On the Vector Quantization by Means of Recurrent Neural Networks

Ma, Ni
Department of Computer Science and Communication Engineering, Research Student

Nishi, Tetsuo
Department of Computer Science and Communication Engineering, Kyushu University

Wei, Gang
South China University of Technology

https://doi.org/10.15017/1498311

出版情報：九州大学大学院システム情報科学紀要. 3 (1), pp.1-6, 1997-12-22. 九州大学大学院システム情報科学研究院

九州大学大学院
システム情報科学研究科報告
第3巻 第1号 平成10年3月

Research Reports on Information Science and
Electrical Engineering of Kyushu University
Vol.3, No.1, March 1998

On the Vector Quantization by Means of Recurrent Neural Networks

Ni MA* , Tetsuo NISHI** and Gang WEI***

(Received December 22, 1997)

Abstract: A vector quantization scheme with a two-stage neural network coding(NNVQ) is developed, where an encoded vector is approximated by the output of the networks driven by a vector from an excitation codebook. The networks and the codebook are optimized to overcome some difficulties in conventional algorithms. The experimental results show that the NNVQ which employs the recurrent neural networks and the optimized learning algorithm performs the best among the reference versions of the VQ algorithms.

Keywords: Vector quantization, Recurrent neural networks, Learning, Optimization

1. Introduction

A vector quantization (abbr. as a VQ) is very important in the fields of signal compressing. The traditional VQ methods, i.e., the LBG algorithm and its improved algorithms [1)7)], are so-called batch clustering learning algorithms, which update codes after the presentation of all training data. They however have drawbacks such as large coding complexity and large storage requirement, especially in the cases of high coding rate or large vector dimension.

Recently some VQ techniques based on the neural networks have been proposed, e.g., Kohonen's self-organizing feature map [2)], single- or multi-layer neural networks as front ends for the quantization and so on [4)]. They can be regarded as generalized LBG algorithms.

To reduce computational efforts and storage demand, the codebook-excited neural network vector quantizer(abbr. as an NNVQ) was developed [5)6)]. In NNVQ the weighting parameters of neural networks are substituted in place of the conventional codeword set in the LBG so that both the storage requirement and the computational load decrease greatly due to the efficient parallel neural computing [3)]. More significantly, owing to the peculiar mechanism of the neural networks, the quantization centroid becomes a function of the partition of the inputs, but not an arithmetic mean as in the LBG algorithms, and thus can be well approximated by means of some efficient learning methods.

* Department of Computer Science and Communication Engineering, Research Student

** Department of Computer Science and Communication Engineering

*** South China University of Technology, P. R. China

The objective of this paper is to present a better neural network vector quantization system. We propose a new quantization scheme which utilizes a two-stage coding program. We use a recurrent neural network instead of single- or multi-layer feedforward network in order to improve the mapping characteristics. Another contribution of this paper is to apply the optimization learning method to NNVQ in order to improve its comprehensive performance. The simulation results show that our new scheme performs better than other VQ methods. That is, concerning the experiments with a Gauss-Markov source, its generalization performance is close to the corresponding theoretical bounds, while the results obtained by using speech signal manifest that it improved the quantization performance over conventional algorithms.

2. NNVQ System

The vector quantization(VQ) can be stated as follows: Given a real vector X consisting of k continuous samples, find a discrete code C that minimizes $d(X, \hat{X})$, where $\hat{X}$ denotes the recovered signal corresponding to C and $d(\cdot)$ is a given distortion measure, e.g., the Euclidean norm. The most important problem for it is how to determine the set of codes, i.e., the codebook.

Since the neural network contains many neuron parameters, we can expect that the assemble of the input signals can be modified so as to facilitate the classification. Hence we employ the neural networks in the fundamental coding systems to compose our NNVQ system.

2.1 Structure and Functions of NNVQ

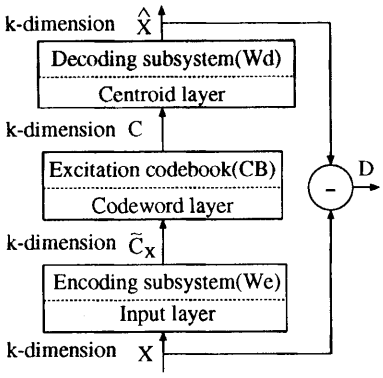

Fig.1 Codebook excited neural networks vector quantization system(NNVQ)

System

The overall architecture of the NNVQ system we propose in this paper is shown in **Fig.1**, which consists of an encoding subsystem with a weight set W_e, a decoding subsystem with a weight set W_d and an excitation codebook CB as the codeword layer. The encoding and decoding subsystems use the neural networks of the same type shown in Sec.2.2. Their parameters are determined by a learning process beginning with randomly initialized entries as will be explained in Sec.2.3. The initial excitation codebook is chosen from Gaussian time series and then is optimized by the same learning process.

Let all dimensions of the input signal X, the discrete mapping $\tilde{C}_X$ of X, the excitation vector C and the output signal $\hat{X}$ in **Fig.1** be k. That is, $X, \tilde{C}_X, C, \hat{X} \in \Re^k$, if we denote one-dimensional real space by $\Re$.

The encoding subsystem(i.e., encoding mapping denoted by Θ) can be described by:

$$\tilde{C}_X = \Theta(X, W_e) \in \Re^k, \tag{1}$$

and the decoding subsystem(i.e., decoding mapping denoted by Φ) can be described by:

$$\hat{X} = \Phi(C, W_d) \in \Re^k. \tag{2}$$

As shown in **Fig.1**, the decoding subsystem is driven by a codeword vector selected from the excitation codebook. Hence the coding procedure is that, for each input vector X, a vector $C_X^* \in \Re^k$ is chosen so as to minimize $d(\tilde{C}_X, C)$ over whole codebook. That is,

$$d(\tilde{C}_X, C_X^*) = \min_{C \in CB} d(\tilde{C}_X, C)$$

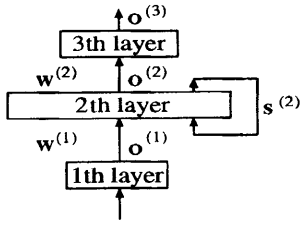

Fig.2 Neural structure in NNVQ system

$$= \min_{C \in CB} d(\Theta(X, W_e), C). \tag{3}$$

Thus the total coding distortion D over an input signal data set χ is defined as follows:

$$D = \sum_{X \in \chi} \min d(X, \hat{X})$$

$$= \sum_{X \in \chi} d(X, \Phi(C_X^*, W_d)). \tag{4}$$

We then define the partial distortion as follows:

$$D_c = \sum_{X \in \chi} d(\tilde{C}_X, C_X^*) = \sum_{X \in \chi} d(\Theta(X, W_e), C_X^*). \tag{5}$$

We first decide the neural structure in the NNVQ in Sec.2.2 and then introduce the learning and optimization algorithms for whole system in Sec.2.3.

2.2 Neural Structure in NNVQ System

To improve the mapping characteristic(Θ and Φ) in the NNVQ, we apply the generalized neural model, i.e., the recurrent neural network(abbr. as an RNN) to implement the maps in the encoding and decoding subsystems shown in **Fig.2**.

Considering the neural networks with three layers in **Fig.2**, let N_l be the number of neurons in the lth layer, where $1 \le l \le 3$ and $N_1 = N_3 = k$. At the iteration n, the outputs $o_i^{(2)}(n)(1 \le i \le N_2)$ of the second layer neurons are determined by:

$$o_i^{(2)}(n) = f\{\sum_{j=1}^{N_1} w_{ij}^{(2)} o_j^{(1)}(n)$$

$$+ \sum_{m=1}^{N_2} s_{im}^{(2)} o_m^{(2)}(n-1) + b_i^{(2)}(n)\}, \tag{6}$$

where $o_j^{(1)}(j = 1, 2, \cdots, N_1)$ are the outputs of the first layer neurons, $b_i^{(2)}$ are the threshold values of neurons in the second layer, $w_{ij}^{(2)}$ are the weights between the the second and the first layers, and $s_{im}^{(2)}$

are the strengths of the recurrent connections within the the second layer, with $1 \leq i, m \leq N_2$. Note that in **Fig.1** W_d and W_e denotes the total sets of the weighting parameters $\{w_{ij}\}$, $\{s_{im}\}$ and $\{b_i\}$ in each RNN.

If $\{s_{im}\}$ are all zero in Eq.(6), the network is a feedforward neural network(abbr. as a FNN). This model can be expected to have better mapping performance than the previous network using FNN [8]. We assume that the the output neurons have linear activation functions, while the activation functions of the other layer units have the type of $f(x) = (1 - e^{-2x})/(1 + e^{-2x})$.

2.3 Learning and Optimization Algorithms

In this section, we describe how to determine the excitation codebook CB and how to train the parameters W_e and W_d of two neural networks.

The excitation codebook CB obtained from Gaussian time series usually shows good performance but is still room to be improved by learning.

We first train Φ and CB together while fixing W_e and then we train Θ while using the trained W_d and CB. Both maps Θ and Φ can be learned by taking into account the unknown probability density function of the full data set through the weight updating.

Fig.3 shows the flowchart for training the map Φ and the codebook CB. We use the criterion D in Eq.(4) for training over the data set χ. The purpose of the above training is to get CB^* and W_d^* such as

$$D(\chi, CB^*, W_d{}^*) \leq D(\chi, CB, W_d). \tag{7}$$

The algorithm is as follows:

$$\Delta W_d = \eta D \nabla_{W_d} O_d, \tag{8}$$

and

$$\Delta C_X^* = \eta D \nabla_{C_X^*} O_d, \tag{9}$$

where Δ denotes the change of the training objective, ∇ the gradient with respect to its subscript, and O_d the set of the outputs $o_i^{(3)} (i = 1, 2, \cdots N_3)$ of the last layer neurons , which belongs to the decoding networks(correspondingly we define O_e those in the encoding networks), and η the corresponding learning rate prescribed as a small positive value.

Training Θ is almost the same as training Φ except for W_d, D, and O_d replaced by W_e, D_c, and

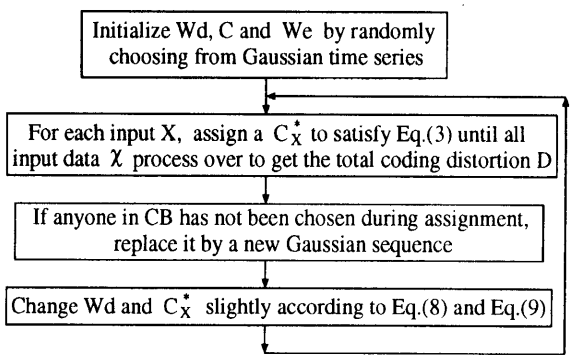

Fig.3 Flowchart for training map Φ and codebook CB

O_e, respectively. That is, we obtain W_e^* such that

$$D_c(\chi, W_e{}^*) \leq D_c(\chi, W_e), \tag{10}$$

by training over χ.

The steepest gradient algorithm is often trapped with local minima. In order to get the optimization design system we employed the simulated annealing(SA) algorithm [10] in the learning procedure, regardless of the additional computation used. Because the NNVQ system is off-line trained, its computational load in the design phase is a secondary concern. Here we use the following temperature schedule [10]: $T_n = \sigma^2(1 - n/\aleph)^3$, as the noise added to the parameters W_d, W_e or C at the nth training iteration, where $\aleph$ defines the number of the training iterations and the variance σ^2 of the training data is set to the initial temperature.

After acquiring the optimized W_e^*, W_d^* and CB^* by above learning, we construct a complete NNVQ system. Hence it is no doubt that the computational efforts in on-line quantization are very small.

3. Experimental Results

3.1 Simulation Results and Discussion

We usually use the Gauss-Markov source to train and evaluate our NNVQ system. That is, the signal $\{x_n\}$ is generated by a difference equation: $x_{n+1} = ax_n + G_n$, where $\{G_n\}$ is a zero mean unit variance independent identical distribution Gaussian time series. Here we set $a = 0.9$, which makes the source to be a highly correlated one such as the speech signals.

For the networks in the NNVQ, the number of the input and output units is a priori decided by the dimensions of the input signal and the excitation vector, as described in Sec.2.1, while the number N_2 of the hidden neurons can be chosen experimentally.

First we compared the performance between our

Table 1 Performance comparisons between NNVQ and LBG algorithm

Coding rate R (bits/vector)	*SNR(dB)*					
	FSVQ LBG	FNNVQ	RNNVQ	SA LBG	SA FNNVQ	SA RNNVQ
1	3.178	3.156	3.177	3.623	3.932	4.641
2	5.915	5.813	5.911	6.762	6.928	7.657
3	8.178	8.101	8.171	8.971	9.065	9.985
4	8.996	8.870	9.001	10.048	10.163	11.258

Table 2 Performance comparisons of NNVQ systems with different structures(coding rate R=1bit/sample)

N_1	N_2	N_3	*SNR(dB) for NNVQ systems*							
			FNNs, UCB		FNNs, TCB		RNNs, UCB		RNNs, TCB	
			Train	Test	Train	Test	Train	Test	Train	Test
4	1	4	8.657	8.399	8.804	8.534	9.122	9.064	9.255	9.208
4	2	4	10.467	10.167	10.819	10.541	11.508	11.399	11.857	11.643
4	4	4	10.478	10.171	10.828	10.517	11.512	11.401	11.857	11.645
4	6	4	10.577	10.279	10.926	10.625	11.609	11.413	11.955	11.766
4	8	4	10.456	10.161	10.818	10.548	11.509	11.359	11.849	11.652

NNVQ and the traditional LBG algorithm under the same conditions. The NNVQ and the LBG are constructed for the input vector dimension $k = 4$ with the coding rate $R = 1, 2, 3$, and 4 bits per input vector, and the number of the hidden units $N_2 = 2$.

Table 1 summarizes the above comparisons of the resulting signal to noise ratios(SNRs measured by total energy of the input signal over that of the total coding distortion in decibels) of the test data, where the excitation codebook CB in the NNVQ system is untrained. In **Table 1**, the words "FNNVQ" and "RNNVQ" mean the NNVQ schemes by means of the feedforward and recurrent networks respectively, the prefix "SA" denotes that the SA algorithm is used in the training process of this scheme, and the prefix "FSVQ" means that the direct "full-search" vector quantization algorithm is employed in the LBG scheme [1].

From **Table 1**, it can be concluded that the NNVQ using both RNNs and the SA algorithm performs the best. Note that without SA, the RNN scheme improves little over the FNN and LBG ones, and even performs worse than the optimized LBG algorithm. This shows that the optimization in VQ is very important, because any VQ scheme is greatly influenced by the initial values. Note also that the optimized RNNVQ indeed performs better than the FSVQ and the optimized LBG algorithm.

Then we compared the performance among the NNVQs by varying some configurations: (a) the number of the hidden units N_2; (b) whether the excitation codebook being trained or not; (c) the network structure being recurrent or feedforward, while we fixed the coding rate $R = 4$ bits per input vector and the dimension $k = 4$ of the input signal. The comparisons of the SNRs are shown in **Table 2**, where the words "UCB" and "TCB" are corresponding to the untrained codebook and the trained codebook respectively.

It is obvious from **Table 2** that the performance of the NNVQ system using RNNs and the trained codebook is the best again. In fact this is the most encouraging quantization performance of the RNNVQ because its generalization performance is very near to the design performance. Such a conclusion can be further verified by **Fig.4**, where some theoretical and simulation performance is obtained for various dimensions of the input vector.

In **Fig.4**, the vertical axis represents the SNRs of the NNVQ results, while the horizontal axis denotes the dimension of the input vector, and the coding rate R is fixed at 1 bit per input sample. The theoretical performance 13.2dB of the Shannon lower bound D^* to the rate-distortion function in this figure is given by [11]:

$$D^* = \frac{1}{2\pi e} e^{-2(R-\hbar)}, \tag{11}$$

where $\hbar \equiv \frac{1}{2}\log(2\pi e\sigma^2)$ is the differential entropy rate of the source data we used and σ is defined in Sec.2.3, while the asymptotically optimal·VQ bound $D_k(R)$ is denoted by [12]:

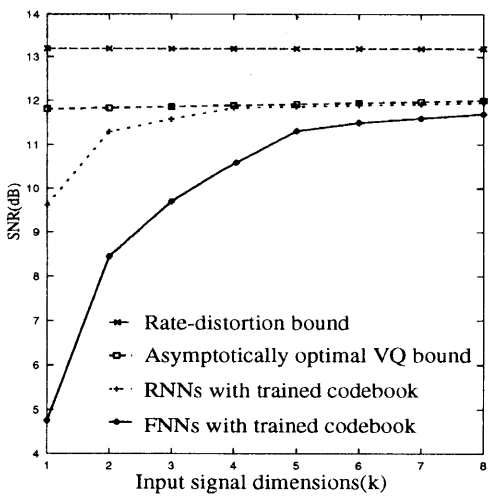

Fig.4 Comparison between quantization performance of NNVQ and corresponding theoretical bounds (coding rate R=1bit/sample)

$$D_k(R) = \frac{e\Gamma(1+k/2)^{2/k}}{1+k/2} D_\infty(R), \tag{12}$$

where $\Gamma(x)$ is the Gamma function and $D_\infty(R)$ is the theoretical value of the optimal VQ.

Fig.4 clearly shows that the performance of our NNVQ is very close to $D_k(R)$, especially when $k \geq 5$.

There exist some other important conclusions as follows:

The so-called "empty cell" problem in conventional VQ systems is caused by a common factor that some codewords are rarely chosen due to the poor initial guesses. Obviously optimizing the excitation codebook during the training can avoid this situation to a great extent, especially in the quantizer with a large excitation codebook. **Table 2** denotes that the schemes using the optimized codebook give the improved results over those with the untrained codebook.

The results in **Table 2** also indicate that in all given number N_2 of the hidden units, the differences of the SNRs between the training and test data for the RNN schemes are smaller, which, from another aspect, proves that the NNVQ using RNNs is a better scheme. In addition **Table 2** shows that $N_2 = 2$ yields a excellent trade-off between the accuracy and the computational efforts for our NNVQ system.

3.2 An Example of Practical Application

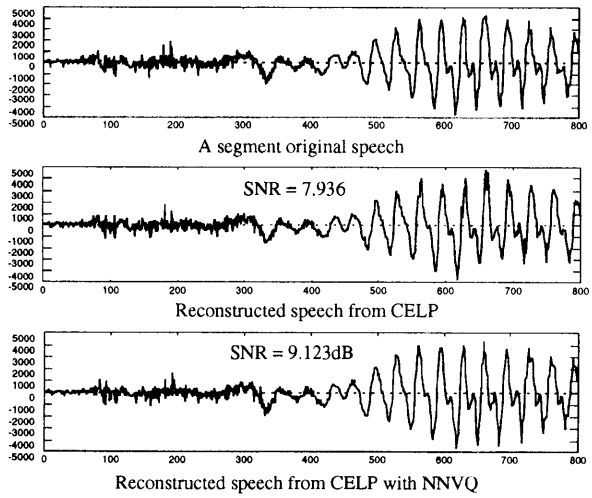

Fig.5 Comparisons of CELP coding between conventional VQ and NNVQ

Table 3 Comparisons of distortion between CELP speech coding and CELP speech coding with NNVQ(NCELP)

R bits/ vector	*SNR(dB)* Training set CELP	Training set NCELP	Test set CELP	Test set NCELP
2	3.164	4.707	-0.658	4.389
4	5.904	6.271	5.323	6.133
6	7.684	8.398	7.509	8.226
8	8.937	9.326	8.692	9.228

tion

As an application of our scheme to speech coding, the Gaussian excitation codebook in the CELP speech coder [9] is replaced by an RNNVQ system. Some simulation results are shown in **Fig.5** and **Table 3**, where the SNRs are measured by the total energy of the original speech signal over that of the difference between the reconstructed and original signal. In simulation both the training and test data consist of 20 sentences from a set of English speech database [13]. In the CELP system we used here, the order of the short- and long-term linear predictive filter are 12 and 3 respectively, the short- and long-term frame length is 160 samples(20ms) and 40 samples(5ms) respectively, and the dimension of the excitation vector is 20.

The results also show that the RNNVQ performs better than the conventional VQ at a high coding vector dimension.

4. Conclusion

Based on a quasi-parallel neural computing approach, we have developed a codebook-excited re-

current neural network vector quantizer. The RN-NVQ system has one notable advantage that the correlation information among the sources can easily be taken into account by the recurrent connections. Experimental results showed that the optimization in the learning procedure could improve greatly the performance. Furthermore, due to the powerful mapping capacity, the codebook-excited recurrent neural networks can be applied to design vector quantizer with any required structural form on its code vectors.

References

1) R. M. Gray, "Vector quantization," in *IEEE ASSP magazine*, April 1984, pp.4-29.
2) T. Kohonen, "Improved versions of learning vector quantization," in *Proceedings of 1990 IEEE International Joint Conference on Neural Networks*, **Vol.1**, 1990, pp.545-550
3) H. W. Frank, K. K. Parhi and K. Ganesan, "Neural network vector quantizer design using sequential and parallel learning techniques," in *IEEE ICASSP'91*, 1991, 637-640.
4) T. Thyssen and S. D. Hansen, "Using neural networks for vector quantization in low rate speech coders," *ICASSP'93*, 1993, II-431-434
5) Z. Wang and J.V.Hanson, "Code-excited neural vector quantization," *ICASSP'93*, 1993, I-497-500.
6) C. Q. Chen, "A modified Generalized Lloyd algorithm for VQ codebook design," in *ICASSP'96*, 1996, pp.542-544.
7) A. Das, V. Rao and A. Gersho, "Variable-Dimension vector quantization," *IEEE Signal Processing Letters*, **Vol.3, No.7**, July 1997, pp.200-202.
8) A. C. Tsoi and A. D. Back, "Locally recurrent globally feedforward networks: acritical review of architecture," *IEEE Transactions on neural networks*, **Vol.5, No.2**, March 1994, pp.229-233.
9) M. Shroeder and B. Atal, "Code-excited linear prediction(CELP): high quality speech in very low rates," in *IEEE ICASSP'85*, Tamapa, 1985, pp.937-940.
10) K. Zegar, J. Vaisey and A. Gersho, "Global optimal vector quantizer design by stochastic relaxation," *IEEE Transactions on Signal Processing*, **Vo.40, No.2**, February 1992, pp.310-322.
11) J. Makhoul and S. Roucos, "Vector quantization in speech coding," *Proceedings of the IEEE*, **Vol.73, No.11**, November 1985, pp.1551-1558.
12) Y. Yamada, Tazaki and R. Gray, "Asymptotic performance of block quantizers with different distortion measures," *IEEE Transactions on Information Theory*, **Vol.IT-26, No.1**, January 1980, pp.6-14.
13) Co. Ltd. International Electronics and Communication Basic Technology Institute, "ATR British English speech data base," 1996.