Journal of Information Hiding and Multimedia Signal Processing ©2017 ISSN 2073-4212
Ubiquitous International Volume **8**, Number **6**, November **2017**

Feature-Guided K-Means Algorithm for Optimal Image Vector Quantizer Design

Yun-Feng Xing and Ming-Xiang Guan
School of Electronic and Communication
Shenzhen Institute of Information Technology
Shenzhen, 518172, China
xingyf@sziit.edu.cn

*Zhe-Ming Lu
School of Aeronautics and Astronautics
Zhejiang University
Hangzhou, 310027, P. R. China
Corresponding author: zheminglu@zju.edu.cn

Received March, 2017; revised July, 2017

Abstract. *This paper presents an improved K-means algorithm for image vector quantizer design. The main purpose of the proposed algorithm is to obtain a better initial codebook guided by the edge, brightness and contrast features of the training vectors. Based on these features, the training set is divided into 32 subsets, then the initial codewords are randomly selected or definitely selected based on the maximum distance initialization technique from each subset, where the number of selected codewords is proportional to the number of training vectors in the subset. Experimental results show that, compared with the conventional and modified K-means algorithms, by and large, our feature-guided K-means algorithm converges to a better locally optimal codebook with a fast convergence speed.*

Keywords: Vector quantization, Codebook design, K-means algorithm, Initial codebook generation, Image compression, Feature-guided K-means algorithm.

1. Introduction.

Vector quantization (VQ) is an efficient technique for data compression and pattern recognition [1]. A vector quantizer Q of dimension n and size K can be defined as a mapping from the n-dimensional Euclidean space R^n into a finite set C containing K n-dimensional vectors, that is, $Q : R^n \to C = \{\boldsymbol{y}_1, \boldsymbol{y}_2, ..., \boldsymbol{y}_K\}$. The set C is called the codebook, and $\boldsymbol{y}_i, 1 \leq i \leq K$, are called the codewords. To evaluate the performance of the vector quantizer Q, the squared error $d(\boldsymbol{x}, Q(\boldsymbol{x})) = \|\boldsymbol{x} - Q(\boldsymbol{x})\|^2$ is often chosen to measure the distortion between any input vector $\boldsymbol{x}$ and its reproduction vector $Q(\boldsymbol{x})$. Thus, we can quantify the performance of Q by the average distortion $D = E[d(\boldsymbol{x}, Q(\boldsymbol{x}))]$.

The task of optimal vector quantizer design is to find the codebook that minimizes the average distortion over all possible codebooks. Two conditions should be satisfied for an optimal vector quantizer. One is the nearest neighbor condition, i.e., each training vector should be assigned with the codeword that is closest to it. The other is the centroid condition, i.e., each codeword must be the centroid of the training vectors that are assigned to it. The above two optimality conditions provide an algorithm named K-means algorithm for the design of a locally optimal codebook with iterative codebook improvement. It is also known as the generalized Lloyd algorithm (GLA) or sometimes called the Linde-Buzo-Gray (LBG) algorithm [2], where each iteration consists of the following two steps:

(1) Given a codebook $C_m = \{\boldsymbol{y}_i; i = 1, 2, \ldots, K\}$ obtained from the m-th iteration, find the optimal partitioning of the space R^n, that is, use the first condition to form the nearest-neighbor cells $V_i = \{\boldsymbol{x} : d(\boldsymbol{x}, \boldsymbol{y}_i) < d(\boldsymbol{x}, \boldsymbol{y}_j); j \neq i\}$.

(2) Adopt the second condition to update the codebook $C_{m+1} = \{centroid(V_i); i = 1, 2, \ldots, K\}$ that is the optimal reproduction codebook based on the cells found in Step 1.

It is known that the above K-means algorithm can converge to a locally optimal codebook. Thus, some researchers have combined global optimization techniques such as genetic algorithm [3], particle swam optimization [4] to improve the performance. On the other hand, for the K-means algorithm, both the convergence speed and the performance of the converged codebook depend on the initial codebook. Thus, many algorithms have been proposed to obtain a good initial codebook, including the well known splitting, pruning, pairwise nearest neighbor design (PNN), random initialization and the maximum distance initialization [5]. Furthermore, Lee et al. presented a modified K-means algorithm [6] by simply providing a modification at above codebook updating step where the codeword updating step is as follows: $newcodeword = currentcodeword + scalefactor * (newcentroid - currentcodeword)$. Since this algorithm adopted a fixed value for the scale factor, to further improve the performance, Paliwal and Ramasubramanian proposed the use of a variable scale factor [7] which is a function of the iteration number, i.e., $scalefactor = 1.0 + 9.0/(9.0 + Numberofiterationsfinished)$. There are also some other works to improve the performance of VQ codebook[8, 9, 10]. However, the above conventional initialization techniques and updating methods do not consider the features of each training vector. Therefore, in this Letter, we propose a simple and efficient initialization technique for the K-means algorithm used in image vector quantization by classifying the input vectors based on their features, and then initializing each sub-codebook with the number of codewords being proportional to the number of training vectors in that subset based on the random selection or maximum distance initialization technique.

2. Feature-Guided K-Means Algorithm.

To make full use of the feature distribution of the training set during the initial codebook generation, our algorithm takes into account three kinds of features of each training vector, i.e., edge, brightness and contrast. Assume the training set is $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_M\}$, the whole algorithm is illustrated in Fig.1. First, each training vector $\boldsymbol{x}_i (i = 1, 2, \ldots, M)$ is input into three classifiers, and an overall index $t_i \in \{1, 2, \ldots, 32\}$ is output to denote which class the training vector belongs to. Then, we collect all the training vectors belonging to the same class to generate a subset, and thus we have 32 subsets P_j of sizes $s_j (j = 1, 2, \ldots, 32)$ respectively, where $s_1 + s_2 + \ldots + s_{32} = M$. Afterwards, we initialize $K \times s_j / M$ initial codewords from each subset P_j based on the random selection or maximum distance initialization technique [5], thus we can totally obtain K initial codewords. Finally, the modified K-means algorithm in [7] is performed to generate the final codebook. During the iteration, if an empty cell appears, we just judge the classes that all current codewords belong to and find the class with the least number of codewords, and randomly select a training vector from this class as a new centroid.

Here, we should explain how to classify the training vectors. Inspired by the Structured Local Binary Kirsch Pattern (SLBKP) in [11] that adopts eight 3×3 Kirsch templates to denote eight edge directions, we propose following eight 4×4 templates for edge classification as shown in Fig.2.

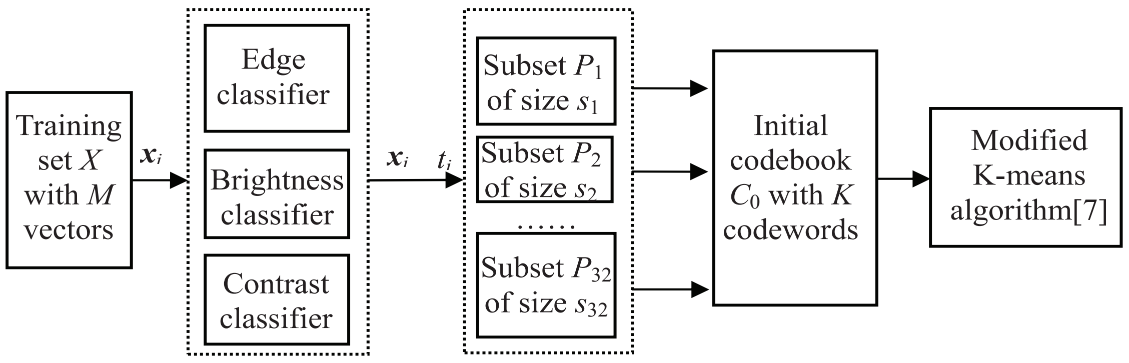
Training set X with M vectors
$\boldsymbol{x}_i$
Edge classifier
Brightness classifier
Contrast classifier
$\boldsymbol{x}_i$ t_i
Subset P_1 of size s_1
Subset P_2 of size s_2
……
Subset P_{32} of size s_{32}
Initial codebook C_0 with K codewords
Modified K-means algorithm[7]

Figure 1. The block diagram of the proposed feature-guided K-means algorithm.

$$\mathbf{E}_1=\begin{bmatrix}-4&2&2&2\\-4&0&0&2\\-4&0&0&2\\-4&2&2&2\end{bmatrix},\mathbf{E}_2=\begin{bmatrix}2&2&2&2\\0&0&2&2\\-4&-4&0&2\\-4&-4&0&2\end{bmatrix},\mathbf{E}_3=\begin{bmatrix}2&2&2&2\\2&0&0&2\\2&0&0&2\\-4&-4&-4&-4\end{bmatrix}$$

$$\mathbf{E}_4=\begin{bmatrix}2&2&2&2\\2&2&0&0\\2&0&-4&-4\\2&0&-4&-4\end{bmatrix},\mathbf{E}_5=\begin{bmatrix}2&2&2&-4\\2&0&0&-4\\2&0&0&-4\\2&2&2&-4\end{bmatrix},\mathbf{E}_6=\begin{bmatrix}2&0&-4&-4\\2&0&-4&-4\\2&2&0&0\\2&2&2&2\end{bmatrix}$$

$$\mathbf{E}_7=\begin{bmatrix}-4&-4&-4&-4\\2&0&0&2\\2&0&0&2\\2&2&2&2\end{bmatrix},\mathbf{E}_8=\begin{bmatrix}-4&-4&0&2\\-4&-4&0&2\\0&0&2&2\\2&2&2&2\end{bmatrix}$$

Figure 2. Eight 4×4 templates for edge classification.

Assume the input image X is divided into non-overlapping 4×4 blocks, the edge classification process can be illustrated as follows: First, we perform eight 4×4 edge orientation templates on each 4×4 block $x(p,q), 0 \le p < 4, 0 \le q < 4$, obtaining an edge orientation vector $\boldsymbol{v} = (v_1, v_2, \dots, v_8)$ with its components $v_i (1 \le i \le 8)$ being computed as follows:

$$v_i = \left|\sum_{p=0}^{3}\sum_{q=0}^{3}[x(p,q)\cdot e_i(p,q)]\right| 1 \le i \le 8 \tag{1}$$

Where $e_i(p,q)$ denotes the element at the position (p,q) of $\mathtt{E}_i$. Thus, an input block $x(p,q)$ is classified into the j-th category if

$$j = \arg\max_{1 \le i \le 8} v_i \tag{2}$$

That is to say, we can classify each vector into one of eight categories according to its edge orientation. For the brightness classifier, we just calculate the average value μ of all components in each training vector as follows

$$\mu = \frac{1}{16}\sum_{p=0}^{3}\sum_{q=0}^{3}[x(p,q)] \tag{3}$$

And then we classify the input vector into the bright or dark class based on the threshold 128.0 for 256 gray-scale images. For the contrast classifier, we calculate the contrast σ by the following formula

$$\sigma = \frac{1}{16}\sum_{p=0}^{3}\sum_{q=0}^{3}[x(p,q) - \mu] \tag{4}$$

And then we classify the input vector into the high-contrast or smooth class based on the threshold 3.0. Thus, in total, based on above three classifiers, we can classify the training vectors into $8 \times 2 \times 2 = 32$ categories.

3. **Experimental Results.** To demonstrate the performance of the proposed feature-guided K-means algorithm, we compared our scheme with the traditional K-means algorithm (A1), the modified K-means algorithm with the fixed scale value 1.8[6] (A2) in the codebook updating step and the modified K-means algorithm with a variable scale value [7] (A3). Two initialization techniques, i.e., random selection and maximum distance initialization, were tested. In our experiments, we used three 512×512 monochrome images with 256 gray levels, Lena Baboon and Peppers. We divided each image into 16384 blocks, and each block is 4×4 pixels in size. We experimented with different codebook sizes of 256, 512, and 1024. The quality of the coded images is evaluated by PSNR. For the case of random selection, the performance is averaged over ten runs.

All the algorithms are terminated when the ratio of the mean squared error difference between two iterations to the mean squared error of the current iteration is within 0.0001 or 0.01%. In Table 1, the PSNR values and the numbers of iterations for different images with different codebook sizes under the random selection are shown, where 'B' and 'A' denote the best and the average results over ten runs respectively. From Table 1, we can see that, if the random selection technique is used, our scheme requires the least average number of iterations than other algorithms and can get better codebooks than A1 and A3 algorithms. In Table 2, the PSNR values and the numbers of iterations for different images with different codebook sizes under the maximum distance initialization [5] are shown. From Table 2, we can see that, if the maximum distance initialization is used, our scheme can obtain the best codebook in a relatively less number of iterations.

4. **Conclusions.** This paper presents an improved K-means algorithm for image vector quantization. The main idea is to classify the training vectors into 32 categories based on the edge, brightness and contrast features of each vector, and then generate initial codewords from each category with the number of codewords proportional to the number of vectors in each category. The experimental results based on three test images show that our algorithm can converge to a better locally optimal codebook with a fast convergence speed by and large.

Acknowledgments. This work was partly supported by the Public Good Research Project of Science and Technology Program of Zhejiang Province, China. (NO.2016C31097) and the Science and Technology Project of QuZhou, Zhejiang, China.(NO.2015Y005).This work was also supported by the National Natural Science Foundation of China (No.61401288), the Guangdong Province higher vocational colleges & schools Pearl River scholar funded scheme (2016).

TABLE 1. Performance comparison with random selection initialization. (CBsize: codebook Size; itr: number of iterations).

Codebook size		**256**		**512**		**1024**	
Performance		**MSE**	**itr**	**MSE**	**itr**	**MSE**	**itr**
Lena	**A1:B**	30.447	25	31.293	29	32.138	27
	A1:A	30.379	36	31.237	38	32.083	34
	A2:B	30.505	22	31.453	23	32.439	20
	A2:A	**30.465**	26	**31.420**	27	32.387	26
	A3:B	30.470	16	31.420	14	32.452	17
	A3:A	30.436	23	31.393	24	32.383	22
	Our:B	30.504	15	31.466	17	32.425	15
	Our:A	30.448	**22**	31.404	**24**	**32.391**	**19**
Baboon	**A1:B**	23.23	35	23.893	31	24.616	20
	A1:A	23.21	44	23.874	39	24.589	26
	A2:B	23.267	30	23.954	26	24.764	19
	A2:A	**23.245**	36	23.938	30	24.745	23
	A3:B	23.253	23	23.946	20	24.746	15
	A3:A	23.231	29	23.927	25	24.723	18
	Our:B	23.255	23	23.965	20	24.763	14
	Our:A	23.236	**29**	**23.940**	**24**	**24.747**	**17**
Peppers	**A1:B**	29.863	32	30.591	24	31.368	19
	A1:A	29.799	43	30.540	37	31.313	29
	A2:B	29.956	25	30.789	25	31.712	18
	A2:A	**29.917**	37	30.714	33	**31.625**	24
	A3:B	29.912	20	30.758	19	31.732	16
	A3:A	29.861	30	30.710	24	31.593	20
	Our:B	29.962	16	30.789	17	31.685	16
	Our:A	29.908	**25**	**30.720**	**23**	31.606	**19**

TABLE 2. Performance comparison with maximum distance initialization. (CBsize: codebook size; itr: number of iterations).

Codebook size		**256**		**512**		**1024**	
Performance		**MSE**	**itr**	**MSE**	**itr**	**MSE**	**itr**
Lena	**A1**	30.325	**20**	31.482	23	32.953	24
	A2	30.438	23	31.591	23	33.071	15
	A3	30.421	21	31.587	**14**	33.078	**13**
	Our	**30.556**	25	**31.724**	19	**33.152**	19
Baboon	**A1**	23.205	35	23.871	19	24.738	23
	A2	23.252	25	23.975	20	24.839	19
	A3	23.229	**18**	23.966	19	24.823	**19**
	Our	**23.259**	22	**23.982**	**19**	**24.840**	24
Peppers	**A1**	29.900	22	31.109	16	32.635	22
	A2	30.070	18	31.226	18	32.709	15
	A3	30.049	19	31.218	**13**	32.692	**13**
	Our	**30.189**	**13**	**31.336**	16	**32.755**	16

REFERENCES

[1] A. Gersho, and R. M. Gray, Vector quantization and signal compression, Springer Science & Business Media, 2012

[2] Y. Linde, A. Buzo, and R. M. Gray, An algorithm for vector quantizer design, *IEEE Transactions on Communications*, vol. 28, no.1, pp. 84–95, 1980.

[3] S. Bhatia, New improved technique for initial cluster centers of K means clustering using genetic algorithm, *IEEE International Conference for Convergence of Technology*, pp. 1-4, 2014.

[4] H. M. Feng, and J. H. Horng, VQ-based fuzzy compression systems designs through bacterial foraging particle swarm optimization algorithm, *The Fifth International Conference on Genetic and Evolutionary Computing*, pp. 256–259, 2011.

[5] I. Katsavounidis, C. C. J. Kuo, and Z. Zhang, A new initialization technique for generalized Lloyd iteration, *IEEE Signal Processing Letters*, vol. 1, no.10, pp. 144–146, 1994.

[6] D. Lee, S. Baek, and K. Sung, Modified K-means algorithm for vector quantizer design, *IEEE Signal Processing Letters*, vol. 4, no.1, pp. 2–4, 1997.

[7] K. K. Paliwal, and V. Ramasubramanian, Comments on modified K-means algorithm for vector quantizer design, *IEEE Transactions on Image Processing*, vol. 9, no.11, pp. 1964–1967, 2000.

[8] T. C. Lu, and C. Y. Chang, A Survey of VQ Codebook Generation, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 1, no. 3, pp. 190-203, 2010.

[9] J. S. Pan, F. R. McInnes and M. A. Jack, Fast Clustering Algorithms for Vector Quantization, *Pattern Recognition*, vol. 29, no. 3, pp.511–518, 1996.

[10] J. S. Pan, F. R. McInnes and M. A. Jack, VQ Codebook Design Using Genetic Algorithms, *Electronics Letters*, vol. 31, no. 17, pp.1418–1419, 1995.

[11] G. Y. Kang, S. Z. Guo, D. C. Wang, L. H. Ma, and Z. M. Lu, Image retrieval based on structured local binary kirsch pattern, *IEICE Transactions on Information and Systems*, vol. 96, no.5, pp. 1230–1232, 2013.