An Iterative Joint Codebook and Classifier Improvement Algorithm for Finite-State Vector Quantization

Keren O. Perlmutter Sharon M. Perlmutter Michelle Effros† Robert M. Gray

Information Systems Laboratory
Stanford University
Stanford, CA 94305

†Department of Electrical Engineering (116-81)
California Institute of Technology
Pasadena, CA 91125

Abstract

A finite-state vector quantizer (FSVQ) is a multi-codebook system in which the current state (or codebook) is chosen as a function of the previously quantized vectors. We here introduce a novel iterative algorithm for joint codebook and next state function design of full search finite-state vector quantizers. We consider the fixed-rate case, for which no optimal design strategy is currently known. A locally optimal set of codebooks is designed for the training data and then predecessors to the training vectors associated with each codebook are appropriately labelled and used in designing the classifier. The algorithm iterates between next state function and state codebook design until it arrives at a suitable solution. The proposed design consistently yields better performance than the traditional FSVQ design method (under identical state space and codebook constraints).

1 Introduction

The main advantage of vector quantization over scalar quantization is that a vector quantizer (VQ) exploits the statistical redundancy between pixels within a block to reduce the bit rate. The performance of a VQ can be improved by increasing the vector dimension. However, the computational complexity of the VQ increases exponentially with the block size. Incorporating memory into the VQ can improve the performance of the VQ without greatly increasing its complexity since such a system can exploit interblock correlation while a moderate block size is retained. A finite-state vector quantizer (FSVQ) [4, 1, 6, 5] is an example of a VQ with memory. The FSVQ consists of a finite state space, an initial state, an encoder, a decoder, and a next state function. The encoder and decoder contain identical copies of a finite set of subcodebooks, where each subcodebook belongs to a distinct state. The next state function applies a classifier to previously encoded vectors to determine the state of the encoder/decoder pair, thereby designating the subcodebook to be used when encoding the current input vector. Thus, given an arbitrary initial state known to both encoder and decoder, the decoder can exactly track the state sequence without using side information. Hence FSVQ allows the use of more codewords at a particular bit rate without the need to transmit additional bits (side information) to the decoder.

Whereas optimal FSVQ design in the variable rate case was accomplished by Chou and Lookabough in [3], the problem of optimal design remains unsolved for fixed-rate codes. We here propose a new iterative technique for FSVQ design. Although this new technique does not guarantee an optimal solution, it consistently yields higher performance than the traditional FSVQ design method (under identical state space and codebook constraints) on all data sets considered by the authors.

2 Algorithm

The proposed algorithm is modelled after the omniscient FSVQ design [5], a technique introduced by Foster et al. [4], developed for image coding by Aravind and Gersho [1], and extended to the variable rate case by Hussain and Farvardin [6]. We first briefly describe the fixed-rate full search omniscient FSVQ design, as outlined in [6] (and which we will hereafter refer to as the traditional FSVQ design), and then present our proposed algorithm.

In the traditional design to image compression using FSVQ [6], the initial classifier is a K-codeword VQ, $C = \{c(v), v\epsilon\{1, 2, ..., K\}\}$, that is designed by applying the generalized Lloyd algorithm (GLA) to

1058-6393/95 $4.00

the training sequence $\{X_i, i = 0, 1, ...\}$. For each state index $m\epsilon\{1, ..., J\}$, where $J = K \times K$, a codebook is constructed using the GLA on the training subsequence $\{X_i : k_1 = \text{argmin}_{v\epsilon\{1,2,...,K\}}\ d(X_{L_i}, c(v))$ and $k_2 = \text{argmin}_{v\epsilon\{1,2,...K\}}\ d(X_{A_i}, c(v))\}$, where (k_1, k_2) is the pair associated with state $m \epsilon \{1, ..., J\}$, i.e., $m = k_1 + K * k_2$, X_{L_i} and X_{A_i} refer to the vectors to the left and above the current input vector, respectively, and $d(\cdot, \cdot)$ is the particular distortion measure used. The next state function f is then defined by $f = (f_1, f_2)$, where $f_1 = \text{argmin}_{v\epsilon\{1,2,...,K\}}\ d(\hat{X}_{L_i}, c(v))$ and $f_2 = \text{argmin}_{v\epsilon\{1,2,...K\}}\ d(\hat{X}_{A_i}, c(v))$, where $\hat{X}_{L_i}$ and $\hat{X}_{A_i}$ represent the reconstructed X_{L_i} and X_{A_i} vectors, respectively. The use of reconstructed vectors in the next state function engenders a closed loop design that enables the decoder to track the state sequence without the use of side information. The state codebooks can then be improved by encoding the training sequence and updating each codeword by replacing it with the centroid of the cell associated with that codeword. The classifier can also be updated in a similar fashion. These update steps can be iterated until specified conditions have been met. We note that this iterative algorithm does not necessarily guarantee descent.

Although this method results in improved performance over full search VQ [6, 1, 5], once the initial classifier is obtained and the codebooks are optimized for it, the classifier is redesigned using only a limited amount of the information available (namely, the reconstructed training sequence) from the newly designed state codebooks. We propose to improve the performance of the FSVQ by using an iterative algorithm that jointly designs the state codebook and the next state function. In particular, we first design a locally optimal set of state codebooks for the training data and then use both the information about which codebook is best for a particular vector in the training sequence and the reconstructed training sequence to construct the classifier.

Given an initial reproduction codebook C_s for each state $s \epsilon \{1, ..., J\}$, the design algorithm is described as follows:

1. Let X_i be the current input vector and $\hat{X}_{C_i}$ be the concatenation of the vectors $\hat{X}_{L_i}$ and $\hat{X}_{A_i}$ (for the first iteration we let $\hat{X}_{L_i} = X_{L_i}$ and $\hat{X}_{A_i} = X_{A_i}$ since we have not yet obtained reconstructed vectors). Also let y_n^k denote the n^{th} codeword in the k^{th} state codebook. Designate $\hat{X}_{C_i}$ as a member of class k if $\min_{n\epsilon k}\ d(X_i, y_n^k) \leq \min_{n\epsilon m}\ d(X_i, y_n^m)$, $\forall\ m \epsilon \{1, ..., J\}$, i.e. k is the state codebook which contains the minimum distortion codeword with vector X_i. For each input vector X_i of the training sequence we now have its corresponding concatenated vector $\hat{X}_{C_i}$ and the concatenated vector's class, S_{c_i}.
2. Construct a classifier $C = \{c(v), v\epsilon\{1, 2, ..., J\}\}$ using the $(\hat{X}_{C_i}, S_{C_i})$ pairs.
3. Classify the $\hat{X}_{C_i}$ using the classifier in step 2.
4. For each state s of the FSVQ, design a reproduction codebook using full search VQ on the training subsequence composed of all the vectors X_i whose corresponding reconstructed concatenated vectors, $\hat{X}_{C_i}$, were classified as class s by the classifier C.
5. Iterate steps 1-4 until convergence or other suitable solution.

The above algorithm jointly designs the classifier and the state codebooks by incorporating information about each of these into the design of the other. In particular, for each iteration of the proposed algorithm, we begin with state codebooks and determine which state codebook would best represent the associated vector X_i of a predictor $\hat{X}_{C_i}$. This information is then incorporated into the design of the classifier. In addition, the state codebooks are designed using the most recently updated classifier. In contrast, in the traditional design the first iteration begins with the design of the classifier, and thereafter, the state codebooks are designed using this classifier. On subsequent iterations, the classifier and state codebooks are refined independently.

The classifier of step 2 will consist of at least one codeword for each state k, where the codewords are a function of the subsequence $\{\ \hat{X}_{C_i} : \min_{n\epsilon k}\ \mathrm{d}(X_i, y_n^k) \leq \min_{n\epsilon m}\ \mathrm{d}(X_i, y_n^m)\ \}\ \forall\ m \epsilon \{1, ..., J\}$. There are numerous ways to construct this classifier. For example, a simple approach would be to take the centroid of all predictors belonging to a particular class. Another method would be to design a classifier using a classification algorithm such as the classification and regression tree algorithm (CART^{TM}) of [2], where various features of the predictors would be used in the classifier design. The use of a concatenated vector, $\hat{X}_{C_i}$, as the predictor for the state of the current vector is different from the traditional method of separately classifying the vectors above and to the left to form a predictor.

Although the algorithm is not guaranteed to converge (it is not a descent algorithm), experimental results show that the iterations generally improve performance. We note that this algorithm is intuitive and simple, and that it has the same encoding complexity as the traditional FSVQ approach.

3 Results

The performance of our algorithm was investigated using 12 bpp computerized tomographic (CT) lung images. Figure 1 illustrates a typical CT image. Five 512 $\times$ 512 training images were used. The algorithm was tested on two images outside of the training set and the results were then averaged. For bit rates less than 0.5 bpp, we used 16-dimensional vectors and for bit rates of 0.5 bpp and greater, we used 4-dimensional vectors. The smaller dimension was selected for the higher rates in order to maintain an adequate ratio of training vectors to codewords. Nine states were considered for all cases (this number was based on design complexity and training sequence size considerations as well). SNR was the measure used to evaluate our algorithm, where $\text{SNR} = -10\log_{10}(\text{distortion}/D_0)$ and D_0 is the distortion on the test sequence using the best zero rate code. The distortion measure used was mean squared error.

As a preliminary investigation of this algorithm, we chose a basic classifier as the next state function. In particular, a centroid was computed for each class using all the concatenated vectors that mapped into that particular class according to step 1. These class centroids were then the codewords of the classifier. The algorithm was halted by the following procedure. Both a distortion difference threshold and a maximum number for iterations were specified. Then, the codebooks selected were those generated on the iteration that produced the least distortion on the training sequence.

Figures 2(a) and 2(b) compare the results of our algorithm with the results obtained with ordinary (memoryless) full search VQ and the fixed-rate full search FSVQ of [6]. We also illustrate the performance achievable if the codebooks are used in collective fashion as a universal code known to an omniscient decoder; that is, the encoder is allowed to select the best word in any codebook and the decoder knows which codebook was selected without counting this side information in the total bit rate. This provides an unbeatable bound to performance for a given collection of codebooks. We will refer to this system as the omniscient system. Figure 2(a) illustrates the performance of the algorithms at rates between 0.1875 and 0.375 bpp. The proposed design outperforms ordinary full search VQ by as much as 3 dB at 0.1875 bpp. The figure illustrates that the former design provided up to 0.6 dB in SNR improvement compared to that of the traditional FSVQ design. In addition, although the algorithm does not explicitly seek to minimize a perceptual distortion measure, we observed that the visual quality of the proposed design was better than that obtained with the traditional FSVQ, even when there was only a slight difference in SNR between the two methods. Figure 3 illustrates the reconstructed CT images obtained with these two FSVQ schemes at 0.375 bpp. Although the SNR difference between the two images is less than 0.1 dB (the two FSVQ algorithms yielded the least difference in SNR at this rate), the image compressed with the proposed design appears to have better quality. These observations illustrate the known fact that SNR is not necessarily a good distortion measure for perceptual quality. Figure 2(a) also illustrates that at rates above 0.3 bpp, the amount of improvement possible with the omniscient system is less than 1 dB over the traditional scheme, and thus we do not expect the joint algorithm to significantly outperform the traditional FSVQ approach at these rates. We suspect that the inability to obtain much higher SNR at these rates is due to an inadequate ratio of the number of 16-dimensional training vectors to the number of codewords in the multi-codebook system.

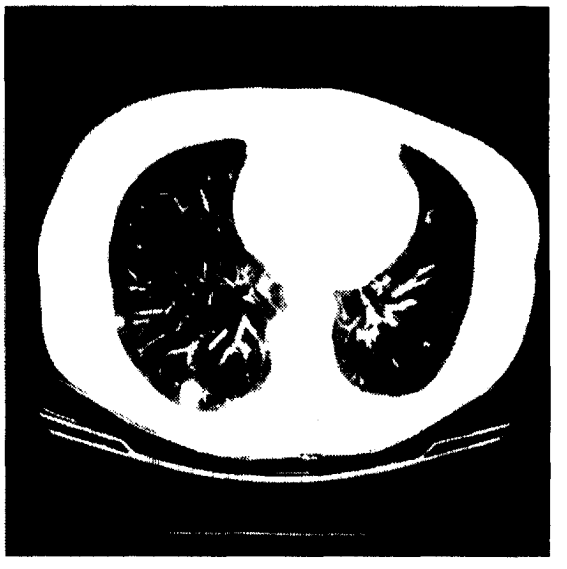

Figure 1: Original CT image

Figure 2(b) illustrates the performance of the four methods at rates between 0.5 and 1.25 bpp. The proposed FSVQ design significantly outperforms the ordinary full search VQ design, with up to 5.2 dB improvement at 0.75 bpp. We also observe that the proposed FSVQ design outperforms traditional FSVQ by as much as 2.3 dB at that rate. Figure 4 illustrates the reconstructed images obtained at 0.75 bpp. Note the better perceptual quality of the joint design around the three circular tumors in the left lung in the figure.

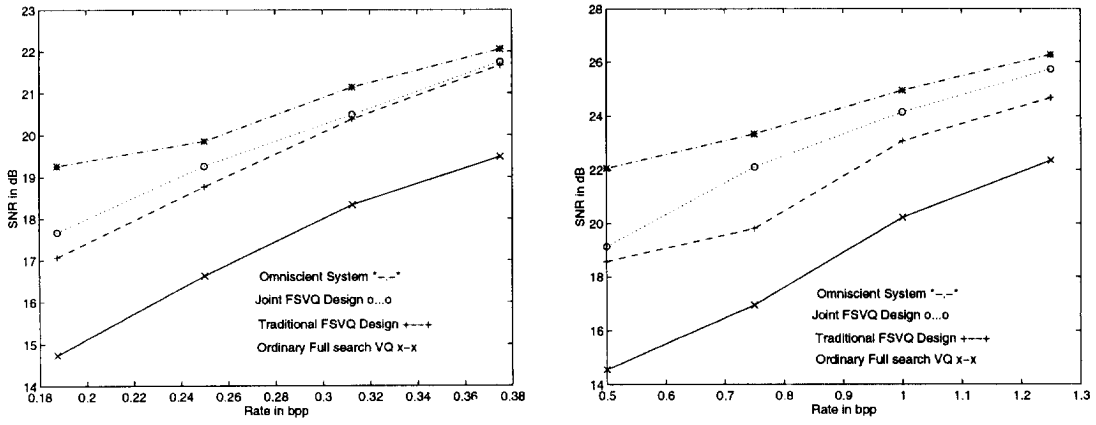

Figure 2: Comparison of algorithms at (a) low rates and (b) high rates

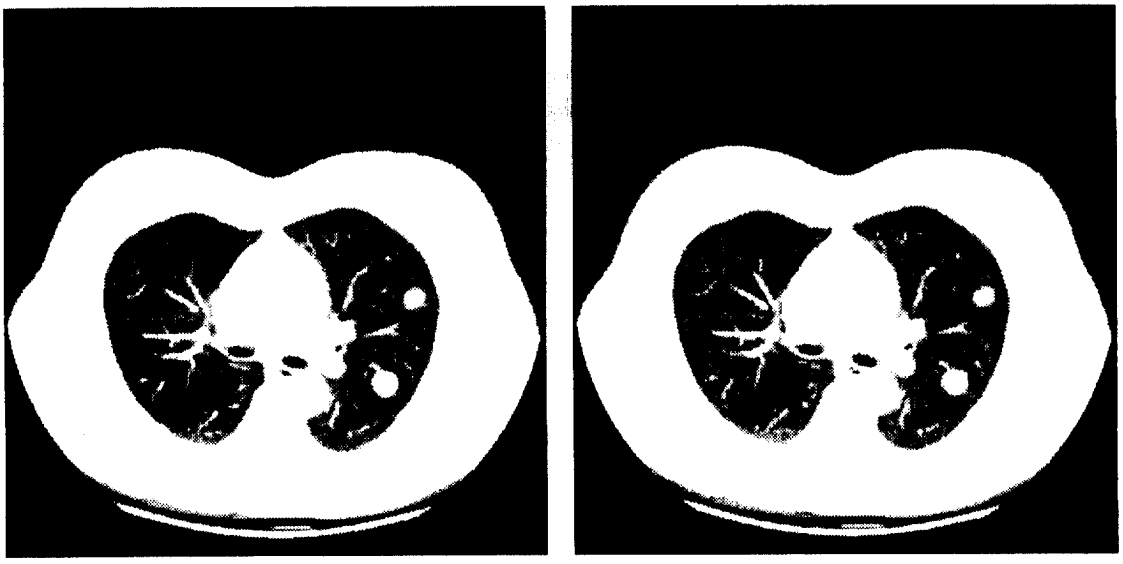

Figure 3: Test image compressed to 0.375 bpp using (a) joint FSVQ design and (b) traditional FSVQ design

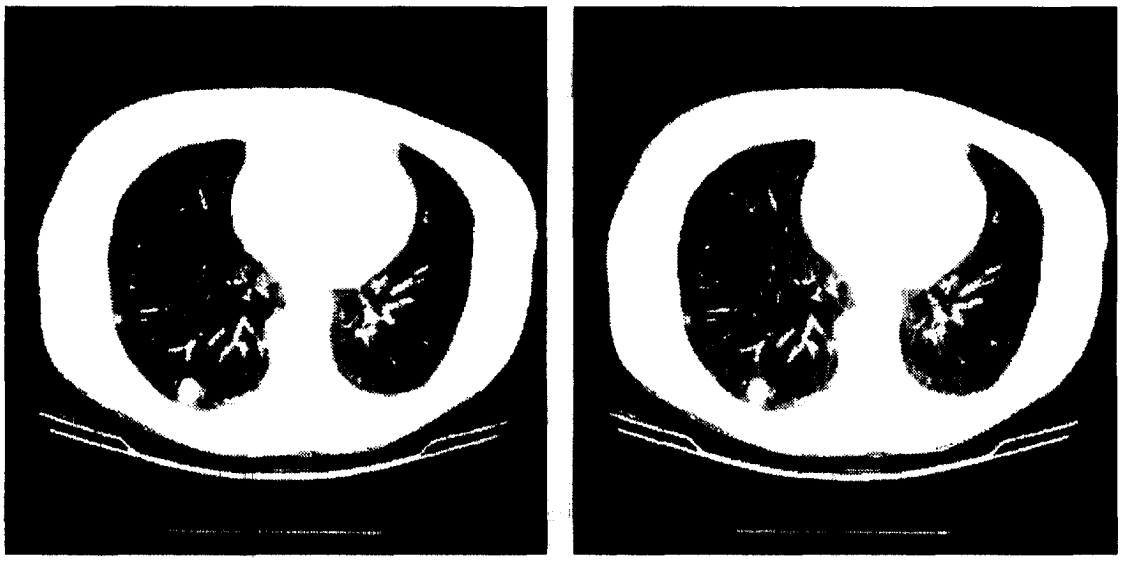

Figure 4: Test image compressed to 0.75 bpp using (a) joint FSVQ design and (b) traditional FSVQ design

4 Conclusions and Future Research

We have presented a novel iterative algorithm for joint codebook and next state function design in constructing an FSVQ. Results presented in this paper demonstrate that the algorithm provides improved performance over traditional FSVQ techniques. Since we obtained improvement using only a very basic classifier, we would expect even better performance with a more sophisticated classifier. In particular, the iterative joint state codebook/classifier improvement algorithm can be used in conjunction with a classification tree grown according to the techniques described in [2]. This may be particularly effective if we start with the codebooks designed by the omniscient system described in Section 3. Another area that can be explored is the use of the proposed algorithm with variable rate full search or tree-structured state codebooks. The results of [6] suggest that this will result in further performance improvement.

5 Acknowledgments

This work was supported by the NSF. The authors gratefully acknowledge the assistance of Dr. Pamela C. Cosman.

References

[1] R. Aravind and A. Gersho. Low-rate image coding with finite-state vector quantization. In *Proceedings ICASSP*, pages 137–140, Tokyo, 1986.

[2] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, CA, 1984.

[3] P. A. Chou and T. Lookabaugh. Conditional entropy-constrained vector quantization of linear predictive coeficients. In *Proceedings ICASSP*, pages 187–200, 1990.

[4] J. Foster, R. M. Gray, and M. Ostendorf Dunham. Finite-state vector quantization for waveform coding. *IEEE Trans. Inform. Theory*, 31:348–359, May 1985.

[5] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, Boston, 1992.

[6] Y. Hussain and N. Farvardin. Variable rate finite-state vector quantization. Presented at the Twenty-Fourth Annual Conference on Information Sciences and Systems, Princeton, NJ, March 1990.