A NEW FUZZY REINFORCEMENT LEARNING VECTOR QUANTIZATION ALGORITHM FOR IMAGE COMPRESSION

Wenhuan Xu, Asoke K. Nandi,

Department of Elec. Engineering and Elec.,
University of Liverpool, Brownlow Hill,
Liverpool L69 3GJ, UK
e-mail:xuwh@liv.ac.uk; a.nandi@liv.ac.uk

Jihong Zhang

Information Engineering Faculty,
Shenzhen University,
Shenzhen 518060, P. R. China
e-mail:zhangjh@szu.edu.cn

ABSTRACT

A new unsupervised *Fuzzy Reinforcement Learning Vector Quantization* (FRLVQ) algorithm for image compression based on the combination of fuzzy K-means clustering algorithm and topology knowledge is proposed. In each iteration of *Reinforcement Learning* (RL), the size and direction of the movement of a codevector is decided by the overall pair-wise competition between the attraction of each training vector and the repellent force of the corresponding winning codevector. While each training vector only affects the winning codevector in the *Generalised Lloyd Algorithm* (GLA) [1] strategy, and only the attraction of training vectors are considered in the *Fuzzy K-means* (FKM) [1] strategy. The competition is measured by the membership function. Simulation results are presented to compare the proposed FRLVQ with GLA and FKM algorithms. It is apparent that FRLVQ has the better quality of codebook design, is very insensitive to the selection of the initial codebook, and relatively insensitive to the choice of learning rate sequences.

1. INTRODUCTION

Vector Quantization (VQ) techniques have been widely used in the past decade as a powerful data compression technique for use in transmission and storage systems. Design of a vector quantizer is accomplished by generating a codebook from the training data based on the minimisation of an average distortion measure. GLA is the most popular minimization technique using a gradient descent algorithm. It is simple to implement but it strongly depends on the selection of the initial codebook. Several studies have been carried out to attempt a globally optimum codebook design. The classical approaches to this problem include *Stochastic Relaxation* (SR) technique introduced by Zeger [1], and *Soft Competition Scheme* (SCS) introduced by Yair [1].

However, all these techniques mentioned above are based on crisp decisions in the sense that each training vector is assigned to a single cluster according to some pre-specified criterion, ignoring the possibility that this training vector may belong to a different cluster. Based on the idea of fuzzy sets, the FKM algorithm was firstly proposed by Dunn [1]. FKM assigns each training vector to multiple clusters with some degree of certainty measured by the membership function. Thus, the partition of training vector space is based on soft decisions. Although FKM produces better result in codebook design, it is typically a time consuming process. To overcome this problem, Karayiannis developed a *Fuzzy Vector Quantization* (FVQ) algorithm [2], which is based on a flexible strategy allowing the transition from soft to crisp decisions. However, these FKM-group algorithms do not result in a better codebook than the original FKM.

In recent years, conventional competitive learning algorithms for unsupervised learning in artificial neural networks have been widely used for vector quantization tasks [3] - [6]. The first study of *Learning Vector Quantization* (LVQ) in clustering algorithm was done by Kohonen [4], which is a supervised learning strategy. This LVQ does have a strong connection with the K-means algorithm. In each iteration, LVQ updates a winner node in the clustering set for each input training vector. Similarly to FKM, a fuzzy learning VQ updates all the nodes for each input training vector regarding the uncertainty degree of clustering. Some related works include *Differential Competitive Learning* (DCL) algorithm introduced by Kong [5], and *Centroid Neural Networks* (CNN) learning algorithm introduced by Park [6]. Both of DCL and CNN introduced a strategy of reward and punishment of learning coefficients for winner nodes and loser nodes.

At this stage, it seems that none of the algorithms mentioned above have achieved a better resulting codebook than FKM. The main reason is that these methods still suffer from the problem of becoming trapped in local minimum of the average distortion measure. Nevertheless, fuzzy clustering algorithms offer an approach to reduce the dependence of the resulting codebook on the selection of the initial codebook.

This paper presents a strategy of reinforcement learning, which exploits the advantages offered by fuzzy clustering algorithms and competitive learning algorithms. The general idea is that not only the Euclidean distances between the present codevectors with each input training vector are considered, but also their topology in multi-dimensional Euclidean space. This strategy enables updating codevectors to map accurately the density distribution of the training vectors to their own. It is noteworthy that RL is a time-consuming process. Thus it is considered as a pre-processing stage before applying other fast FKM algorithms.

2. VECTOR QUANTIZATION

Let X be a training vectors set of size M and dimension l, i.e $X = \{x_1, x_2, ..., x_M\}, x_i \in R^l, \forall i = 1, 2, ..., M$, where R is an l-dimensional Euclidean space. Let Y be a codewords set of size N and dimension l, i.e. $Y = \{y_1, y_2, ..., y_N\}, y_j \in R^l, \forall j = 1, 2, ..., N$. A vector quantiser is designed by assigning the M training vectors to N clusters. Each training vector is represented by a codeword. The quality of the codebook design is often meas-

ured by the average of the absolute distortion between the original and reconstructed vectors, which is often represented by D:

$$D = \frac{1}{M}\sum_{m=1}^{M} d_{1min}^2(x_i) = \frac{1}{M}\sum_{m=1}^{M} \min_{y_j \in Y} d_1^2(x_i, y_j) \quad (1)$$

where Euclidean distance $d_1(x_i, y_j) = \| x_i - y_i \|$.

2.1. GLA Algorithm

The strategy of GLA [1] is from the K-means clustering algorithm. In each iteration, the GLA algorithm assigns each training vector to a certain cluster based on the nearest neighbour condition, which can be defined as a membership function as following:

$$\mu_j(x_i) = \begin{cases} 1 & \text{if } d_1(x_i, y_j) = d_{1min}(x_i) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

Then the codebook vectors can be evaluated by the function defined below:

$$y_j = \frac{\sum_{i=1}^{M} \mu_j(x_i)^m x_i}{\sum_{i=1}^{M} \mu_j(x_i)^m} \quad (3)$$

where m is a parameter that controls the "fuzziness" of the membership function, $0 < m < \infty$, normally $1 \leq m \leq 2$. In this paper, m is chosen to be 1 for all VQ algorithms. The GLA algorithm is summarised in Table 1.

	Begin
	Select a threshold ε
	Select an initial codebook $Y = \{y_1, y_2, ..., y_N\}$
	Evaluate D according to (1)
1	$D_{old} = D$
	$i = 0$
2	$i \leftarrow i + 1$
	Evaluate $\mu_j(x_i)$ using (2),$\forall j = 1, 2, ..., N$
	If $i < M$, then go to step 2
3	Evaluate y_j using (3),$\forall j = 1, 2, ..., N$
4	Evaluate D according to (1)
	If $(D_{old} - D)/D_{old} > \epsilon$, then go to step 1
	End

Table 1. GLA algorithm.

2.2. Fuzzy K-means algorithm

The FKM [1] algorithm assigns each training vector a membership value between zero and one that indicates the possibility of belonging to a certain cluster of the codebook. The most popular membership function is defined as:

$$\mu_j(x_i) = \left[\sum_{p=1}^{N} \left(\frac{d_1(x_i, y_j)}{d_1(x_i, y_p)}\right)^{\lambda}\right]^{-1} \quad (4)$$

where λ is a parameter that controls the "fuzziness" of the distortion, normally $0 < \lambda < \infty$. In this way, a fuzzy partition of the training vectors specifies the degree of membership of each vector in each of the N clusters. The algorithm strategy is very similar to GLA shown in Table 1. The only difference is that the membership function eq (2) is replaced by eq (4).

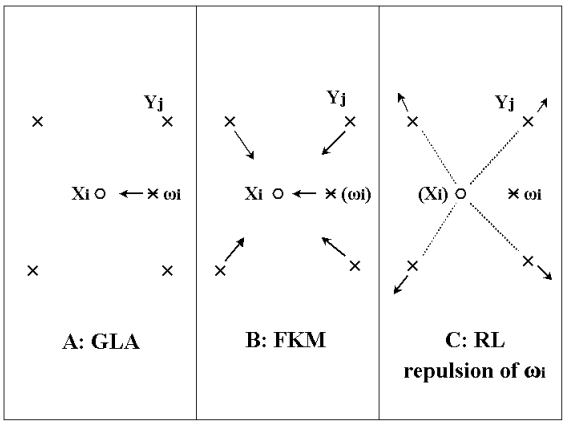

Fig. 1. Illustration of strategies of CKM, FKM and RL depicting the movement of the winning codevector $w_i(*)$ and other codevectors $y_j(\times)$ around the training vector $x_i(\circ)$.

2.3. Fuzzy Learning Vector Quantization

FVQ is an example of a fast-FKM algorithm. The detail of FVQ can be found in reference [2]. FLVQ is a combination of FVQ and LVQ, whose codebook update equation is grafted from the LVQ algorithm. The benefit is that then we can apply the VQ algorithm as an on-line adaptive algorithm. The update equation is define as:

$$y_j^{(v+1)} = y_j^{(v)} + \alpha^{(v)} \frac{\sum_{i=1}^{M} \mu_j(x_i)^m (x_i - y_j^{(v)})}{\sum_{i=1}^{M} \mu_j(x_i)^m} \quad (5)$$

where $\alpha^{(v)}$ is the *learning rate sequence (lrs)* at vth iteration. When $\alpha^{(v)} = 1/v$, FLVQ has the same resulting performance as FVQ. Note that eq (3) for GLA and FKM can be replaced by eq (5) when $\alpha^{(v)} = 1/v$.

3. STRATEGY FOR REINFORCEMENT LEARNING

To date, VQ techniques in image compression may be separated into two groups: GLA algorithms and FKM algorithms.

For GLA, each training vector x_i is only assigned to one winning codevector w_i. To be simple, a 2-dimensional illustration shown in Fig. 1(A). Only w_i will be moved towards x_i, as w_i is the nearest neighbour to x_i.

For FKM, x_i attracts all the codevectors. The more codevectors around, the higher the likelihood of x_i being represented well. This is shown in Fig. 1(B). The size of the movement of each codevector is inversely proportional to its Euclidean distance from the training vector x_i, i.e. the winning vector moves the most towards x_i.

Fig. 1(C) shows a third core. When a codevector becomes the winning codevector w_i for x_i, it repels all other codevectors while itself moving towards x_i. This new idea is named *Reinforcement Learning* (RL) in this paper. The size of the movement of each codevector affected by the winning codevector w_i is inversely proportional to its Euclidean distance from w_i.

As a further stage, FKM and RL are combined to create a new approach shown in Fig. 2. For an input x_i, the resulting movement of a certain codevector y_j is decided by the competition between

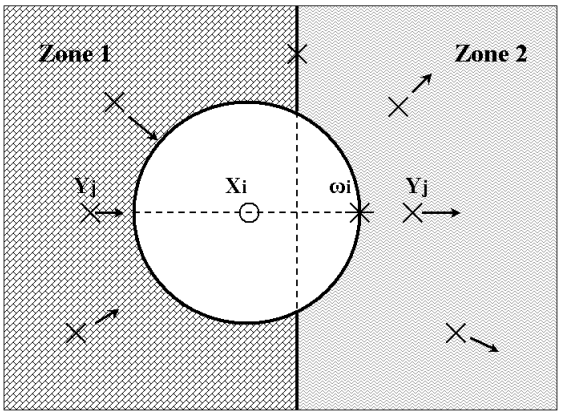

Fig. 2. Illustration of resulting movement of codevectors affected by a certain x_i and its corresponding w_i in a RL iteration.

the attraction of x_i and the repellent force of w_i. The measurement of the attraction and the repellent force are defined in the form of membership functions $\gamma_j(x_i)$ and $\eta_j(w_i)$ in eq (7) and eq (8), respectively. Thus, any codevector y_j in Zone 1 moves towards x_i, because $\gamma_j(x_i) > \eta_j(w_i)$, i.e. the attraction is greater than the repellent force. In contrast, all the codevectors in Zone 2 move away from x_i. Specially, the winning codevector w_i and the codevectors right at the edge between Zone 1 and Zone 2 are kept stationary.

Following this strategy, a new codebook update equation is introduced as follows

$$y_j^{(v+1)} = y_j^{(v)} + \alpha^{(v)} \frac{\sum_{i=1}^{M} \gamma_j(x_i)^m \left(\frac{\gamma_j(x_i)-\eta_j(w_i)}{\gamma_j(x_i)+\eta_j(w_i)}\right)\left(x_i - y_j^{(v)}\right)}{\sum_{i=1}^{M} \gamma_j(x_i)^m} \tag{6}$$

Here $\gamma_j(x_i)$ and $\eta_j(w_i)$ are defined as the membership functions of x_i and w_i, respectively.

$$\gamma_j(x_i) = \left\{ \sum_{p=1}^{N} \left[\frac{d_1(x_i, y_j)}{d_1(x_i, y_p)}\right]^{\lambda} + \sum_{p=1}^{N} \left[\frac{d_1(x_i, y_j)}{d_1(w_i, y_p)}\right]^{\lambda} \right\}^{-1} \tag{7}$$

$$\eta_j(w_i) = \left\{ \sum_{p=1}^{N} \left[\frac{d_1(w_i, y_j)}{d_1(x_i, y_p)}\right]^{\lambda} + \sum_{p=1}^{N} \left[\frac{d_1(w_i, y_j)}{d_1(w_i, y_p)}\right]^{\lambda} \right\}^{-1} \tag{8}$$

where w_i is the winning codevector of x_i. Note that when $y_p = w_i$, $d_1(w_i, y_p)$ is replaced by $d_1(x_i, y_p)$ in eq (7) and eq (8). In this case, since $\gamma_j(x_i) = \eta_j(w_i)$, RL does not affect the winning codevector.

When $\eta_j(w_i) = 0$, i.e. RL process is not considered, eq (6) is the same as eq (5). Thus FRLVQ transforms to FKM.

4. FUZZY REINFORCEMENT LEARNING VQ

In each FRLVQ iteration, a RL loop and a FKM loop are applied in order. The reason for step 4 is to avoid over-spreading of codevectors, during an update, in the multi-dimensional Euclidean space caused by using a high value of learning rate α in the RL loop. The FRLVQ algorithm is summarised in Table 2.

	Begin
	Select iteration times V (normally V=3)
	Select an initial codebook $Y^{(0)} = \{y_1, y_2, ..., y_N\}$
	$v = 0$
1	$v \leftarrow v + 1$
	$i = 0$
2	$i \leftarrow i + 1$
	Find w_i based on the nearest neighbour condition
	Evaluate $\gamma_j(x_i)$ using (7),$\forall j = 1, 2, ..., N$
	Evaluate $\eta_j(w_i)$ using (8),$\forall j = 1, 2, ..., N$
	if $i < M$, then go to step 2
3	Evaluate y_j using (6),$\forall j = 1, 2, ..., N$
4	Adjust y_j to be inside the numerical range required for the codebook, $\forall j = 1, 2, ..., N$
	$ii = 0$
5	$ii \leftarrow ii + 1$
	Evaluate $\mu_j(x_i)$ using (4),$\forall j = 1, 2, ..., N$
	if $ii < M$, then go to step 5
6	Evaluate $Y^{(v)} = \{y_1, y_2, ..., y_N\}$ using (3)
	If $v < V$, then go to step 1
7	Apply FVQ [2], using $Y^{(V)}$ as the initial codebook
	End

Table 2. FRLVQ algorithm. RL loop:step 2-4; FKM loop: step 5-6. Note that FVQ in step 7 can be replaced by other FKM algorithms.

5. EXPERIMENTAL RESULTS

The standard Lenna image of size 256×256 was used as an experimental training vector set. The pixels of this image take values between 0 to 255. The training vectors were obtained by dividing the Lenna image into 4096 blocks of size 4×4. Let X be the Lenna image set, which contains 4096 vectors in R^{16}. Let Y be the codebook, which contains 256 codevectors in R^{16}. Then the compression rate was 0.5 *bits per pixel (bpp)*. The resulting images were evaluated by the *peak signal to noise ratio* (PSNR), which is defined as

$$PSNR = 10 \log_{10} \frac{255^2}{\frac{1}{M^2} \sum_{i=1}^{M} \parallel x_i - w_i \parallel^2} \tag{9}$$

The overall performance of FRLVQ algorithm proposed in this paper was compared with that of GLA, FKM and FVQ algorithms on the basis of a criterion comprising their computational efficiency and the quality of codebook design.

Fig. 3 shows the PSNR as a function of the number of iterations v, when the same initial codebook was applied to the different VQ algorithms. $\alpha^{(v)}$ is an essential parameter in FRLVQ. Note that only 3 iterations are used in FRLVQ, i.e. $V = 3$, before applying step 6 in Table 2.

To investigate the impact of codebook design quality on the selection of the initial codebook, two alternative approaches were tested, firstly selecting first 256 vectors from the training vectors set, and secondly selecting vectors at random. These were then applied to the VQ algorithms. Results were shown in the first and second half of Table 3, respectively. All simulations were run on Sun Blade1000, CPU:600MHz, ULTRASPARC III using code written in MATLAB.

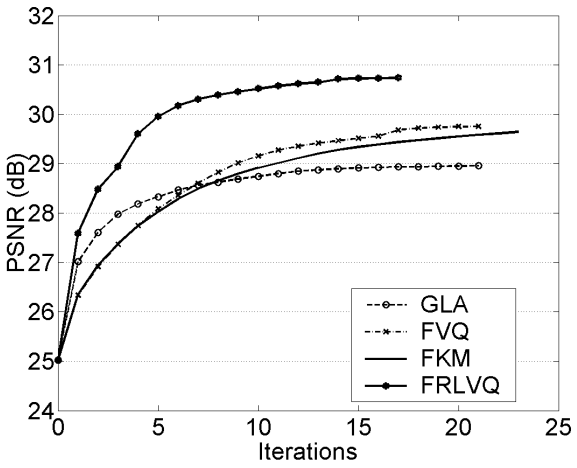

Fig. 3. Codebook design performances of GLA, FVQ, FKM and FRLVQ.applied to the Lenna image. The initial codebook selected by the first 256 training vectors. Parameters defined as: M=4096, N=256, L=16, m=1, $\varepsilon = 10^{-3}$, $\lambda = 10$ for all VQs; $\varepsilon' = 10^{-2}$ for FVQ (see reference [2]); V=3, $\alpha^{(v)} = \frac{100}{v}$ for FRLVQ.

Fig. 4. Lenna image reconstructed from the codebook designed by using FRLVQ algorithm. (M=4096, N=256, *bpp*=0.5, PSNR=30.82dB)

Algorithm	Iter.	$D^{\frac{1}{2}}$	PSNR (dB)	Time (min)	Initial Codebook
GLA	21	36.4	28.96	16	the first
FVQ [2]	21	33.1	29.75	27	256
FKM	**49**	**32.1**	**30.05**	**520**	training
FRLVQ	**17**	**29.6**	**30.74**	**84**	vectors
GLA	13	33.1	29.75	10	random
FVQ [2]	20	32.0	30.08	25	selection of
FKM	**35**	**32.1**	**30.05**	**435**	256 training
FRLVQ	**17**	**29.4**	**30.82**	**84**	vectors

Table 3. Performance comparison of different VQ algorithms applied to the Lenna image. Parameters values same as in Fig. 3.

lrs	Iter.	$D^{\frac{1}{2}}$	PSNR(dB)	Time(min)
$20/v$	20	30.7	30.42	87
50 $/v$	15	29.4	**30.82**	84
100 $/v$	18	29.4	**30.82**	85

Table 4. Performance of FRLVQ with different *lrs*. The 256 initial codevectors selected at random.

Table 3 shows two important results. Firstly, FRLVQ achieved the best quality of codebook design compared to the other VQ algorithms considered. Based on the selection of a random initial codebook, i.e. in the 2nd half of Table 3, a 0.77dB improvement of PSNR at 0.5bpp compression rate from FKM to FRLVQ is very significant, while 0.30dB improvement from GLA to FKM. This showed that FRLVQ is a new powerful approach to the globally optimum solution for VQ. Secondly, FRLVQ is insensitive to the randomness of the selection of the initial codebook. There was only 0.08dB difference between two selections of initial codebook for FRLVQ, while 0.79dB for GLA. There is, in addition, one further point to make. With regard to training time, FRLVQ requires only 84 minutes to converge, much shorter than the 435 minutes required for FKM. The finest reconstructed Lenna image was shown in Fig. 4.

The effect of the selection of α on the performance is shown in Table 4. It is clear that the algorithm is relatively robust with regard to the value of α. Performance of FRLVQ still exceeds that of FKM, the next best performer. Note that α is the only additional parameter introduced by FRLVQ.

The FRLVQ algorithm has been tested and found to work successfully with other standard images.

6. CONCLUSION

This letter has presented a new FRLVQ algorithm, which uses RL as a pre-processor before applying an FKM algorithm, for image compression and other applications. The simulation results have shown that a significant improvement in the quality of the resulting codebook is achieved by FRLVQ. Further investigations have indicated that FRLVQ is insensitive to the selection of the initial codebook and is relatively insensitive to the choice of the selection of the learning rate control parameter. It is now intended to develop and test FRLVQ as a general clustering algorithm.

7. REFERENCES

[1] A. Gersho, and R.M. Gray, *Vector Quantization and Signal Compression*, Boston, MA:Kluwer Academic, pp.362-372, 1992.

[2] N. B. Karayiannis, and P.-I Pai, "Fuzzy vector quantization algorithms and their application in image compression", IEEE Trans. on Image Processing, VOL. 4, NO.9:1193-1201, Sep. 1995.

[3] J. Jiang, "Image compression with neural networks - a survey", Signal Processing: Image Communication 14:737-760, 1999.

[4] T. Kohonen, *Self-Organization and Associative Memory*, Berlin, Germany:Spring-Verlag, 1989, 3rd ed.

[5] S.-G. Kong, and B. Kosko, "Differential competitive learning for centroid estimation and phoneme recognition", IEEE Trans. on Neural Networks, VOL.2, NO.1:118-124, Jan. 1991.

[6] D.-C. Park, "Centroid neural network for unsupervised competitive learning", IEEE Trans. on Neural Networks, VOL.11, NO.2:520-528, Mar. 2000.