Fuzzy En tropy-Constrained Competitiev Learning Algorithm

Wen-Jyi Hwang, Chienmin Ou, Shi-Chiang Liao and Ching-Fung Chine

> Department of Electrical Engineering Chung Yuan Christian University Chungli, 32023, Taiwan

Abstract. A novel variable-rate vector quantizer (VQ) design algorithm using both fuzzy and competitive learning technique is presented. The algorithm enjoys better rate-distortion performance than that of other existing fuzzy clustering and competitive learning algorithms. In addition, the learning algorithm is less sensitive to the selection of initial reproduction vectors. Therefore, the algorithm can be an effective alternative to the existing variable-rate VQ algorithms for signal compression.

1. Introduction

Entropy-constrained vector quantization (ECVQ) [1] algorithm is a well-known method for variable-rate VQ design. The technique employs batch training process where components of VQ are constructed iteratively one at a time. In addition, during the course of the iterative construction procedure, ECVQ algorithm utilizes crisp clustering technique where each training vector is assigned to only one cluster for training set partitioning. The algorithm have the following disadvantages. First, the performance of iterative design procedure is sensitive to the selection of initial reproduction vectors. Therefore, a bad local optimum might be achieved if an improper set of initial codewords is selected. Moreover, since many training vectors might have features belonging to more than one clusters, it is not nature to assign each training vector to only one cluster during set partitioning process.

To alleviate these drawbacks, a number of competitive learning (CL) algorithms [4, 6] and fuzzy clustering techniques [3, 5] have been proposed for VQ design. As compared with the traditional iterative algorithms, the CL-based algorithms are shown to have more robust rate-distortion performance because the training process of these algorithms are performed in continuous mode. The fuzzy clustering algorithms can also improve the performance of the VQ design since better codevectors might be found by fuzzifying the boundary of clusters. Although using CL or fuzzy clustering algorithm along can improve the design of VQs, it is expected that the fusion of these two techniques can improve the performance of VQs further because the resulting VQs can have the advantages of both algorithms.

The objective of this paper is to present a novel variable-rate VQ design algorithm, termed fuzzy entropy-constrained competitive learning (FECCL) algorithm, which performs VQ training in both continuous and fuzzy mode. To achieve continuous training, similar to the CL algorithms, the weight vectors of the algorithm are updated for each input training vector. However, the continuous training process is not based on the winner-take-all scheme adopted by usual CL algorithms, which results in crisp training. In the FECCL algorithm, for any input training vector, its membership grade in each cluster is first computed. All the weight vectors are then updated according to these membership grades. In order to control the average rate of the VQ after training process, when computing the membership functions, the length of channel codeword representing each weight vector will be taken into account. Simulation results show that the FECCL algorithm has more robust and superior rate-distortion performance than other existing VQ design algorithms where only crisp or batch training are used.

2. Preliminaries

Recall that the encoder \mathcal{W} of a variable-rate VQ [1] can be decomposed into two parts. The first part is the function \mathbf{w} which maps a source vector \mathbf{x} into a source index *i*. The second part, γ , maps the source index *i* into a channel codeword c_i . The decoder \mathcal{Z} can be similarly decomposed into two parts: γ^{-1} and \mathbf{z} . The mapping γ^{-1} is the inverse of γ , and maps c_i back to source index *i*. The mapping \mathbf{z} outputs a reproduction vector $\mathbf{z}(i)$ for the source index *i*. The pair $(\mathcal{W}, \mathcal{Z})$ is referred to as a variable-rate VQ because a source vector is represented by a single channel codeword, and the channel codewords for different source vectors are usually not of the same length.

Given a source vector \mathbf{x} , let $|\gamma(\mathbf{w}(\mathbf{x}))|$ be the length of the channel codeword representing \mathbf{x} . In addition, let $d_n(\mathbf{u}, \mathbf{v})$ be the mean-squared distance between vectors \mathbf{u} and \mathbf{v} . For a given set of training vectors $\mathcal{X} = \{\mathbf{x}_j, j = 1, ..., t\}$, where t is the number of training vectors in the training set, the average rate and average distortion of the VQ are then given as $R = \frac{1}{nt} \sum_{j=1}^{t} |\gamma(\mathbf{w}(\mathbf{x}_j))|$ and $D = \frac{1}{nt} \sum_{j=1}^{t} d_n(\mathbf{x}_j, \mathbf{z}(\mathbf{w}(\mathbf{x}_j)))$, respectively. Let $D_n(R)$ be the function indicates the lowest possible average distortion that can be achieved by a VQ with blocklength n under the rate constraint R. Although the Lagragian methods can not be used for finding D(R) [1], they can be used for finding the convex hull of D(R) be minimizing the following functional:

$$C(\mathbf{w}, \mathbf{z}, \gamma) = \sum_{j=1}^{t} d_n(\mathbf{x}_j, \mathbf{z}(\mathbf{w}(\mathbf{x}_j))) + \lambda |\gamma(\mathbf{w}(\mathbf{x}_j))|.$$
(1)

Here, λ has an interpretation as the slope of a line supporting the convex hull.

We now discuss the optimal conditions for \mathbf{w}, \mathbf{z} and γ that minimizes the functional $C(\mathbf{w}, \mathbf{z}, \gamma)$. Fix γ and \mathbf{z} , the mapping \mathbf{w} that minimizes (1) is the one that satisfies

$$\mathbf{w}(\mathbf{x}_j) = \arg\min_{1 \le i \le N} J(\mathbf{x}_j, \mathbf{z}(i)), \tag{2}$$

where N is the number of reproduction vectors, and $J(\mathbf{x}_j, \mathbf{z}(i)) = d_n(\mathbf{x}_j, \mathbf{z}(i)) + \lambda |\gamma(i)|$. Next, for fixed **w** and γ , the mapping **z** that minimizes (1) is given as

$$\mathbf{z}(i) = \text{centroid of } \{\mathbf{x}_j : \mathbf{w}(\mathbf{x}_j) = i\}.$$
(3)

Finally, the optimal channel encoder γ for fixed **w** and **z** is the entropy encoder designed based on the probability p(i) of each reproduction vector $\mathbf{z}(i)$. That is, the optimal γ satisfies

$$|\gamma(i)| = \log \frac{1}{p(i)}.\tag{4}$$

The ECVQ algorithm uses (2)(3)(4) iteratively for the minimization of $C(\mathbf{w}, \mathbf{z}, \gamma)$ given in (1). The scheme is sensitive to the initial parameters since reproduction vectors are constructed in batch mode. If the initial γ and \mathbf{z} are improperly chosen, then the ECVQ algorithm can easily converge to a bad local minimum. In addition, the clustering operation is a crisp clustering process. Better codewords might be found if a fuzzy clustering process is employed.

The objective of ECCL algorithm [4] is also to minimize $C(\mathbf{w}, \mathbf{z}, \gamma)$. However, unlike the ECVQ algorithm, the ECCL performs the continuous training process for the codebook construction. Given γ , the ECCL algorithm finds the optimal \mathbf{w} and \mathbf{z} satisfying eqs. (2) and (3), respectively, using the following winner-take-all competition scheme [4]:

$$\mathbf{z}(i^*) \leftarrow \mathbf{z}(i^*) + \eta(i^*)(\mathbf{x}_j - \mathbf{z}(i^*)), \tag{5}$$

$$\mathbf{z}(i) \leftarrow \mathbf{z}(i), \quad i \neq i^*,$$
 (6)

where \mathbf{x}_j is an input training vector in the training set \mathcal{X} , $\eta(i)$ is the learning rate of the *i*-th weight vector, and i^* is the index of the winning weight vector based on the following competition rule:

$$i^* = \arg\min_{1 \le k \le N} J(\mathbf{x}_j, \mathbf{z}(k)), \tag{7}$$

Since the ECCL algorithm performs weight vector updating for each input training vector, the algorithm has more robust performance as compared with ECVQ algorithm.

The FECVQ algorithm [3] performs fuzzy clustering during the course of iterative training process. In order to incorporate the membership functions of training vectors into the clustering process, the objective of FECVQ is to minimize the following objective function

$$C^{f}(\mu, \mathbf{z}, \gamma) = \sum_{j=1}^{t} \sum_{i=1}^{N} (\mu_{i,j})^{m} (J(\mathbf{x}_{j}, \mathbf{z}(i))),$$
(8)

where m > 1 indicates the degree of fuzziness, and $\{\mu_{i,j}, i = 1, ..., N\}$ is the set of membership values associated with training vector \mathbf{x}_j , satisfying $\sum_{i=1}^{N} \mu_{i,j} =$ 1 and $\mu_{i,j} \ge 0$. The algorithm employs an iterative training procedure that optimizes μ , \mathbf{z} , and γ one at a time. In particular, for fixed \mathbf{z} and γ , the optimal membership functions μ minimizing $C^f(\mu, \mathbf{z}, \gamma)$ are given as [3]

$$\mu_{i,j} = \frac{\left(\frac{1}{J(\mathbf{x}_j, \mathbf{z}(i))}\right)^{\frac{1}{m-1}}}{\sum_{l=1}^{N} \left(\frac{1}{J(\mathbf{x}_j, \mathbf{z}(l))}\right)^{\frac{1}{m-1}}}.$$
(9)

Since performing fuzzy clustering for codebook construction, FECVQ can find better reconstruction vectors. However, FECVQ also performs training in batch mode, and therefore is sensitive to initial parameters.

3. FECCL Algorithm

The FECCL algorithm has the advantages of both FECVQ and ECCL algorithms. In the algorithm, similar to ECCL technique, the training process is performed in continuous mode. However, unlike the ECCL algorithm, where the weight vector updating is based on winner-take-all scheme, the FECCL algorithm updates weight vectors according to the membership functions, which measure the degree of closeness between the input training vector to each weight vector. During the course of FECCL design, the reproduction vectors $\mathbf{z}(i), i = 1, ..., N$, are first updated in fuzzy and continuous mode based on a fixed channel encoder γ . The source encoder \mathbf{w} is then constructed based on the updated \mathbf{z} . After that, the channel encoder γ is designed according to the new \mathbf{w} . The same process is repeated until the convergence of the algorithm. The design procedure of FECCL algorithm is summarized in Figure 1.

We now discuss each step of the algorithm in more detail. For any input training vector $\mathbf{x}_j \in \mathcal{X}$, the FECCL algorithm updates weight vectors $\mathbf{z}(i), i = 1, ..., N$, according to the following rule:

$$\mathbf{z}(i) \leftarrow \mathbf{z}(i) + \eta(i)(\mu_{i,j})^m (\mathbf{z}(i) - \mathbf{x}_j), \tag{10}$$

where $\eta(i)$ is the learning rate of $\mathbf{z}(i)$. At the first K iterations of FECVQ algorithm, where K > 0 can be prespecified before the design, the membership functions $\mu_{i,j}$, i = 1, ..., N, are computed according to (9), and therefore all the weight vectors are updated for each input training vector \mathbf{x}_j . Consequently, FECVQ algorithm is insensitive to the selection of initial weight vectors. After K iterations, since the VQ encoding process is a crisp selection process, in order to find codewords well-suited for crisp encoding, the FECCL algorithm employs a novel fuzzy-to-crisp transition process for codebook construction. In the process, as the number of iterations increases, the number for weight vectors to be updated for each training vector is gradually reduced until winner-take-all updating mode is reached.

To describe the fuzzy-to-crisp transition process in more detail, we first denote $\mathcal{I}_{j}(k)$ as the set of weight vectors to be updated when \mathbf{x}_{j} is used as the



Figure 1: Design procedure of FECCL algorithm.

input training vector at k-th iteration. Therefore, given any training vector \mathbf{x}_j , the set $\mathcal{I}_j(k) = \{\mathbf{z}(i), i = 1, ..., N\}$ for k = 1, ..., K. Since all weight vectors in the VQ are included in the set $\mathcal{I}_j(k)$ for updating, similar to FECVQ, the computation of mem bership function is based on (9). For k > K, we gradually reduce the number of weight vectors in $\mathcal{I}_j(k)$. In the procedure for removing weight vectors at iteration k > K, we first compute $\overline{J}(k-1)$ defined as

$$\bar{J}(k-1) = \frac{1}{\mathcal{N}(\mathcal{I}_j(k-1))} \sum_{\mathbf{z}(i)\in\mathcal{I}_j(k-1)} J(\mathbf{x}_j, \mathbf{z}(i)),$$
(11)

where $\mathcal{N}(\mathcal{I}_j(k-1))$ is the number of weight vectors in set $\mathcal{I}_j(k-1)$. Based on $\mathcal{I}_j(k-1)$ and $\bar{J}(k-1)$, the set $\mathcal{I}_j(k)$ is constructed according to

$$\mathcal{I}_j(k) = \{ \mathbf{z}(i) \in \mathcal{I}_j(k-1) : J(\mathbf{x}_j, \mathbf{z}(i)) \le \overline{J}(k-1) \}.$$
(12)

Given \mathbf{x}_j , for the weight vectors which are outside $\mathcal{I}_j(k)$, their corresponding membership functions are set to zero, and the weight vectors are not updated. On the other hand, for the weight vectors $\mathbf{z}(i)$ which are inside $\mathcal{I}_j(k)$, the corresponding membership functions are then given as

$$\mu_{i,j} = \frac{\left(\frac{1}{J(\mathbf{x}_j, \mathbf{z}(i))}\right)^{\frac{1}{m-1}}}{\sum_{\mathbf{z}(l) \in \mathcal{I}_j(k)} \left(\frac{1}{J(\mathbf{x}_j, \mathbf{z}(l))}\right)^{\frac{1}{m-1}}}.$$
(13)

These weight vectors are then updated according to (10) with $\mu_{i,j}$ given in (13).

Note that when k increases, the number of weight vectors in $\mathcal{I}_j(k)$ decreases. Therefore, in the FECCL algorithm, the iteration number k might achieve a point where, for each of the training vector $\mathbf{x}_j \in \mathcal{X}$, its $\mathcal{I}_j(k)$ contain only one element. In this case, the corresponding weight vector updating process is identical to the winner-take-all scheme adopted by the ECCL algorithm.

At each iteration of FECCL, after all the training vectors \mathbf{x}_j have been used for weight vector updating, the source encoder \mathbf{w} is then constructed by the following rule:

$$\mathbf{w}(\mathbf{x}) = \arg\min_{1 \le i \le N} J(\mathbf{x}, \mathbf{z}(i)), \tag{14}$$

where **x** is any input vector for encoding, and $\mathbf{z}(i), i = 1, ..., N$, are the reproduction vectors obtained from weight vector updating.

After **w** has been determined, we then partition the training set \mathcal{X} into N clusters $C_1, ..., C_N$, where $C_i = \{\mathbf{x}_j : \mathbf{w}(\mathbf{x}_j) = i\}$. The probability p(i) that the reproduction vector $\mathbf{z}(i)$ will be selected by the source encoder can then be approximated as $p(i) = \frac{\mathcal{N}(C_i)}{t}$, where $\mathcal{N}(C_i)$ is the number of training vectors in C_i . The optimal channel encoder γ is then designed according to (4).

We note that the selection of λ value effects the resulting rate of the VQ design. However, similar to other variable-rate VQ methods, it is difficult to find the relationship between average rate of a VQ and λ value analytically. Hence, the λ value achieving a given target rate R_c might have to be found on the trial-and-error basis.

Finally, as the number of iterations k increases, the FECCL algorithm gradually reduce to ECCL algorithm due to the fuzzy-to-crisp transition process. Since ECCL is guaranteed to converge [4], the weight vectors of FECCL algorithm also converges for sufficient number of iterations.

4. Sim ulation Results and Concluding Remarks

This section presents some simulation results of FECCL algorithm. The training images for the VQ design are two 512×512 images "Pepper" and "Girl." The dimension of training vectors and weight vectors is 4×4 .

Figure 2 shows the rate-distortion performance of various VQ design algorithms. All the methods are designed subject to the same number of reproduction vectors N = 512. The degree of fuzziness for FECCL algorithm is set to be m = 1.1. The fuzzy-to-crisp transition process will be started in the FECCL algorithm after K = 2 iterations are completed. The PSNR values for each algorithm shown in the figure are defined as $10 \log 255^2$ / (MSE of the reconstructed images). Note that the PSNR values are measured on the test image "House" with dimension 512×512 . From Figure 2, it is observed that the FECCL algorithm significantly outperforms FECVQ and ECVQ algorithms. In particular, at the average rate 0.2 bpp, the PSNR of FECCL algorithm is 29.4 dB; whereas, the PSNRs of FECVQ and ECVQ algorithms are only 26.6 dB and 28.0 dB, respectively. This is because FECCL trains weight vectors in



Figure 2: The performance of various VQ design algorithm for image coding. The PSNR is measured on the test image "House."

both fuzzy and continuous modes, and therefore the algorithm is less sensitive to the initial weight vectors, and better reproduction vectors can be found.

Although the FECCL algorithm gradually reduce to the ECCL algorithm as the iteration continues, the FECCL algorithm still has more robust performance with respect to the selection of initial weight vectors. Figure 3 shows the ratedistortion performance of both FECCL and ECCL algorithms with a poor set of initial weight vectors in which the value of every elements of each vector is 100.0 + r, and r is taken randomly from the interval [-1,1]. From Figure 3, it is observed that FECCL outperforms ECCL by at most 1.45 dB (at rate 0.31 bpp). Because of using the fuzzy updating/clustering scheme, the initial weight vectors have less impact on the the resulting rate-distortion performance for FECCL algorithm. On the other hand, the ECCL algorithm is based on the crisp winner-take-all scheme, and therefore the corresponding performance is less robust as compared with FECCL algorithm.

From the results shown above, we conclude that the FECCL algorithm has robust and superior performance for image coding. Therefore, the algorithm can be an effective alternative to other existing variable-rate VQ algorithms.



Figure 3: The PSNRs of FECCL and ECCL with a poor initial set of 512 codewords.

References

- P. A. Chou, T. Lookabaugh, R. M. and Gray, "Entropy-Constrained Vector Quantization," *IEEE Trans. Acoust., Speech, and Signal Processing*, Vol. 37, pp. 31-42, Jan. 1989.
- [2] A. Gersho and R.M. Gray, Vector Quantization and Signal Compression. Kluwer, Norwood, Massachusetts, 1992.
- [3] W.J. Hwang and S.L. Hong, "A Fuzzy Entropy-Constrained Vector Quantizer Design Algorithm and Its Applications to Image Coding," *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sci*ences, Vol. E82-A, pp.1109-1116, June 1999.
- [4] W.J. Hwang, B.Y. Ye and S.C. Liao, "A Novel Entropy-Constrained Competitive Learning Algorithm for Vector Quantization," *Neurocomputing*, pp.133-147, 1999.
- [5] N.B. Karayiannis and P.I. Pai, "Fuzzy Vector Quantization Algorithms and Their Application in Image Compression," *IEEE Trans. Image Pro*cessing, Vol. 4, pp. 1193-1201, 1995.
- [6] T. Kohonen, Self-Organization and Associative memory, 3rd ed., Springer-Verlag, Germany, 1989.