H. Kobayashi
L. R. Bahl

Image Data Compression by Predictive Coding I: Prediction Algorithms

Abstract: This paper deals with predictive coding techniques for efficient transmission or storage of two-level (black and white) digital images. Part I discusses algorithms for prediction. A predictor transforms the two-dimensional dependence in the original data into a form which can be handled by coding techniques for one-dimensional data. The implementation and performance of a fixed predictor, an adaptive predictor with finite memory, and an adaptive linear predictor are discussed. Results of experiments performed on various types of scanned images are also presented. Part II deals with techniques for encoding the prediction error pattern to achieve compression of data.

Introduction

Data sources such as facsimile transmission signals and digitized images to be stored in a computer memory contain a substantial amount of redundancy. Source coding [1] techniques (also called data compaction or compression) can be used to efficiently encode the outputs of such sources. There are two obvious applications of these compression methods. The first is in communication systems. By encoding the source, we can transmit the information over a communication channel in a shorter time period. Alternatively, we could use a channel with a smaller bandwidth to transmit the coded data (hence the often used term "bandwidth compression"). The second application is in storage systems, which can be used more efficiently by reducing the amount of data to be stored.

In this paper we discuss some theory and methods for compressing two-dimensional black and white image data through the use of predictive coding. Part II [2] describes encoding techniques designed to achieve compression. Before going into detailed discussions on the particular approaches taken, we briefly review some of the recent progress in the general area of image compression. Techniques in the art of compressing two-dimensional image data can be classified into two categories: 1) time-domain (or space-domain) encoding and 2) transform-domain encoding. The time domain techniques that appear practical are mostly of the prediction-comparison type. This includes schemes like delta modulation and differential pulse code modulation [3]. Most studies of such systems are based on the classical communication theory approach. The first information theoretic treatment of this subject was done by Elias [4], who called the technique "predictive coding." In predictive coding the dependence inherent in the data can be removed by a good predictor that transforms the original data into a form such that successive data symbols are nearly independent of each other. The transformed data can then be encoded by techniques applicable to independent sources. Application of predictive coding to compression of pictorial data has been reported by Wholey [5] and by Arps [6]. Recent results in the theory of data compression systems that use prediction and interpolation have been discussed by Balakrishnan [7] and by Davisson [8]. Applications of their studies to weather satellite pictures are reported by Kutz and Sciulli [9].

The transform-domain methods [10–12] include the application of Fourier, Hadamard-Walsh, Karhunen-Loeve, and other transforms. In all these methods a block of data samples is decomposed into a set of coefficients of orthogonal functions, and the coefficients are transmitted. Compression of data is obtained by eliminating insignificant coefficients or by reducing the number of levels of quantization in the transform domain. There is, of course, some degradation in the reconstructed image. With the advent of fast transform methods [13–15] that are particularly suited for high-speed digital computers, the transform techniques have recently received considerable attention.

For comprehensive treatments of the subject, the reader is referred to the recently published books by Huang and Tretiak [16], Andres [17], and Rosenfeld [18], and to a special issue of journal papers [19].

Image-data compression systems with predictive coding

In the present work our main interest lies in efficient source coding of two-dimensional digital image data, i.e., data from sources that are discrete both in space and in amplitude [20]. Typical examples are the quantized values of facsimile scanner output and graphic display data of computer systems. We limit ourselves to those cases where the amplitude is quantized to just two levels.

Figure 1 illustrates a communication system that uses predictive coding. Assume a two-dimensional data source $s(i, j)$, $1 \leq i \leq I$, $1 \leq j \leq J$, in which the value of each picture element (pel) s can be either 0 (white) or 1 (black). We can expect the value of pel $s(i, j)$ to be closely related to the values of its neighboring pels, $s(i-1, j-1)$, $s(i-1, j)$, $s(i-1, j+1)$, $s(i, j-1)$, etc. (Fig. 2). It is therefore possible to predict $s(i, j)$ with a high probability of success from the values of neighboring pels. This is the function of the predictor in Fig. 1. The set of points on which the predictor bases its prediction of the generic pel (i, j) is called the memory set M of the predictor. For example, in the case of the 4-pel predictor of Fig. 2, the memory set is

$$M = \{s(i-1, j-1), s(i-1, j), s(i-1, j+1), s(i, j-1)\}. \quad (1)$$

The output $\hat{s}(i, j)$ of the predictor module is compared with the actual value $s(i, j)$ and an error signal $e(i, j)$ is derived from the operation

$$e(i, j) = s(i, j) \oplus \hat{s}(i, j), \quad (2)$$

where $\oplus$ means modulo 2 addition, an operation that can be realized by an EXCLUSIVE OR circuit.

If the two-dimensional data $s(i, j)$ are read by raster scanning, they are transformed into a one-dimensional time series denoted by $s(n)$, where $n = (i-1)J + j$ and $1 \leq n \leq IJ$. The sequence $s(n)$ is first read into a buffer memory. The 4-pel predictor of Fig. 2 is given at time n the values $s(n-1)$, $s(n-J+1)$, $s(n-J)$, and $s(n-J-1)$ from the memory and it generates $\hat{s}(n)$, a predicted value of $s(n)$. The original data $s(i, j)$ tend to be highly redundant so most predictions are found to be correct. This results in an error pattern $e(n)$ [or equivalently $e(i, j)$] which is very sparse in binary 1's.

The error signal is encoded by some efficient data compression scheme and is then transmitted or put into a data storage system. At the receiving end, or in retrieving the data from a storage system, the decoder recovers the error signal from its encoded form and the original data are reconstructed by passing $e(i, j)$ into a circuit that contains a predictor in its feedback loop (Fig. 1). The structure of this predictor and its prediction rule should be identical

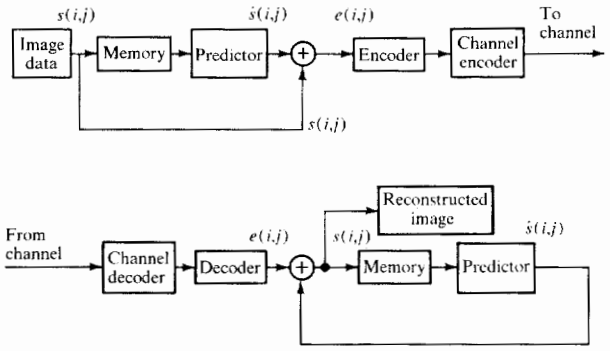


Figure 1 An image data compression system with predictive coding and decoding.

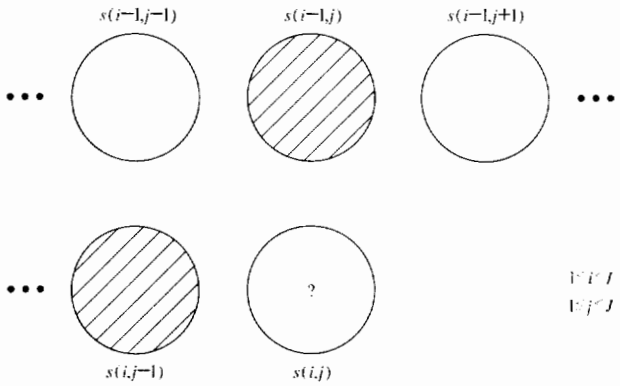


Figure 2 Two-dimensional data and their prediction based on the neighbors.

to those of the predictor at the transmitter side. The $s(i, j)$ are reconstructed according to the inverse rule of (2):

$$s(i, j) = e(i, j) \oplus \hat{s}(i, j). \quad (3)$$

The system represented by Fig. 1 can be extended to multilevel (grey level) signals, e.g., the case in which the $s(i, j)$ take on m different levels, $0, 1, \cdots, m-1$. The mod 2 additions in (2) and (3) should then be replaced by mod m subtraction and mod m addition, respectively. The prediction error pattern $e(i, j)$ is therefore an m-level pattern, with few non-zero values, since most predictions will be correct.

The information in the error signal $e(i, j)$ is precisely the information in the original image $s(i, j)$ because either can be obtained uniquely from the other. Note that prediction by itself does not achieve any compression. However, a substantial difference in the efficiency of coding may occur depending on whether one encodes $s(i, j)$ or $e(i, j)$. An efficient code for $s(i, j)$ would require that a large number N of pels be encoded simultaneously. This imposes two requirements on the encoding:

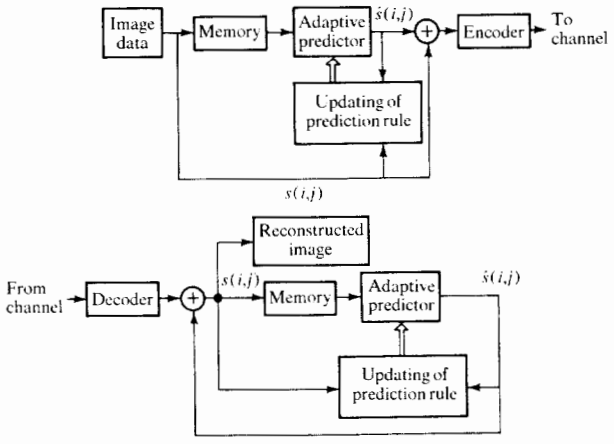


Figure 3 An image data compression system with adaptive predictor.

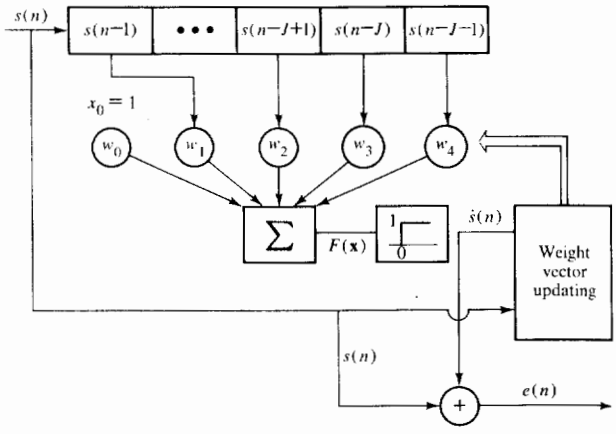


Figure 4 Adaptive linear threshold logic predictor with $N = 4$.

1. There must be an active memory that stores the past N or more terms of $s(i, j)$.
2. There must be a codebook memory that lists the codewords for all possible patterns of N terms. The number of entries in the codebook is 2^N.

Furthermore, if each block is encoded independently, the inherent dependence between adjacent blocks is not exploited. In predictive coding, on the other hand, the error sequence $e(i, j)$ is, in most cases, approximated by the output of a memoryless (zero-memory) source because the serial (or spatial) redundancy is removed by the predictor. As discussed in Part II, this latter result allows convenient encoding of data without resorting to a large codebook.

Prediction algorithms

• *Predictors and criterion for optimality*

One problem in the design of a predictor is the choice of the memory set. Clearly, points adjacent to the point to be predicted are the most useful and should be contained in the memory set. We also restrict the memory set to contain points that have been scanned prior to the point currently being predicted. (It is possible to construct look-ahead predictors which violate the above constraint, but the advantages, if any, of such predictors have not been investigated.) It is advantageous to have as large a memory set as possible, but since the complexity of the predictor grows with the size of the memory set, practical considerations generally limit the memory set to contain only a few points, generally less than 10.

For a given memory set, the "best predictor" is defined as one that requires the minimum number of bits for the transmission of its error pattern. However, we need a more specific definition to get meaningful results. The purpose of predictive coding is to remove the redundancy from the image pattern. If the predictor does a good job of redundancy removal the error pattern can be considered to be a completely random pattern. If we make the assumption that the error pattern is truly random, the compression obtained by encoding the error pattern is determined only by the prediction error probability. Therefore we choose the minimization of prediction error probability as our criterion of optimality. In practice, a predictor does not remove all of the redundancy in the image, only a substantial portion of it. Consequently the error pattern is not completely random. However, assuming the error pattern to be random is the most pessimistic assumption and it gives us a lower bound on the performance of the compression system. Another justification for choosing the minimization of prediction error probability as the optimality criterion is its practicality in analysis and implementation.

Definition A predictor is called optimum if the prediction error terms $e(i, j)$ [or $e(n)$] contain, on the average, the least number of 1's, which identify prediction errors.

If the size of the memory set is N (i.e., an N-pel predictor), the set can take on 2^N different states (or values) which we denote by M_k, $k = 1, 2, \cdots, 2^N$.

• *Fixed predictors*

Assume that the data sequence is stationary and we are given the conditional probability

$$q(1/M_k) \equiv \Pr\{s = 1 | \text{memory set} = M_k\}, \tag{4}$$

for all $k = 1, 2, \cdots, 2^N$. We further assume that prediction is performed pel-by-pel according to a fixed prediction rule. Then we obtain the following theorem for an optimal fixed prediction rule.

Theorem 1 If the memory set takes on state M_k, then

$$\hat{s} = \begin{cases} 0 \text{ if } q(1|M_k) < 0.5; \\ 1 \text{ if } q(1|M_k) \geq 0.5. \end{cases} \tag{5}$$

The minimized probability of prediction error is given by

$$P_e = \sum_{k=1}^{2^N} r(M_k) \min \{q(1|M_k), 1 - q(1|M_k)\}, \tag{6}$$

where

$$r(M_k) = \Pr \{\text{memory set} = M_k\}. \tag{7}$$

The proof of the above theorem is given by a straightforward application of the well-known Bayes' decision rule [20]. In actual applications $q(1|M_k)$ should be obtained empirically.

The appropriate choice of a memory set must be left to the designer's judgment since there seems to be no general way of defining an optimal memory set. However, one can see intuitively that performance would be improved by expanding any given memory set. This, in fact, can be proven under the same assumption made in the theorem. We state this fact as a separate theorem.

Theorem 2 The probability of prediction error of an optimal fixed predictor is not increased by expanding the memory set.

The formal proof is given in [20] and is lengthy: it is based on the concave property of the function $f(x) = \min(x, 1 - x)$.

One of the disadvantages of a fixed predictor is that the size of the decision table grows exponentially with N, and hence its dimension becomes huge when N gets large. It is possible to reduce the predictor complexity to some extent by applying techniques from switching theory, such as Karnaugh maps [21] or the Quine-McClusky reduction method [22].

• *Adaptive predictors*

A fixed predictor of the type discussed above is justified if the statistics of the data [i.e., $q(1|M_k)$] are known or available in advance (e.g., by prescanning) and if the data from the source are stationary. The adaptive predictors provide solutions that are practical when data statistics are unknown or nonstationary. The use of these adaptive predictors requires no extra channel space: The predictor at the receiver operates in exactly the same manner as the one at the transmitter side, and updating of its prediction rule can be done synchronously with that of the transmitter because the adaptation is controlled by the signal $s(i, j)$, which is available at both ends of the system (Fig. 3).

Adaptive threshold-logic predictor

The predictor output $\hat{s}$ forms a finite set of discrete levels

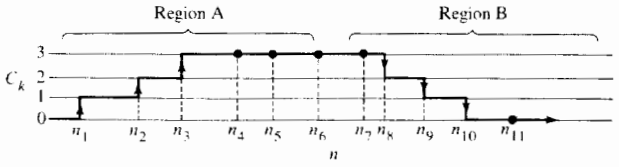


Figure 5 Typical behavior of C_k.

(0 and 1 in our case), so any of the pattern classifier or discriminant functions can be used as a predictor [23]. The simplest type of adaptive classifier is a linear learning machine or adaptive threshold logic unit (TLU).

Let $\{x_1, x_2, \cdots, x_N\}$ be values of data in the memory set of dimension N. For example, in the case of the 4-pel predictor of Fig. 4, these x's at time n are

$$x_1 = s(n-1),\ x_2 = s(n-J+1),\ x_3 = s(n-J), \text{ and } x_4 = s(n-J+1). \tag{8}$$

Define an $(N+1)$-dimensional vector $\mathbf{x}$ by

$$\mathbf{x} = (1, x_1, x_2, \cdots, x_N), \tag{9}$$

and consider the following linear function of the vector:

$$F(\mathbf{x}) = \sum_{i=1}^{N} w_i x_i + w_0 = \langle \mathbf{w}, \mathbf{x} \rangle, \tag{10}$$

where $\mathbf{w}$ is an $(N+1)$-dimensional vector,

$$\mathbf{w} = (w_0, w_1, w_2, \cdots, w_N), \tag{11}$$

and the w_i's are real numbers. For practical purposes the w_i's can be integers. This condition is met automatically if the initial values of $\mathbf{w}$ and α of (13) are integers. Then the prediction is done according to the following simple rule:

$$\hat{s} = \begin{cases} 0 \text{ if } F(\mathbf{x}) < 0; \\ 1 \text{ if } F(\mathbf{x}) \geq 0. \end{cases} \tag{12}$$

This linear predictor can be made adaptive by the following rule: The initial choice of the weight vector $\mathbf{w}$ may have any value. The $\mathbf{w}$ is modified only when the prediction fails. That is, let the new weight vector $\mathbf{w}'$ be

$$\mathbf{w}' = \begin{cases} \mathbf{w} + \alpha\mathbf{x} \text{ if } s = 1 \text{ and } \hat{s} = 0; \\ \mathbf{w} - \alpha\mathbf{x} \text{ if } s = 0 \text{ and } \hat{s} = 1. \end{cases} \tag{13}$$

The choice $\alpha \geq 0$ for the coefficients determines the stability and convergence speed of the predictor. Several rules for choosing α are known [24, 25]. The simplest types are a) fixed increment rule, where α is any fixed positive number, and b) absolute correction rule, where α is the smallest integer greater than $|\langle \mathbf{w}, \mathbf{w} \rangle| / |\mathbf{x}|^2$.

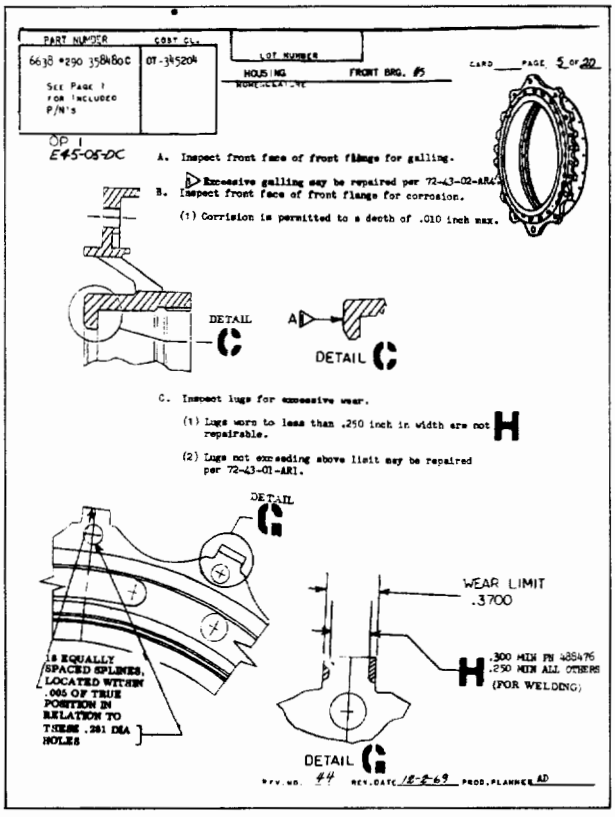
In its usual applications to pattern recognition, the adaptation of this discriminant function is done in a training period that precedes the test period. Although this operational mode is also applicable to the problem at hand, we can provide a more attractive mode in our ap-

the small computers need to be backed up, they can have a link to the large central computers.

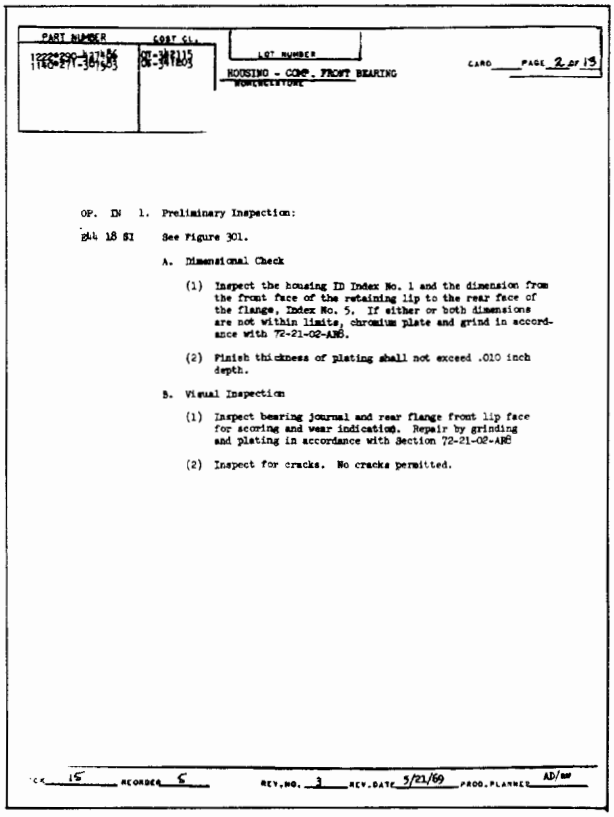
Since in many experiments the raw events can be converted easily into a more economical format, or frequently even rejected as useless after the simplest geometrical checks, another way of "stretching" the capacity of the on-line computers is to perform a maximum of simple logical operations on the data before they enter the computer. This reduces the amount of valuable memory taken up by buffers for useless information and leaves much more memory (and also time) for programs treating the useful data. The system designed for the experiment on line to the 360/44 is an attempt in this direction. Defining the center of a cluster of triggered wires in a wire spark chamber by special hardware is another.

ist, for although a huge effort went into each program, the result was usually a program extremely efficient in execution time and space usage. With the arrival of newer computers, with support of on-line applications even extending as far as time-sharing systems, the tendency has been to program in machine language only those operations which would be extremely inefficient in a higher level language, for example handling the input/output and buffering, and to make use of the flexibility of the system and the high level languages for most other tasks. However, in some applications it has been found that the high overheads imposed by these systems, often written for process control applications, have put such restriction on the data-taking capacity as to outweigh the advantages given by the flexibility.

(a)



(b)



(c)

Figure 6 Documents used for experimentation. (a) Portion of page from the *IBM Journal of Research and Development* (Vol. 13, No. 1, p. 110); (b) machine jobshop chart A; (c) machine jobshop chart B.

plication. That is, there is no need to set up a training period separate from the test period. We can operate this learning machine in an adaptive mode all the time without any interception or prescanning.

So far we have discussed only the linear learning machine because of its simplicity. However, essentially all types of learning machines and discriminant functions can be used as adaptive predictors. A Φ function [24] is defined by

$$\Phi(\mathbf{x}) = w_0 + w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) + \cdots + w_M f_M(\mathbf{x}), \quad (14)$$

where $f_i(\mathbf{x})$, $i = 1, 2, \cdots, M$, are any real functions of $\mathbf{x}$. For example, they can be chosen to extract important features of patterns. Another class of pattern classifiers is "layered machines" [24, 25]. A layered machine is a network of TLUs, organized in layers. This class includes the piecewise-linear discriminant function machines and the "α-perceptron."

Adaptive predictor with finite memory

In the fixed predictor discussed previously, an optimal prediction table must contain one bit of information for each state M_k of the memory set, $k = 1, 2, \cdots, 2^N$. The bit indicates whether the conditional probability $q(1/M_k)$ is greater than 0.5. But prior to setting up the decision table, one needs to estimate $q(1/M_k)$ by prescanning or some other means. Now, by assigning a few extra bits to each entry of this decision table, we can develop a practical scheme that does not need prior knowledge of data statistics. We associate with each state M_k a counter C_k. Let each counter have L bits ($L \geq 1$); therefore C_k can count from 0 to $2^L - 1$.

The prediction and adjusting algorithms proceed as follows. For a given memory set M_k, predict according to

$$\hat{s} = \begin{cases} 1 \text{ if } C_k \geq 2^{L-1} \\ 0 \text{ if } C_k < 2^{L-1}. \end{cases} \quad (15)$$

After the value of a pel is predicted, the actual value at that pel is used to update the up-down counter C_k as follows: Its new value is

$$C_k = \begin{cases} \min\{C_k + 1, 2^L - 1\} & \text{if } s = 1, \\ \max\{C_k - 1, 0\} & \text{if } s = 0. \end{cases} \quad (16)$$

Typical behavior of C_k is illustrated in Fig. 5 in which a two-bit counter ($L = 2$) is chosen. The memory set takes a state M_k at times $n_1, n_2, n_3, \cdots$, during which C_k moves either up or down according to the rule (16). In this illustrative example a transition takes place from one region to another in which statistical properties of the image patterns are different: In region A $q(1/M_k)$ appears to be larger than 0.5 and in region B, less than 0.5. Prediction errors occur at times n_1, n_2, n_8, and n_9. The behavior of C_k can be treated as a one-dimensional random walk with reflecting barriers at $C_k = 0$ and $C_k = 2^L - 1$ [26].

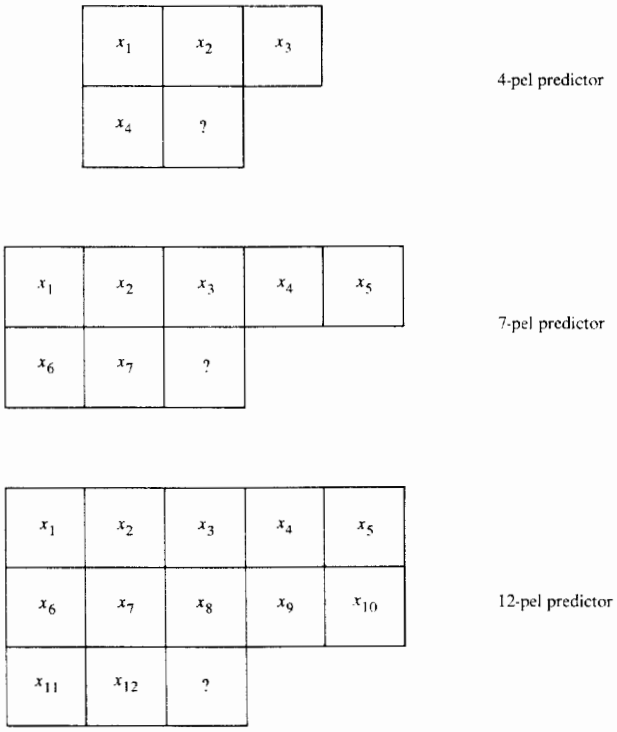


Figure 7 Predictors used in the experiments.

The appropriate choice of bit-size L for the counter C_k should be a compromise between two conflicting factors: If L is small, the value of C_k is susceptible to irregular or noisy data; on the other hand, if L is large, it takes a large number of data points to reach convergence and also to accomplish a transition from one region to another.

Many interesting variations of the update algorithm (16) are possible. For example, we could make the contents of all counters decay with time to the median value. Or, if the behavior of certain states is correlated, we could update more than one state each time, and so on. However, we have found through experiments that the value of reducing the prediction error rate by adopting such variations is negligible in comparison with the increased complexity of the algorithms.

Experimental results and conclusions

The techniques of prediction discussed in the previous section have been applied to data from three different sources: a page from the *IBM Journal of Research and Development* [Fig. 6(a)] and two line-drawings from a machine jobshop [Figs. 6(b) and 6(c)]. The size of the documents from which the data were obtained is $8\frac{1}{2}$ in. × 11 in. and the scanning resolution is 125 pels per inch, i.e., each pel is 8 mils (0.2 mm) square.

Figure 7 lists the memory sets of the predictors used in this experimental study. When fixed predictors are used,

...line computers is to perform a limited set of simple logical operations on the data before they enter the computer. This reduces the amount of valuable memo taken up by buffers for useless information and leav much more memory (and also time) for programs treatir the useful data. The system designed for the experime on-line to the 360/44 is an attempt in this directio

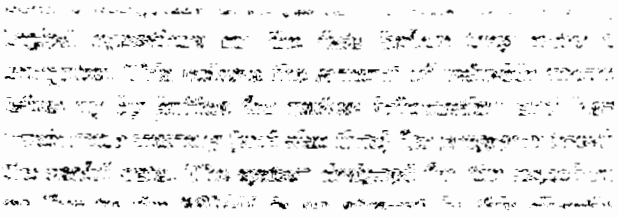


Figure 8 Source data $s(i, j)$ and prediction error $e(i, j)$ displays for the journal page experiment.

Table 1 Experimental results on journal page image (125 pels per inch).

Type of predictor	*Prediction error (percent)*
7-pel finite memory adaptive ($L = 3$)	4.79
7-pel fixed (prescanning)	5.14
4-pel finite-memory adaptive ($L = 3$)	5.16
4-pel fixed	5.85
12-pel adaptive linear ($\alpha = 1$)	6.14
4-pel adaptive linear ($\alpha = 1$)	7.22

Table 2 Experimental results on machine jobshop charts (125 pels per inch).

Type of predictor	*Prediction error (percent)*	
	Chart A	*Chart B*
7-pel fixed	2.24	1.25
4-pel finite-memory adaptive	2.30	1.20
4-pel fixed	2.37	1.35
12-pel adaptive linear ($\alpha = 1$)	2.50	1.52
4-pel adaptive linear ($\alpha = 1$)	2.79	1.67

20% or more of the entire page is prescanned to derive an optimal predictor rule. Adaptive linear predictors are applied with the initial setting $w_i = 0$ for $i = 0, 1, \cdots, N$ and no prescanning. Finite-memory adaptive predictors start with every counter set to half its capacity, i.e., $C_k = 2^{L-1}$, $k = 1, 2, \cdots, 2^N$.

Figure 8 shows a portion of a scanned image $s(i, j)$ and the corresponding prediction error pattern $e(i, j)$ when a 4-pel adaptive linear predictor is used. The prediction error patterns obtained for the other types of predictors look quite similar to this example. Table 1 summarizes the performance of the three predictors applied to the journal page image and it clearly shows that the finite-memory adaptive predictors excel for this type of image data.

Figure 9 shows plots of the prediction error rate of 4- and 7-pel adaptive predictors with finite memory as the counter size L is changed. The 7-pel adaptive predictor with $L = 3$ yields the lowest prediction rate, 4.79 percent.

An adaptive linear predictor, on the other hand, does not yield high performance. This seems to indicate that the linear constraint on a predictor is too restrictive. A substantial performance improvement is achieved by enlarging the memory set from $N = 4$ to $N = 12$, but the performance is still not as good as that of the other types of predictors. Therefore, if one wants to build a learning-machine type of predictor, he should probably use piecewise-linear or polynomial predictors.

Table 2 summarizes the results obtained on the machine jobshop charts A and B. We observe again the superiority of the finite-memory adaptive predictor over the other types. The results from chart B show that the 4-pel finite memory adaptive predictor surpasses even the 7-pel fixed predictor.

Acknowledgments

We express our thanks to L. S. Loh for his programming assistance and to D. J. Min, E. V. Eiseln, K. Myer and F. Wood for providing us with the scanned data used in our simulation study. Some of the work reported in this paper was done in collaboration with our colleagues, D. I. Barnea and D. D. Grossman. We are also indebted to R. B. Arps for many stimulating discussions on the subject. Finally we express our appreciation to P. E. Green, J. Raviv and M. G. Smith for the encouragement they gave us during the course of the work.

References

1. R. G. Gallager, *Information Theory and Reliable Communication*, John Wiley & Sons, Inc., New York, 1968.
2. L. R. Bahl and H. Kobayashi, "Image Data Compression by Predictive Coding II: Encoding Algorithms," *IBM J. Res. Develop.*, **18**, 172 (1974), following paper.
3. H. R. Schindler, "Delta Modulation," *IEEE Spectrum* **7**, 69 (October 1970).

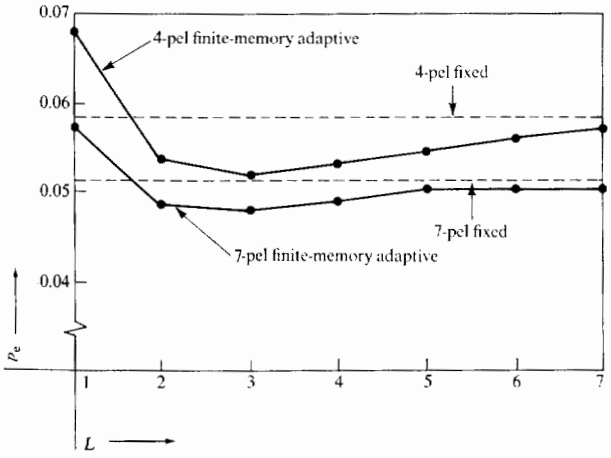


Figure 9 Prediction error rate p_e vs counter size L: experimental results from the journal page data.

4. P. Elias, "Predictive Coding, Parts I and II," *IRE Trans. Information Theory* **IT-1,** 16 (March 1955).
5. J. S. Wholey, "The Coding of Pictorial Data," *IRE Trans. Information Theory* **IT-7,** 99 (1961).
6. R. B. Arps, "Entropy of Printed Matter at the Threshold of Legibility for Efficient Coding in Digital Image Processing," Report No. 31, Stanford Electronics Laboratory, California, 1969.
7. A. V. Balakrishnan, "An Adaptive Non-linear Data Predictor," *Proc. Nat. Telemetry Conf.*, Washington, D.C., May 23-25, 1962, Paper No. 6-5.
8. L. D. Davisson, "The Theoretical Analysis of Data Compression Systems," *Proc. IEEE* **56,** 176 (1968).
9. R. L. Kutz and J. A. Sciulli, "The Performance of an Adaptive Image Data Compression System in the Presence of Noise," *IEEE Trans. Information Theory* **IT-14,** 273 (1968).
10. W. K. Pratt, J. Kane and H. C. Andrews, "Hadamard Transform Image Coding," *Proc. IEEE* **57,** 58 (1969).
11. H. C. Andrews and W. K. Pratt, "Transform Image Coding," in *Proceedings of the Symposium on Computer Processing in Communications*, Polytechnic Press of the Polytechnic Institute of Brooklyn, New York, 1970, pp. 63-84.
12. H. C. Andrews and K. L. Caspari, "A Generalized Technique for Spectral Analysis," *IEEE Trans. Computers* **C-19,** 16 (1970).
13. I. J. Good, "The Interaction Algorithm and Practical Fourier Analysis," *J. Roy. Stat. Soc.*, Series B, **20,** 361 (1958).
14. J. W. Cooley and J. W. Tukey, "An Algorithm for the Machine Calculation of Complex Fourier Series," *Math. Computation* **19,** 297 (1965).
15. J. E. Whetchel and D. F. Guinn, "The Fast Fourier-Hadamard Transform and Its Use in Signal Representation and Classification," *EASCON Record*, 1968, pp. 561-573.
16. T. S. Huang and O. J. Tretiak (eds.), *Picture Bandwidth Compression*, Gordon and Breach Science Publishers, Inc., New York, 1972.
17. H. C. Andrews, *Computer Techniques in Image Processing*, Academic Press, Inc., New York, 1970.
18. A. Rosenfeld, *Picture Processing by Computer*, Academic Press, Inc., New York, 1967.
19. Special issue on "Signal Processing for Digital Communications," *IEEE Trans. Communications* **COM-19,** No. 6, 1971.
20. H. Kobayashi and L. R. Bahl, "Image Compaction by Predictive Coding: Fundamentals," Research Report RC 3249, IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598, February 1971.
21. M. Karnaugh, "The Map Method for Synthesis of Combinatorial Logic Circuits," *Trans. AIEE* **72,** Part I, 593 (1953).
22. E. J. McCluskey, *Introduction to the Theory of Switching Circuits*, McGraw-Hill Book Co., Inc., New York, 1965.
23. H. Kobayashi, "Adaptive Data Compression System," *IBM Tech. Disclosure Bull.* **14,** 1305 (1971).
24. N. J. Nilsson, *Learning Machines*, McGraw-Hill Book Co., Inc., New York, 1965.
25. K. S. Fu, "Learning Control Systems—Review and Outlook," *IEEE Trans. Automatic Control* **AC-15,** 210 (1970).
26. W. Feller, *An Introduction to Probability Theory and Its Applications*, Vol. I, 3rd ed., John Wiley & Sons, Inc., New York, 1968.

Received April 25, 1973; revised July 30, 1973.

The authors are located at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598.