An Affinity Propagation Based method for Vector Quantization Codebook Design

Wu Jiang, Fei Ding and Qiao-liang Xiang
Department of Optoelectronic Engineering
Nanjing University of Posts and Telecommunications
{albert.w.jiang, danix800, qiaoliangxiang}@gmail.com

***Abstract*—In this paper, we firstly modify a parameter in affinity propagation (AP) to improve its convergence ability, and then, we apply it to vector quantization (VQ) codebook design problem. In order to improve the quality of the resulted codebook, we combine the improved AP (IAP) with the conventional LBG algorithm to generate an effective algorithm call IAP-LBG. According to the experimental results, the proposed method not only improves its convergence abilities but also is capable of providing higher-quality codebooks than conventional LBG method.**

I. Introduction

Vector quantization (VQ) is an effective method in lossy data compression, and is widely used in the field of speech coding, image coding, video compression, etc. [1] [2] [3]. According to the VQ processes, we know that, for image compression, the quality of the reconstructed image highly depends on the quality of the codebook. Hence the codebook design is a very important task for VQ. Essentially, the codebook design is a clustering problem. Given a set of training vectors $\mathbf{V} = \{V_n : n = 1, 2, \ldots, N\}$, the goal is to search for a map : $\mathbf{V} : \{V_n : n = 1, 2, \ldots, N\} \rightarrow \mathbf{C} : \{C_m : m = 1, 2, \ldots, M\}, N \gg M$ which maps each training vector (or training point) V_n to its cluster centroid C_m. This map shall minimize the cost function. Here, a simple squared Euclidean distance measure $E = \Sigma\|V_n - C_m\|^2$ is used as the cost function.

A generalized algorithm was proposed by Linde, Buzo, and Gray (LBG) [4]. It is the most popular codebook design method. LBG iteratively applies two optimality conditions (nearest neighbor condition and centroid condition) to generate a codebook. However, it suffers from local optimality and is sensitive to the initial solution. If the initial solution is poor, the resulted codebook's quality will probably be poor, and as a result it will be difficult to produce a high-quality image.

Recently, a powerful algorithm called Affinity Propagation (AP) for unsupervised clustering was proposed by Frey and Dueck [5] . In AP algorithm, each point in a set is viewd as a node in a network. AP is based on message passing along edges of the network, following the idea of belief propagation [6] [7]. AP takes input real-value similarities $s(n, m)$ which indicate how well the data point m is suited to be the cluster centroid to data point n, and then, two kinds of real-value messages "*responsibility*" $r(n, m)$ and "*availability*" $a(n, m)$ are exchanged among data points until a high-qulity set of cluster centroids and corresponding clusters gradually emerges [5]. Breifly, there are two significant advantages of AP: one is its high-quality clustering capabilty; the other is its computational efficiency, especially for large data sets [8]. However, in AP, for *self-similarity* is the same for each point, all data points are simultaneously considered as potential clustering centroids. Actually, this feature brings a drawback for AP, since it will be more difficult to converge.

In this paper, we propose an improved AP (IAP) algorithm by modifying a parameter called *network-support similarity* $ns(m, m)$ for each point which improves the algorithm's convergence abilities. In the original AP, all the *self-similarities* [5] are the same for all points,however, *network-support similarity* $ns(m, m)$ for each point changes according to the network supports. As to a point in the network, *network-support similarity* equals to the average squared Euclidean distance from this point to the other points in the whole network. We consider that data points with larger values of *network-support similarities* are more probably to be chosen as clustering centroids. Also we offer a parameter called *ratio of similarity* rs to control the number of codewords needed. In addition, based on IAP algorithm, we propose an algorithm called IAP-LBG to effectively design the VQ codebook. Due to the strong clustering ability of IAP, the codebook's quality is further improved.

II. LBG Algorithm

Suppose that we want to design a codebook with M codewords from a training set $\mathbf{V} = \{V_n : n = 1, 2, \ldots, N\}$, where $N \gg M$. The processes of the LBG algorithm are described as follows.

Step 1 Randomly select M vectors from the training set $\mathbf{V}$ as the initial codebook $\mathbf{C} = \{C_m : m = 1, 2, \ldots, M\}$;

Step 2 For $\forall V_n \in \mathbf{V}$, find the closest codeword in the current codebook $\mathbf{C}$ according to the squared Euclidean distance $E = \|V_n - C_m\|$ and then add the training vectors into the corresponding cluster of the closest codeword found.

Step 3 For $\forall C_m \in \mathbf{C}$, calculate the overall average distortion $D = \frac{1}{J \times M} \sum_{m=1}^{M} \sum_{j=1}^{J} E$ between the codeword and each training vector of the associated cluster. J is the total number of training vectors in the associated cluster.

Step 4 If the difference in overall average distortion between the last two successive iterations is smaller than some threshold or a certain number of iterations is reached, then terminate the iteration.

Step 5 For each codeword in the current codebook $\mathbf{C}$, evaluate the centroid of its associated cluster and take the centroid as a new codeword for the next iteration. Go back to Step 2.

The conventional LBG algorithm suffers from local optimality and is sensitive to the initial solution. So if the initial solution is poor, the generated codebooks quality will probably be poor, and as a result it will be difficult to produce a high-quality image when decodeing.

III. Affinity Propagation Algorithm

AP takes as input similarities $s(n,m)$ which indicate how well the data point m is suited to be the centroid for data point n. Here the similarity is set to be a negative squared Euclidean distance:

$$s(n,m) = -\|V_n - C_m\|^2. \tag{1}$$

$s(m,m)$ indicates that data points with larger values are more likely to be chosen as clustering centroids. The number of the final examplars is influenced by the value of $s(m,m)$. In the conventional AP, all data points are simultaneously considered as potential examplars so the authors set all $s(m,m)$ to be the same value [5].

In the processing, two kinds of message are exchanged among data points, and each takes into account a different kind of competition. The *"responsibility"* $r(n,m)$, sent from data point n to candidate exemplar point m, reflects the accumulated evidence for how well-suited point m is to serve as the exemplar for point n, taking into account other potential exemplars for point n. The *"availability"* $a(n,m)$ sent from candidate exemplar point m to point n, reflects the accumulated evidence for how fitting it would be for point n to choose point m as its exemplar, taking into account the support from other points that point m should be an exemplar. To begin with, the availabilities are initialized to zero, and in the whole process, they followes the updating rule below.

$$r(n,m) = s(n,m) - \max_{m' \text{ s.t. } m' \neq m} \left\{a(n,m') + s(n,m')\right\} \tag{2a}$$

$$r(m,m) = s(m,m) - \max_{m' \text{ s.t. } m' \neq m} \left\{a(m,m') + s(m,m')\right\} \tag{2b}$$

$$a(n,m) = \min\left\{0, r(m,m) + \sum_{n' \text{ s.t. } n' \notin n,m} \max\left\{0, r(n',m)\right\}\right\} \tag{2c}$$

$$a(m,m) = \sum_{n' \text{ s.t. } n' \neq m} \max\left\{0, r(n',m)\right\} \tag{2d}$$

Messages are updated on the basis of simple formula that search for minima of an appropriately chosen energy function. At any point in time, the magnitude of each message reflects the current affinity that that one data point has for choosing another data point as its exemplar.

For point n, the value of m that maximizes $a(n,m) + r(n,m)$ either identifies point n as an exemplar if $m = n$, or identifies the data point that is the exemplar for point n [5]. The message-passing procedure may be terminated after a fixed number of iterations, after changes in the messages fall below a thereshold, or after the local decisions stay constant for some number of iterations.

IV. Proposed Algorithm

Since in the conventional AP, the authors consider that all data points can be equally suitable as exemplars, they set *self-similarities* of each point to be the same. However, we propose a different view of $s(m,m)$. We argue that the *self-similarity* of each point should vary according to the similarities between this point and the others. A point may "love" to take itself as a exemplar more if it "knows" there are more other points choosing it to be a exemplar. We call this rule *network-support similarities* which, in this paper, is denoted as $ns(m,m)$:

$$ns(m,m) = \frac{\sum_{m' \text{ s.t. } m' \neq m} \left\{s(m',m)\right\}}{N-1} \tag{3}$$

We consider that the point whose $ns(m,m)$ is larger would be more appropriate to be an examplar. Because the cluster shape is regular in VQ codebook design, there is only one centroid for each cluster. As to a point, when more points support it to be a centroid, it should prefer to choosing itself as a centroid than other points. In order to get the very number of codewords, we set a tuning parameter called *ratio of network-support similarities* rs. And we find that the codeword number decreases monotonously with rs.

$$s(m,m) = rs \times ns(m,m) \tag{4}$$

To fine-tune the final solution, we use LBG algorithm after the process of the IAP. Since some codewords may be replaced, we must update the associated clusters to reduce distortion errors. Because IAP could generate good codebooks in the first step, the following LBG process would generate effective codebooks finally.

The process of the proposed algorithm is as follows.

Step 1 Caculate *similarities* $s(n,m)$ for all data points according to Eq.(1).
Step 2 Caculate *network-support similarities* $ns(m,m)$ for all data points according to Eq.(3), and initialize $rs = 0.1$.
Step 3 Initialize all $a(n,m) = r(n,m) = 0$.
Step 4 $\forall n \in \{1 : N\}$, update the N *responsibilities* $r(n,m)$ and then the N *availabilities* $a(m,n)$ parallelly according to Eqs.(2).
Step 5 Identify the exemplars C_m by looking at the maximum value of $r(n,m) + a(n,m)$ for given n.
Step 6 Repeat Steps 4-5 until there is no change in exemplars for a large number of iterations.
Step 7 Modify rs and repeat Step 3-6 to get the codebook of the right size.
Step 8 Use the codebook generated in Step 6 as an initial codebook then continue to use LBG method to generate the finial codebook.

Compared with the original AP, firstly, our improved AP (IAP) could converge much faster (see Figure 1). Secondly, our IAP-LBG algorithm could performance better on generating codebooks for VQ (see Table 1 and Table 2).

V. Experimental Results

Five 256×256 pixels monochrome images ("peppers", "lena", "bridge", "camera", and "baboon") with 256 gray levels are used to evaluate the effectiveness of the proposed algorithm. Firstly, we train respective codebooks of size 256 with 16 dimensions using corresponding images, then we take the codebook of image "peppers" as a universal codebook and use all images to test its quality. In addition, in the process of generating the codebook of "peppers", we also compare the convergence abilities of IAP with the conventional AP algorithm.

Performance comparisons are made among the conventional LBG algorithm [4], the conventional AP algorithm [5], the IAP algorithm and the IAP-LBG algorithm. For LBG, the maximum number of iterations is set to be 50, and threshold is 10^{-8} and the final results are obtained after 20 runs. For AP, IAP and IAP-LBG, when updating the message, we also use a parameter called *damping factor* [5] to avoid numerical oscillations that arise in some circumstances. The value of damping factor is set to be 0.5 in all of our experiments. As to AP, IAP and IAP-LBG, since rs is modified by the characters of the training sets, rs varies as image changes. We tune the parameter rs to obtain 256-size codebooks. They are given as follows, $rs(peppers) = 0.086, rs(lena) = 0.14, rs(camera) = 0.149, rs(baboon) = 0.280, rs(bridge) = 0.203$.

Firstly, We compare the convergence abilities of IAP with the conventional AP algorithm. From Figure 1, we can clearly see that it only takes IAP 71 iterations to converge, however, it takes AP more than 100 iterations. Moreover, the value of the energy function is almost the same.

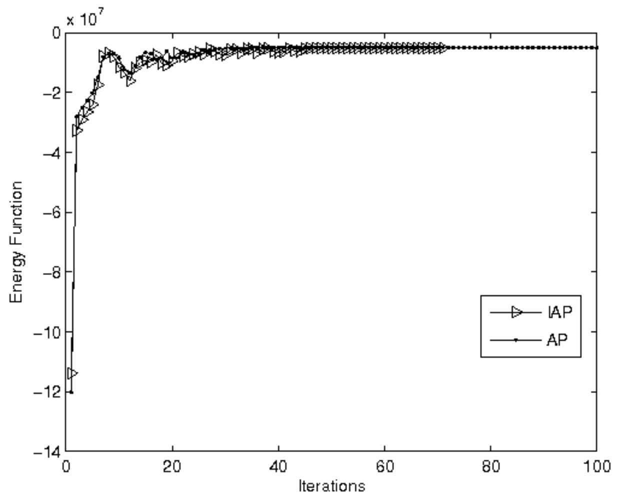

Fig. 1. Comparisons on convergence abilities of AP and IAP

Comparisons measured by PSNR (dB) on genarating codebooks for the five different images are compared among the four methods. Results are shown in Table 1 and Table 2. The codebooks used in Table 1 are generated from the training sets accordingly, and the codebook used in Table 2 is generated from the training set of the "peppers". From Table 1, we can see that IAP-LBG method can improve the PSNR of the generated codebook by 0.62 dB compared with conventional AP, and 0.95 dB compared with conventional LBG averagely. From Table 2 we can see that IAP-LBG algorithm can improve the PSNR by 0.18 compared with conventional AP, and 0.28 compared with conventional LBG averagely. In a word, the proposed algorithm in this paper is really effective.

Algorithms	LBG	AP	IAP	IAP-LBG
peppers	31.18	31.46	31.38	32.04
lena	29.67	30.02	29.94	30.64
camera	25.95	27.68	27.69	28.49
baboon	26.52	26.11	26.03	26.71
bridge	25.48	25.18	25.07	25.67
average	27.76	28.09	28.02	28.71

TABLE I
PSNR (dB) within training set

VI. Conclusions

In this paper we propose a method called IAP-LBG which improves the quality of VQ codebook. Firstly we improve the convergence abilities of the conventional AP algorithm by modifying a parameter called *network-support similarities*, then take this codebook as initial codebook and

Algorithms	LBG	AP	IAP	IAP-LBG
peppers	31.18	31.46	31.38	32.04
lena	27.34	27.40	27.39	27.54
camera	22.55	22.79	22.57	22.75
baboon	24.97	24.89	25.01	25.05
bridge	23.65	23.68	23.66	23.74
average	25.94	26.04	26.00	26.22

TABLE II
PSNR (dB) outside training set

use the conventional LBG method to generate a high-quality codebook. In the experiment, performance comparisons made among five different images potently proved its effectiveness in reconstructed images quality.

VII. Acknowledgements

We acknowledge very useful discussions with Yu-Xuan Wang, and we are grateful to ECHO lab for providing us with the facilities we need to finish our work.

References

[1] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Kluwer Academic Publishers, 1992.
[2] R. M. Gray, “Vector quantization,” *IEEE Acoustics, Speech, Signal Processing Magazine, Vol. 1, pp. 4-29*, April 1984.
[3] N. M. Nasrabadi and R. A. King, “Image coding using vector quantization: A review,” *IEEE Transactions on Communications, Vol. 36, pp. 957-971*, 1988.
[4] Y. Linde, A. Buzo, and R. M. Gray, “An algorithm for vector quantizer design,” *IEEE Transactions on Communications, Vol. 28, pp. 84-95*, January 1980.
[5] B. J. Frey and D. Dueck, “Clustering by passing messages between data points,” *Science Magazine, Vol. 315, pp. 972-976*, February 2007.
[6] J. S. Yedidia, W. T. Freeman, and Y. Weiss, “Understanding belief propagation and its generalizations,” *Mitsubishi Electric Research Laboratories*, 2002.
[7] J. S. Yedidia, W. T. Freeman, and Y. Weiss, “Belief propagation and generalizations,” *IEEE Transactions on Information Theory, Vol. 51, pp. 2282*, 2005.
[8] M. Leone, Sumedha, and M. Weight, “Clustering by soft-constraint affinity propagation: Applications to gene-expression data,” *arXiv: 0705.2646vl*, 2007.