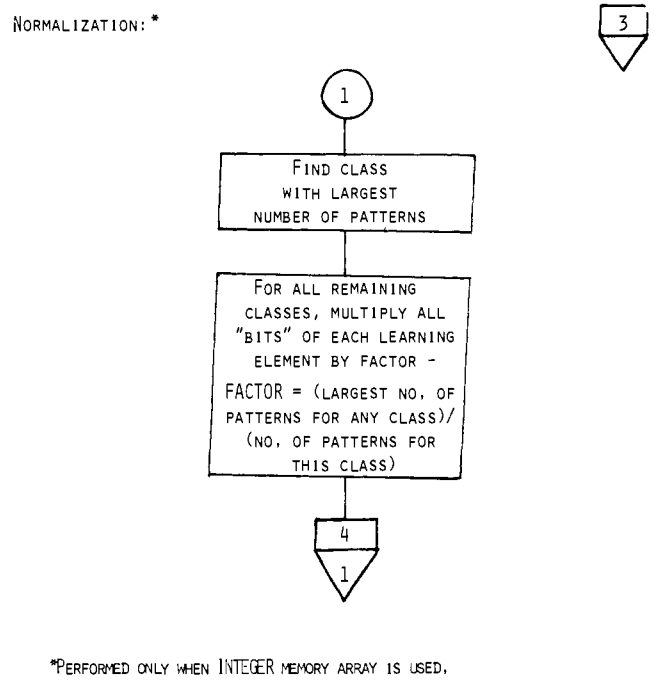
In our modification employing INTEGER memory arrays, each storage location in each digital learning element is capable of storing an integer. The contents of a particular location are incremented each time that location is addressed by a subpattern in a training spectrum. Since this procedure makes the values in the memory locations in a template dependent on the number of patterns used to train that template, it is necessary to compensate for the varied amounts of training received by the different templates. This was accomplished following training by multiplying each template by a scale factor, n_{max}/n_i, where n_{max} is the number of patterns in the most populous training set and n_i is the number of patterns in the set being scaled. Further, since the subpattern (0000) is the most frequently encountered but carries the least information,[7] the contents of the storage locations addressed by that subpattern were not utilized in the recognition or prediction process when INTEGER memory arrays were used.

Appendix B









References and Notes

(1) L. J. Soltzberg is a visiting Associate Professor during the 1975–1976 academic year from Simmons College, Boston, Mass.
(2) L. Uhr, "Pattern Recognition, Learning, and Thought", Prentice-Hall, Englewood Cliffs, N.J., 1973, Chapter 1.
(3) P. C. Jurs and T. L. Isenhour, "Chemical Applications of Pattern Recognition", Wiley-Interscience, New York, N.Y., 1975.
(4) G. L. Ritter, S. R. Lowry, C. L. Wilkins, and T. L. Isenhour, *Anal. Chem.*, **47**, 1951 (1975).
(5) B. R. Kowalski and C. F. Bender, *Anal. Chem.*, **44**, 1405 (1972).
(6) J. B. Justice, Jr., D. N. Anderson, T. L. Isenhour, and J. C. Marshall, *Anal. Chem.*, **44**, 2087 (1972).
(7) T. J. Stonham and M. A. Shaw, *Pattern Recognition*, **7**, 235 (1975).
(8) T. J. Stonham, I. Aleksander, M. Camp, W. T. Pike, and M. A. Shaw, *Anal. Chem.*, **47**, 1817 (1975).
(9) W. W. Bledsoe and I. Browning, "Pattern Recognition and Reading by Machine", reprinted in "Pattern Recognition", L. Uhr, Ed., Wiley, New York, N.Y., 1966, pp 301–316.
(10) L. Uhr, ref 2, pp 27–29.
(11) L. Uhr, ref 2, p 72.
(12) R. O. Duda and P. E. Hart, "Pattern Recognition and Scene Analysis", Wiley, New York, N.Y., 1973, Chapter 5.
(13) H. Rotter and K. Varmuza, *Org. Mass Spectrom.*, **10**, 874 (1975).
(14) L. J. Soltzberg, C. L. Wilkins, S. L. Kaberline, T. F. Lam, and T. R. Brunner, *J. Am. Chem. Soc.*, preceding paper in this issue.
(15) E. Stenhagen, S. Abrahamssen, and F. W. McLafferty, "The Registry of Mass Spectral Data", Wiley-Interscience, New York, N.Y., 1974.
(16) This is not the case for phenyl compounds (class 1) and phenols (class 5), for which the class definitions are not mutually exclusive.
(17) T. F. Lam, C. L. Wilkins, T. R. Brunner, L. J. Soltzberg, and S. L. Kaberline, *Anal. Chem.*, in press.