ONE-PASS ADAPTIVE UNIVERSAL VECTOR QUANTIZATION *

M. Effros[1] *P. A. Chou*[2] *R. M. Gray*[1]

[1]Information Systems Laboratory, Stanford University, Stanford, California 94305-4055, USA
[2]Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, California 94304, USA

ABSTRACT

We introduce a one-pass adaptive universal quantization technique for real, bounded alphabet, stationary sources. The algorithm is set on line without any prior knowledge of the statistics of the sources which it might encounter and asymptotically achieves ideal performance on all sources that it sees. The system consists of an encoder and a decoder. At increasing intervals, the encoder refines its codebook using knowledge about incoming data symbols. This codebook is then described to the decoder in the form of updates on the previous codebook. The accuracy to which the codebook is described increases as the number of symbols seen, and thus the accuracy to which the codebook is known, grows.

1. INTRODUCTION

Universal data compression is data compression that asymptotically achieves the distortion-rate bound on all sources (within a class of sources). By observing more and more data, universal compressors essentially learn the statistics of the source in operation. This must be done so that optimal performance is eventually achieved.

There exist two basic approaches to universal noiseless coding and universal quantization. The first approach is alternately known as two-pass adaptive, off-line, or batch universal coding, while the second approach is termed one-pass adaptive, on-line, or incremental universal coding. For the sake of simplicity and intuition, we will henceforth refer to members of the first category as non-adaptive codes and members of the second category as adaptive codes.

All universal codes must learn the statistics of the source in operation if they are to achieve optimal performance on that source. What then is the difference between an adaptive universal code and a non-adaptive universal code? While formal definitions for the two types of codes will be postponed until Section 2, roughly speaking, adaptive codes are codes that *incrementally* update their data coding strategies as knowledge about a particular data stream is acquired. In contrast, non-adaptive codes first garner all of the information they will use to encode the data sequence under consideration and then use that information to encode the entire sequence.

Universal noiseless coding and quantization schemes that are non-adaptive have been studied extensively, based on the foundational work of Davisson[1], Ziv[2], and others. Universal noiseless coding schemes that are adaptive, such as the Ziv-Lempel algorithm[3] and Rissanen's context algorithm[4], are also well understood. However, attempts to generalize these adaptive noiseless coding algorithms in a provably universal fashion to the case of quantization have until recently been unsuccessful. In [5], Zhang and Wei present an adaptive universal quantization technique for independent, identically distributed (iid) symbols from finite, discrete alphabet sources. Here we introduce an adaptive universal quantization technique for stationary, real-valued sources with respect to the squared error fidelity criterion and, in the iid case, explore the convergence rates for the given algorithm.

*THIS MATERIAL IS BASED UPON WORK PARTIALLY SUPPORTED BY THE NATIONAL SCIENCE FOUNDATION UNDER AN NSF GRADUATE FELLOWSHIP, BY A GRANT FROM THE CENTER FOR TELECOMMUNICATIONS AT STANFORD, AND BY AN AT&T PH.D. SCHOLARSHIP.

2. ADAPTIVE CODING

In any code, the reproduction $\hat{x}_k$ of any symbol x_k in a data sequence $x_1, \ldots, x_n$ depends functionally on a subset of the past data, say $\{x_{k-j} : 0 \leq j \leq m\}$, and on a subset of the future data, say $\{x_{k+j} : 0 \leq j \leq a\}$. The minimum such m and a are called the *memory* $\mathcal{M}(k,n)$ and *anticipation* $\mathcal{A}(k,n)$ of the code at time k for data length n. Thus $\mathcal{W}(k,n) = \mathcal{M}(k,n) + 1 + \mathcal{A}(k,n)$ is the length of the smallest window of data on which $\hat{x}_k$ depends. We call a code *adaptive* if for each k, this window length (as a function of n) is bounded, i.e.,

$$\sup_{n \in \mathbb{N}} \mathcal{W}(k,n) < \infty \qquad \text{for all } k \in \mathbb{N},$$

or equivalently if the anticipation (as a function of n) is bounded. Intuitively, an adaptive code can reproduce the kth symbol by looking ahead by a bounded amount, regardless of the length of the sequence in which that symbol occurs. Notice that according to this definition, any fixed dimension memoryless vector quantizer (VQ) is an adaptive code since the anticipation at any time k is at most the vector dimension. The formal definition given here for adaptive codes becomes interesting really only in the case of adaptive universal codes, where the bounded anticipation constraint is joined by the requirement that the asymptotic performance approach optimality.

To understand the basis for the chosen definition of adaptive codes, consider the following universal quantization scheme posed by Ziv in [2]. Given a length n sequence of data, the encoder reads in the entire sequence and designs the optimal ℓ-dimensional codebook for that sequence, where the dimension ℓ is chosen as a function of n. The encoder then describes that quantizer to the decoder (at a rate which is also a function n) and uses that codebook to describe the data. This algorithm is universal, but intuitively it is not adaptive. Although the algorithm can deal

0-7803-1775-0/94 $3.00 © 1994 IEEE

with a sequence of any length n, the description of a sequence of length $n+1$ will require that the entire algorithm be re-initiated. Consider, then, the formal definition of an adaptive code. The encoding of any symbol x_k in a sequence $x_1, \ldots, x_n$ requires knowledge of the entire sequence. Thus, in this case, the window length required to reproduce the kth symbol of a sequence of length n is $\mathcal{W}(k,n) = n$ for all $k \in \{1, \ldots, n\}$. If the data sequence in which the kth element occurs is now allowed to grow, the window length required will likewise grow indefinitely. In order to encode even the first symbol we are required to process the entire sequence. In this case, $\sup_{n \in \mathbb{N}} \mathcal{W}(k,n) = \infty$ for all k and the algorithm is classified as non-adaptive.

In contrast to the above example, consider the Ziv-Lempel algorithm [3] as an example of a universal noiseless code which meets our intuitive definition of adaptivity. The reproduction of the kth symbol in a sequence of length n depends on the string of symbols into which the symbol is parsed and also on the table of previously occurring strings. Thus the reproduction of x_k depends on the entire data up to time k and also on some amount of data into the future. Specifically, $\mathcal{M}(k,n) = k-1$, and $\mathcal{A}(k,n) \leq \sqrt{2k}$, since the maximum length of the string into which the kth symbol can parsed is bounded above by $\sqrt{2k}+1$, as a simple argument shows. Thus $\sup_{n \in \mathbb{N}} \mathcal{W}(k,n) \leq k + \sqrt{2k}$ for every k. We thereby confirm that the Ziv-Lempel code is indeed an adaptive universal algorithm, and that our intuition and definition again match.

3. THE ALGORITHM

We next describe an algorithm for adaptive universal quantization of bounded alphabet sources. We assume that the bound is known to both encoder and decoder. Without loss of generality, we here set the region of support equal to $[0, B]$ for some $B \in \mathbb{R}$. This paper considers fixed-rate nearest neighbor encoding schemes. Variable-rate schemes will be considered in a later work.

In simple terms, the data compression process proceeds as follows. Both the encoder and decoder start with a default, fixed-rate codebook of a set dimension. As incoming data is received, that data is broken into vectors and used to train a codebook. After a pre-specified interval, the encoder describes the codebook to the decoder and then uses that codebook to describe the data. The encoder then reads in another collection of data vectors, uses that collection to update the codebook, describes the changes in the codebook to the decoder, and sends the latest increment of data to the decoder using the updated codebook.

The length of the intervals starts small and grows. As more and more data is processed, the accuracy to which the codebook is known increases. In fact, Pollard's k-means central limit theorem [6] shows that the difference between the empirically optimal codewords and the optimal codewords, when multiplied by $\sqrt{m}$, where m is the length of the training data in vectors, converges to a zero-mean Gaussian distribution. We therefore know the optimal codewords to a precision proportional to $1/\sqrt{m}$, and we need to describe the codewords to no more than this precision. This implies that after seeing $m = 2^{2j}$ vectors, we can describe the codebook with essentially one additional bit per component per codeword more than the description of the codebook after seeing $m = 2^{2(j-1)}$ vectors. This progressive codebook transmission scheme naturally partitions the data into increasing intervals of length $2^{2j} - 2^{2(j-1)}$ times the vector dimension, for $j = 1, 2, \ldots$.

More specifically, the algorithm is summarized by the following iterative process. The encoder and decoder are initialized with the same arbitrary codebook and the index j is set to 1.

1. Read in more vectors until the total number of vectors seen so far is 2^{2j}.
2. Update the current codebook to find the best codebook for the entire collection of symbols seen thus far.
3. Quantize the components of each codeword to bins with width proportional to $B/2^j$.
4. Use a conditional entropy code to describe the quantization bins for the new codebook given the quantization bins for the current codebook.
5. Encode the recently read data with the new codebook.
6. Increment j by 1 and iterate.

The conditional entropy code is used to describe the quantization bins for the new codebook incrementally, given the quantization bins for the current codebook, rather than describing the new codebook from scratch each time. If a component of a current codeword lies in a bin of width $B/2^{j-1}$, then with high probability, according to Pollard's asymptotic distribution, the corresponding new codeword component will lie in one of the two refining bins of width $B/2^j$. With low probability, the new codeword component will lie in some other bin. Thus the average rate of the entropy code is slightly greater than one bit.

The algorithm is adaptive, since it has memory $\mathcal{M}(k,n)$ equal to k times the vector dimension and anticipation $\mathcal{A}(k,n)$ at most $2^{2j} - 2^{2(j-1)}$ times the vector dimension, where $2^{2(j-1)} < k \leq 2^{2j}$, for all n. In principle the dimension of the current vector must also grow slowly with k, so that the performance of the algorithm can asymptotically achieve the distortion-rate bound. We have so far implemented only a fixed vector dimension for all k, since the optimal rate of growth is so slow that the dimension does not change over the range of data lengths we consider in our experiments. However, in our analysis of the algorithm (in the next section) we assume that vector dimension does indeed grow, and that each time the dimension increases, the entire codebook is retransmitted.

The worst-case anticipation of the system can be reduced from $2^{2j} - 2^{2(j-1)}$ times the vector dimension to just the vector dimension, by training the new codebook only on data that has already been transmitted, rather than by waiting to include data that has not yet been transmitted. This variation, which we shall call *low-delay*, of course incurs some performance penalty. This penalty is asymptotically negligible, as we will see in the next section.

4. UNIVERSALITY AND RATES OF CONVERGENCE

Let $\{X_i\}$ be independent symbols from a real-valued stationary random process with process measure P_θ, $\theta \in \Lambda$, and let β be an arbitrary ℓ dimensional vector quantizer. Define the per letter distortion and rate redundancies for β as $\Delta_D(\beta,\theta) = \frac{1}{\ell} E_\theta d(X^\ell, \beta(X^\ell)) - D_\theta(R)$ and $\Delta_R(\beta,\theta) = \frac{1}{\ell} E_\theta |\beta(X^\ell)| - R$ respectively, where E_θ is the expectation with respect to P_θ, $D_\theta(R)$ is the corresponding Shannon distortion-rate function, and $|\beta(X^\ell)|$ denotes the length of the code for the vector X^ℓ. Let $D_{\theta,\ell}(R)$ be the ℓth order operational distortion-rate function, which is the minimal

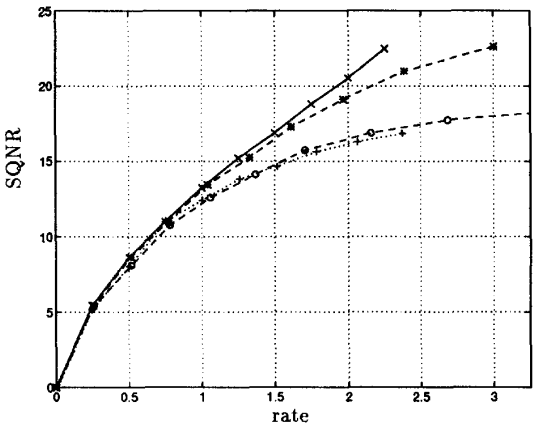

Figure 1. Adaptive and non-adaptive universal coding on a single medical brain scan are compared to standard fixed-rate VQ trained and tested on the same image: — standard VQ, -*- adaptive universal VQ, -o- low-delay adaptive universal VQ, ··· non-adaptive universal VQ.

expected distortion achievable by a rate R fixed-rate ℓ dimensional quantizer on P_θ. We next consider the universality of the algorithm and the rates at which the distortion and rate redundancies decay to zero.

Three terms contribute to the distortion redundancy Δ_D. Quantizing the codebook in Step 3 of the algorithm above contributes one term. The second term results from using a codebook matched to the sample distribution of m vectors rather than the true underlying distribution. For a codebook dimension ℓ, the discrepancy between $D_{\theta,\ell}(R)$ and $D(R)$ represents the final contribution. The rate redundancy of a universal code results from the need to describe the codebook to the decoder. In [7], Linder et al. analyze these factors for the Ziv universal quantizer described in Section 2 and show that with appropriate choice of the codeword dimension and the rate at which the codebook is described, the sum of the two redundancy terms for the non-adaptive universal quantizer can be made to approach zero as $O\left(\sqrt{\frac{\log\log n}{\log n}}\right)$, where n is once again the total number of symbols to be encoded. Using a parallel analysis, the total redundancy for the adaptive universal quantizer has been shown in [8] to achieve the same $O\left(\sqrt{\frac{\log\log n}{\log n}}\right)$ weighted redundancy, whether or not the adaptive universal quantizer is low-delay, though with somewhat larger constants.

5. EXPERIMENTAL RESULTS

The adaptive universal VQ described above was tested on images from a database of magnetic resonance imaging (MRI) brain scans. The image size was 256×256 and a constant vector dimension of four was used in all experiments. A random permutation was used to scramble the sequence of vectors, so that the resulting sequence was approximately independent. Figure 1 compares the performances of the adaptive universal VQ and its low-delay variation on a single medical image. Also included are the results of Ziv's non-adaptive universal VQ [2] with the parameters specified in [7]. The performance of these universal schemes, which all employ fixed-rate, full-search VQ, is compared to

Figure 2. Standard fixed-rate VQ of a single USC database image using a codebook trained on a sequence of medical brain scans. SQNR = 10.1, rate = .75 bpp

the performance of a standard, fixed-rate, full-search VQ trained specifically for that image. While the performance of the non-adaptive universal code here lags behind that of the adaptive universal codes, this gap could almost certainly be closed by proper choice of the constants accompanying the parameters specified in [7].

If a codebook is designed for one type of source and operated on another, the optimal performance of the source in operation is not achieved. Figures 2, 3, and 4 demonstrate the results of codebook/data mismatch. In Figure 2 a USC database image is encoded with a standard VQ trained for medical brain scans. The resulting performance is clearly inferior to the performance of a standard VQ trained specifically for that image, as shown in Figure 3. Figure 4 shows the same image coded by the adaptive universal quantization algorithm. All three experiments were conducted with fixed-rate codebooks containing eight codewords per codebook. The rate in the adaptive case is slightly higher due to the overhead associated with describing the codebook.

6. CONCLUSIONS

The algorithm here presented demonstrates the advantages of adaptive universal compression compared to non-universal or non-adaptive alternatives. In the long term, a universal code is guaranteed to perform better than a non-universal code trained on a mismatched source and virtually as well as a non-universal code trained on the source in operation. While a universal code's optimal performance is only guaranteed asymptotically in the length of the data to be coded, the experiments here presented demonstrate good performance even on small data sets. In particular, the performance of the adaptive universal scheme on a single image is comparable to the performance of a non-universal code which is allowed prior access to the source statistics and is thus perfectly matched to the source in operation.

The benefits of adaptive universal codes over non-adaptive universal codes include shorter delay and lower

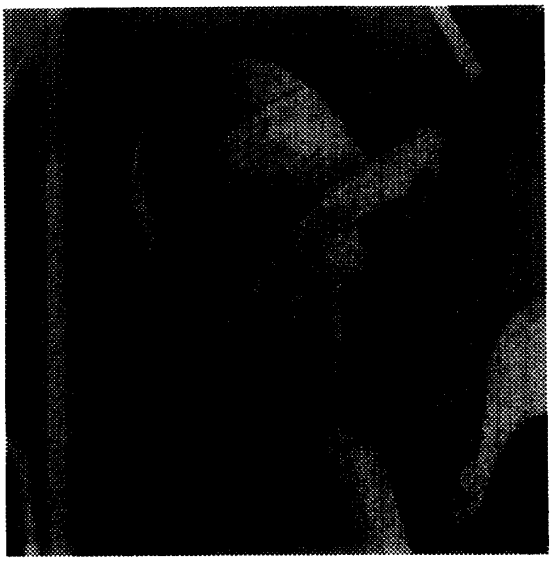

Figure 3. Standard fixed-rate VQ of a single USC database image using a codebook trained on that image. SQNR = 14.5, rate = .75 bpp

Figure 4. Adaptive universal VQ using a fixed-rate codebook. SQNR = 14.4, rate = .755 bpp

storage requirements. The delay associated with an adaptive code is equal to the anticipation, which is bounded for any particular symbol of an adaptive universal code but unbounded in non-adaptive universal codes. In low-delay adaptive universal VQ, the anticipation is reduced to the vector dimension while maintaining performance quality comparable to that of the non-adaptive universal quantizer. Such reductions in delay also translate to reductions in storage requirements at the encoder.

Further reductions in encoder storage can be gained by replacing the complete codebook update (here implemented using the Lloyd algorithm) at each iteration by incremental codebook update techniques such as the stochastic gradient algorithm of [9]. While implementation of the Lloyd algorithm at each iteration requires that all previously received symbols be stored for use in training future codes, the stochastic gradient technique attempts to track the incremental codebook modification associated with each incoming data vector and therefore requires storage of at most one vector of previously received information. Unfortunately, application of the stochastic gradient technique to adaptive universal VQ yields significant performance degradation and the results are therefore not included in Section 5.

The adaptive universal VQ presented here is based on fixed-rate standard VQ and achieves performance comparable to the best fixed-rate standard VQ even on small data sets. A goal of future work will be to apply the adaptive techniques developed here to more sophisticated VQ frameworks.

REFERENCES

[1] L. D. Davisson. Universal noiseless coding. *IEEE Transactions on Information Theory*, 19(6):783–795, November 1973.

[2] J. Ziv. Coding of sources with unknown statistics—Part II: Distortion relative to a fidelity criterion. *IEEE Transactions on Information Theory*, 18(3):389–394, May 1972.

[3] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *IEEE Transactions on Information Theory*, 24(5):530–536, September 1978.

[4] J. Rissanen. Stochastic complexity and modeling. *Annals of Statistics*, 14:1080–1100, September 1986.

[5] Z. Zhang and V. K. Wei. An on-line universal lossy data compression algorithm via continuous codebook refinement. May 1993. preprint.

[6] D. Pollard. *Convergence of Stochastic Processes*. Springer-Verlag, New York, 1984.

[7] T. Linder, G. Lugosi, and K. Zeger. Rates of covergence in the source coding theorem, in empirical quantizer design, and in universal lossy source coding. *IEEE Transactions on Information Theory*, 1993. To appear.

[8] M. Effros, P. A. Chou, and R. M. Gray. Rates of convergence in adaptive universal vector quantization. In *Proceedings of the IEEE International Symposium on Information Theory*, Trondheim, Norway, June 1994. Submitted.

[9] P.-C. Chang and R. M. Gray. Gradient algorithms for designing predictive vector quantizers. *IEEE Transactions on Acoustics Speech and Signal Processing*, ASSP-34(4):679–690, August 1986.