Int. J. Electron. Commun. (AEÜ) 72 (2017) 178–183

Contents lists available at ScienceDirect

International Journal of Electronics and Communications (AEÜ)

journal homepage: www.elsevier.com/locate/aeue

Regular paper

An improved vector quantization method using deep neural network

Wenbin Jiang [a,*], Peilin Liu [a], Fei Wen [a,b]

[a] Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai 200240, PR China
[b] Air Control and Navigation Institution, Air Force Engineering University, Xian 710000, China

ARTICLE INFO

Article history:
Received 19 March 2016
Accepted 5 December 2016

Keywords:
Deep neural network
Vector quantization
Auto-encoder
Binary coding

ABSTRACT

To address the challenging problem of vector quantization (VQ) for high dimensional vector using large coding bits, this work proposes a novel deep neural network (DNN) based VQ method. This method uses a *k*-means based vector quantizer as an encoder and a DNN as a decoder. The decoder is initialized by the decoder network of deep auto-encoder, fed with the codes provided by the *k*-means based vector quantizer, and trained to minimize the coding error of VQ system. Experiments on speech spectrogram coding demonstrate that, compared with the *k*-means based method and a recently introduced DNN-based method, the proposed method significantly reduces the coding error. Furthermore, in the experiments of coding multi-frame speech spectrogram, the proposed method achieves about 11% relative gain over the *k*-means based method in terms of segmental signal to noise ratio (SegSNR).

1. Introduction

Vector quantization (VQ) is a fundamental technique for data compression, such as video coding and audio coding. In traditional VQ methods, the *k*-means or Linde-Buzo-Gray (LBG) algorithm [1,2] is most commonly used in codebook training (clustering). However, when it comes to large vector dimensions and codebook sizes, direct use of the VQ method suffers from a serious complexity barrier. Some constrained VQ methods, such as Partitioned VQ, are commonly used to reduce storage and computation complexity [3,4]. Unfortunately, these compromised methods may severely increase the coding error.

Recently, inspired by the success of deep neural network (DNN) in data dimensionality reduction [5,6], DNN-based approaches have been developed to address this problem [7–9]. In [7], a deep auto-encoder (DAE) with a binary coding layer was learned to code the high-dimensional vector. In speech spectrogram coding, this method showed a considerable performance gain over traditional VQ technology. Nevertheless, when many of the activations of the coding units are far from binary, quantifying them to binary values may cause large distortions. In order to make the activations of the coding layer as close to binary as possible, an effective approach is to add Gaussian noise to the input of the coding layer [8]. Another approach is to force the coding layer to be binary during the forward pass in the fine-tuning [9]. All the above works were aimed at getting a binary coding layer from the real-valued activations of a DNN. In principle, quantifying a floating-point value to a single bit would inevitably cause distortion.

More recently, in [10], the authors utilized the traditional VQ method (*k*-means) as an initializer to learn a DNN-based vector quantizer for acoustic information retrieval. The output of the vector quantizer is the codeword label obtained by the traditional VQ method. The output layer of the neural network is a softmax layer whose node number is the same as the codeword number. In fact, this architecture is designed to learn speech content information from the initializer. However, as mentioned by the authors, the frame accuracy is not high (below 50%) for the training and development set. Thus, this architecture is unsuitable for data compression applications. Moreover, it is generally impractical to implement a VQ system with such an architecture when the number of coding bits (N) is large, since the number of the softmax output layers nodes, which is equal to the codeword number, is 2^N in this case.

This work proposes a novel DNN-based VQ method to achieve improved performance for quantizing high dimensional vector with a large-size codebook. Firstly, we learn a DAE using greedy layer-wise pre-training and back-propagation fine-tuning methods. Then, a DNN, which is initialized by the decoder network of DAE and fed with the codes obtained by the traditional VQ method, is trained as the VQ decoder. Unlike the DNN architectures using binary coding layer in [7] and binary output layer in [10], the input data of the proposed DNN architecture is binary. From the view of VQ system, the method in [7] learns both an encoder and a

* Corresponding author.
E-mail addresses: jwb361@sjtu.edu.cn (W. Jiang), liupeilin@sjtu.edu.cn (P. Liu), wenfei@sjtu.edu.cn (F. Wen).

http://dx.doi.org/10.1016/j.aeue.2016.12.002
1434-8411/

decoder, and the method in [10] learns an encoder, whereas our method tries to learn a decoder. The main advantage of the proposed method over the DAE is that it avoids the distortions induced by the binary coding layer. Moreover, utilizing the strong representation power of DNN framework, it has the capability to reduce the coding error of the traditional VQ system. Experiments on speech spectrogram quantization have been conducted to evaluate the performance of the new method in comparison with several representative methods. The results showed that the new method has smaller distortions compared with the traditional *k*-means based method and the recently proposed DNN-based method.

The rest of this paper is organized as follows. In Section 2, we briefly review the *k*-means based VQ method. Section 3 introduces the details of the proposed DNN-based VQ method. Section 4 presents an implementation of the new VQ method for speech signal. The experimental results are given in Section 5. Finally, we summarize our work in Section 6.

2. Review of vector quantization based on *k*-means

Traditional VQ methods commonly employ the *k*-means or LBG algorithm for codebook training. In the training stage, a standard *k*-means algorithm is used to find the cluster centroids which are called codewords. In the coding stage, the encoder finds the nearest centroid for a new vector and transmits its index to the decoder, based on which the decoder retrieves the corresponding codeword in the codebook. The overall operation of VQ can be regarded as composition of two operations

$$Q(X) = D(I) = D(E(X)) \tag{1}$$

where $D(\cdot)$ denotes a decoder, $E(\cdot)$ denotes an encoder, and I is the index (i.e. code). The distortion is defined as

$$d = \|X - \widehat{X}\|^2 \tag{2}$$

where $\widehat{X} = Q(X)$ is the quantized vector.

It is well known that the *k*-means optimization problem is NP-hard in general [11]. Moreover, for training high dimensional vector with large size of clusters, the *k*-means algorithm suffers from large memory consumption and slow convergence speed. Thus, the largest codebook sizes used typically range from 2^{10} to 2^{12} and the largest vector dimensions used are typically from 40 to 60. To break through this limitation, some constrained VQ methods, such as Partitioned VQ, are commonly used. The Partitioned VQ strategy partitions a high dimensional vector into two or more subvectors, and then, training and coding each subvector individually. However, this strategy may severely degrade the performance when there is substantial statistical interdependence between different subvectors. More recently, many strategies have been proposed to speed up the *k*-means algorithm, such as using triangle mini-batch optimization [12], utilizing graphics processor units (GPUs) [13], and seeding carefully [14]. But these methods are still impractical and inefficient when both the vector dimension and cluster number are large.

This paper focuses on VQ of high dimensional vector using large coding bits and its application on speech signal compression. Firstly, we quantize 121-dimensional speech power spectra using 54-bit. Due to the impracticality of applying the standard VQ method, the spectrum vector is partitioned into four subvectors with dimensions of 30, 30, 30, and 31, and these subvectors are quantized with bits of 10, 9, 9, and 8, respectively. This quantizer is used to exploit the intra-frame correlation of the speech spectrum. Then, we quantize N frames speech power spectra using $9 * N$ bits, and each speech frame is quantized with 9-bit applying standard VQ. This quantizer is used to exploit the inter-frame correlation. These vector quantizers are constituents of the proposed DNN-based VQ method, and are also baselines for the evaluation.

3. Proposed DNN-based vector quantization framework

In this section, firstly, we introduce the framework of the proposed DNN-based VQ method. Then, the training procedure of the deep auto-encoder is described. Finally, a DNN-based decoder is presented in details. In the following, we call the proposed method DNN-VQ for short.

3.1. Framework of the DNN-VQ

Fig. 1 illustrates the framework of the proposed DNN-VQ system, which employs a traditional VQ quantizer as the encoder and a DNN as the decoder. The codebook in the encoder is trained with the *k*-means algorithm discussed in Section 2. The codeword index I (i.e., code) for each vector X is obtained by the standard VQ process. The DNN in the decoder is initialized with a DAE which will be introduced in the following section. The input of the DNN is the code I obtained from the encoder, and the expected output is exactly the input vector X.

The proposed DNN-VQ system utilizes an encoder function $E(\cdot)$ in (1) to map the input vector X to code I, and uses a DNN-based non-linear function $F(\cdot)$ to perform the reverse mapping. The distortion of the DNN-VQ system is $\|X - F(E(X))\|^2$, which is also the cost function used in the DNN training. The main advantage of the proposed DNN-VQ system over the traditional *k*-means based VQ system is as follows. The traditional decoder just retrieves the vector using a codeword from the trained codebook, while the DNN-based decoder performs a sophisticated feed-forward pass to reconstruct the vector. The DNN can extract essential features of the training data, and the code I makes sure that the coding layer of DNN is binary. Using the combined strategy, DNN-VQ has the capability to reduce the coding error of the VQ system.

3.2. Deep auto-encoder

In this work, we use a stack of auto-encoders to learn each layer of the DNN rather than restricted Boltzmann machine (RBM) used in [7–9]. Compared with RBM, the auto-encoder is easier to train and can be used to obtain any parametric layer [15].

An auto-encoder firstly maps an input vector $\mathbf{x}$ to a hidden representation $\mathbf{y}$ using a non-linear mapping function $f_\theta(\mathbf{x}) = s(\mathbf{W}\mathbf{x} + \mathbf{b})$, parameterized by $\theta = \{\mathbf{W}, \mathbf{b}\}$. $\mathbf{W}$ is a weight matrix, $\mathbf{b}$ is a bias vector, and $s(x) = \frac{1}{1+e^{(-x)}}$ is an active function. Then, the resulting hidden representation $\mathbf{y}$ is mapped back to a reconstructed vector $\mathbf{z}$ using a reverse function $f'_{\theta'}(\mathbf{y})$, with $\theta' = \{\mathbf{W}^{\mathbf{T}}, \mathbf{b}'\}$. The parameters of this model are optimized by minimizing the average reconstruction error over the training set

$$\theta^*, \theta'^* = \arg\min_{\theta,\theta'} \frac{1}{n}\sum_{i=1}^{n} L(\mathbf{x}^{(i)}, \mathbf{z}^{(i)}) \tag{3}$$

where $\mathbf{z}^{(i)} = f'_{\theta'}(f_\theta(\mathbf{x}^{(i)}))$, n is the size of the training set and L is the squared error loss function

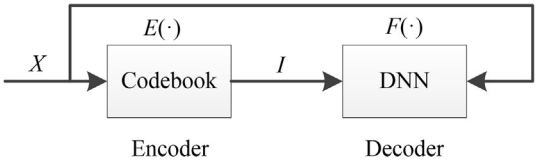

Fig. 1. Framework of the proposed DNN-VQ system.

$$L(\mathbf{x},\mathbf{z}) = \|\mathbf{x}-\mathbf{z}\|^2 \quad (4)$$

A deep auto-encoder (DAE) is a deep neural network that is built by a stack of auto-encoders, where the output of each layer is wired to the inputs of the successive layer [16]. A popular approach to train the stacked auto-encoders is to conduct greedy layer-wise training, which is demonstrated in Fig. 2. The training procedure firstly train an auto-encoder to minimize square error (i.e., E in Fig. 2) of reconstructed error of the raw input, and then iteratively train the successive layer using the hidden layers' outputs (i.e., *code* in Fig. 2) of previous auto-encoder. After that, unfold all the auto-encoders into a DAE and initialize its parameters. Finally, fine-tune all the parameters using global reconstruction error.

In summary, the training procedure is as follows.

(1) Train the first layer as an auto-encoder using the raw input data.
(2) Iteratively train the successive layer using the hidden units' outputs of previous auto-encoder.
(3) Unfold all the auto-encoders into a DAE, and fine-tune the DAE using back-propagation to make its output as similar as possible to its input.

Fig. 3 illustrates an example for learning a stack of auto-encoders and fine-tuning the deep auto-encoder. In the left, there are four auto-encoders, with input-hidden layer size 121-2048, 2048-2048, 2048-2048, 2048-36, respectively. Thus, the unfolded DAE has a 121-dimensional input/output layer and seven hidden layers. In the right, the network 121-2048-2048-2048-36 (denoted by network layer's size) is encoder, and the reverse network is decoder. The middle hidden layer is called coding layer, with 36 hidden units.

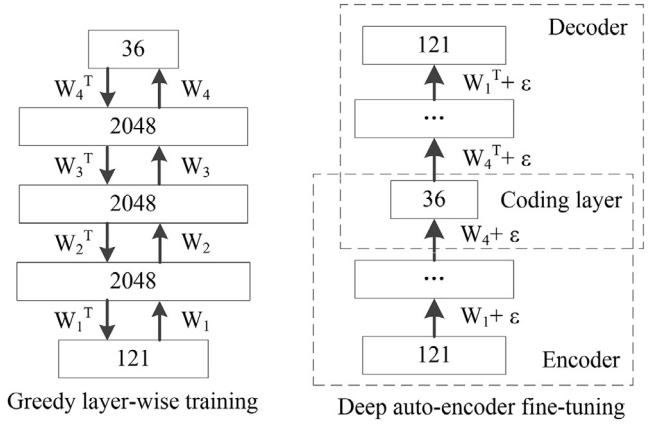

Fig. 3. Left: illustration of learning a stack of auto-encoders. Right: description of the fine-tuning procedure of deep auto-encoder.

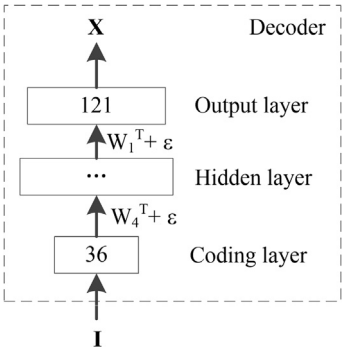

Fig. 4. Training procedure of the DNN-based decoder.

3.3. The DNN-based decoder for vector quantization

For the fine-tuned DAE, we can quantize the output of the coding layer to either zero or one with a threshold (e.g., 0.5). These quantized codes are identical to the VQ codes. In the case that most of the outputs of the coding layer are far from binary, we can add Gaussian noise to the input of the coding layer or force the output of the coding layer to be binary to make the distribution of coding layer closer to binary. Nevertheless, quantizing a floating-point value to a single bit would inevitably cause more or less distortion. If the input of decoder network itself is binary, this distortion can be avoided.

Fig. 4 illustrates the diagram of the proposed DNN-based decoder network for VQ. The network architecture is the same with the decoder of DAE in Fig. 3. That is, the network architecture is 36-2048-2048-2048-121 (denoted by network layer's size). The weights of the network are initialized by the decoder network of fine-tuned DAE. We are convinced that the fine-tuned DAE has a strong capability to represent the input data with the real-valued coding layer. In Fig. 4, the input of the DNN-based decoder is the codeword index (i.e., I in (1)) rather than the output of the encoder network of DAE, and the expected output is the input data of the VQ system (i.e., X). Compared with those relative DNN's structures [8–10], this structure ensures that the input value of the decoder network is exactly binary.

Theoretically, the DNN in Fig. 4 is expected to learn a non-linear function F that maps the codeword I to the raw input data $X, F: I \rightarrow X$. The learned function is used to replace $D(\cdot)$ in (1) to reduce distortion of the traditional VQ system. The objective function used for training is $\|F(I)-X\|^2$, which is equal to the distortion in (2). From another point, the DNN has the capability to capture essential features of the training data, such as phonological features of speech signal. This makes sure the DNN-based decoder can reconstruct feature-based data rather than retrieves codeword from codebook.

4. Vector quantization of speech spectrogram

The proposed DNN-VQ system can be applied to compression of various kinds of high-dimensional data, such as image, video and audio. As an example, this section presents an implementation of the new VQ method for speech signal in the frequency domain. It should be noticed that we only quantize the speech magnitude spectra.

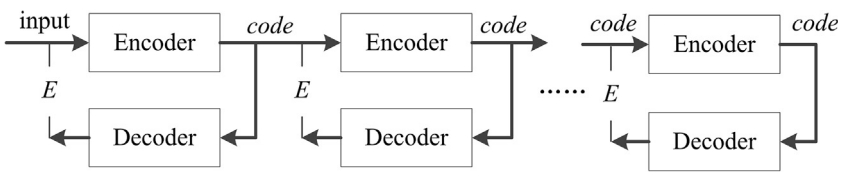

Fig. 2. Illustration of greedy layer-wise training.

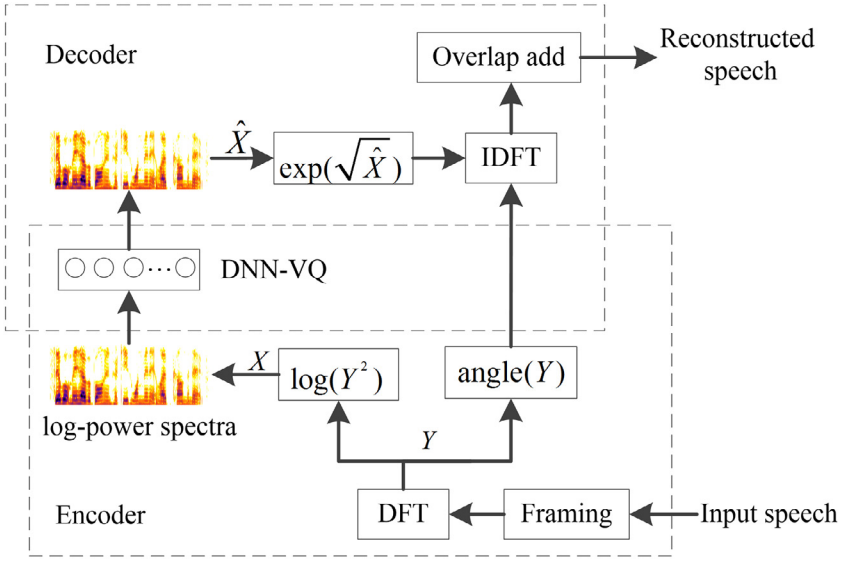

Fig. 5. Diagram of VQ for speech spectrogram using DNN-VQ system.

Table 1
Bit allocation of partitioned VQ for 121-dimensional vector using the *k*-means algorithm.

Total	Bit allocation for each partition			
121	1–30	31–60	61–90	91–121
36-bit	10-bit	9-bit	9-bit	8-bit

Table 2
Comparison of average distortions (squared Euclidean norm) on the training set, validation set, and test set.

VQ system	Training set	Validation set	Test set
DNN-AN	0.211	0.345	0.347
DNN-FB	0.404	0.406	0.406
DNN-VQ	**0.210**	**0.212**	**0.212**
k-means-VQ (baseline)	0.232	0.232	0.232

The diagram of VQ for speech spectrogram is shown in Fig. 5. In the encoder, the speech signal is framed and transformed into the frequency domain. The log-power magnitude spectra are normalized and coded by the trained DNN-VQ system. In the decoder, the log-power magnitude spectra are decoded by the DNN-VQ system. Subsequently, a de-normalization process is used to obtain the actual spectrogram. Then, an inverse transform is performed to obtain the time-domain signal. Finally, an overlap-add method is utilized to synthesize the waveform of the speech signal.

We use log-spectral distortion (LSD) to measure the spectra coding error. The LSD between spectra $P(\omega)$ and $\widehat{P}(\omega)$ is defined as

$$D_{LS} = \sqrt{\frac{1}{2\pi}\int_{-\pi}^{\pi}\left[10\log_{10}\frac{P(\omega)}{\widehat{P}(\omega)}\right]^2 d\omega} \tag{5}$$

For the reconstructed speech signal, we use segmental signal to noise ratio (SegSNR) to assess the objective speech quality

$$SNR_{seg} = \frac{10}{M}\sum_{m=0}^{M-1}\log_{10}\frac{\sum_{n=N_m}^{N_m+N+1}x^2(n)}{\sum_{n=N_m}^{N_m+N+1}(x(n)-\hat{x}(n))^2} \tag{6}$$

where $x(n)$ is the input signal, $\hat{x}(n)$ is the reconstructed signal, N is the frame length, M is the number of frames in the signal, and N_m is the start index of mth frame.

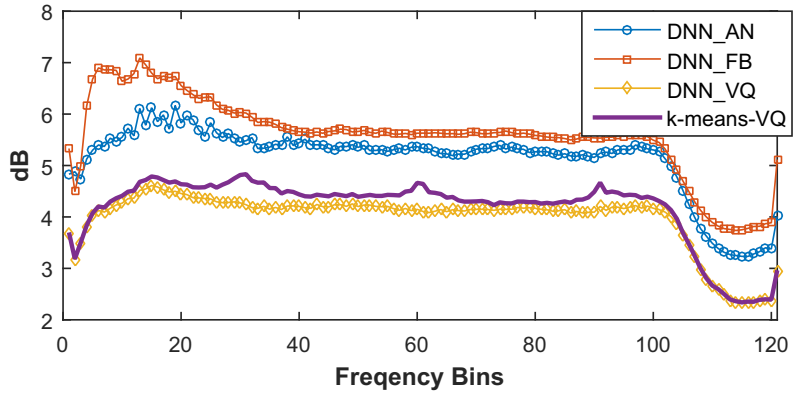

Fig. 6. Average LSD versus frequency bins on the test set.

Table 3
Average LSD (dB), SegSNR (dB) and PESQ results of different VQ methods on the test set.

VQ system	LSD	SegSNR	PESQ
DNN-AN	6.64	5.07	2.60
DNN-FB	7.15	4.11	2.30
DNN-VQ	**5.19**	**7.35**	**3.25**
k-means-VQ (baseline)	5.44	6.84	3.08

In addition, perceptual evaluation of speech quality (PESQ), which has a high correlation with subjective score [17], is also used to evaluate the quality of the reconstructed speech signal. More details of the evaluation settings will be introduced in the following section.

5. Evaluation

In this paper, we use Resource Management corpus [18] for evaluation. There are totally 36,608 utterances (about 29.5 h) in this corpus. We randomly pick 32,608 utterances as training set, 2000 utterances as validation set, and the remainder 2000 utterances as test set. All these speech data are down-sampled to 8 kHz and framed to 240 samples with hamming window. The frame shift is 120 samples (50% overlapped). A short-time Fourier analysis is used to compute the discrete Fourier transform (DFT) of each overlapped frame. Then, the 121-dimensional log-power spectra are used to train the neural network.

The settings for the DNN training are as follows: the datasets are divided into small "mini-batches" of 128 cases. In the pre-training of each stacked auto-encoder, the momentum is set to 0.5, the learning rate is 0.05, and the number of epoch is 20. In the fine-tuning of the DAE, the momentum is set to 0.9, and the initial learning rate is set to 0.1. The learning rate is gradually reduced by a factor of 0.9 when the decrease of the validation error between two consecutive epochs is less than 0.02%. The training process is stopped when the validation error decrease is less than 0.01%. The training of the DNN-based decoder follows the same setting.

5.1. Evaluation of the proposed DNN-VQ system

The baseline of this evaluation is a 36-bit vector quantizer that is based on *k*-means (denoted by *k*-means-VQ). Since it is unfeasible to implement the *k*-means method with 2^{36} clusters for 121-dimensional vector, the Partitioned VQ strategy is adopted. The 121-dimensional vector is partitioned into four sub-vectors, and each partition is allocated with a few of bits. The details of vector partition and bit allocation are shown in Table 1. The sub-vector 1–30 which contains relatively more speech information is allocated with more bits, and the sub-vector 91–121 which contains relatively less speech information is allocated with fewer bits. As there are totally more than 6 million frames in the training set, we randomly pick 0.1 million frames for the *k*-means clustering. This VQ system is implemented not only for comparison but also for training the proposed DNN-based VQ system (denoted by DNN-VQ).

A DAE with binary coding layer is also trained for comparison. Two methods are used to make the coding layer closer to binary: adding Gaussian noise to the input of the coding layer (denoted by DNN-AN), forcing the output of the coding layer to be binary (denoted by DNN-FB). The DAE architecture is 121-2048-2048-20 48-36-2048-2048-2048-121, which is illustrated in Fig. 3. The decoder network architecture is 36-2048-2048-2048-121, which is illustrated in Fig. 4. For the DNN-AN system, the mean of the added Gaussian noise is set to zero, and the standard deviation is chosen via cross-validation.

The average distortions (i.e. squared Euclidean norm defined in (2)) of different systems on the training set, validation set, and test set are listed in Table 2. The distortions of *k*-means-VQ on all data sets are identical, but those of the DNN-based VQ systems are not. This is because the DNN-based system is generally more likely to suffer from the overfitting problem. We use the early-stopping strategy to combat this problem by monitoring the model's performance on the validation set. The results in Table 2 show that DNN-VQ significantly outperforms the baseline. In contrast, DNN-AN and DNN-FB underperform the traditional VQ system. This is because DNN-AN and DNN-FB are suitable for coding long frames (e.g., 9 frames in [7]), but unsuitable for short frames, which is the case in our experiment settings.

Then, we apply the de-normalization process to obtain the spectrogram and examine more details of the coding errors in terms of LSD across the frequency range. The average LSD results of each frequency bin on the test set are shown in Fig. 6. Clearly, DNN-VQ achieves the smallest LSD in most frequency bins. Especially, in the frequency bins 30, 60, and 90, the distortions of *k*-means-VQ increase dramatically, which is caused by the vector partition. The LSD results averaged over all the frequency bins are listed in the second column of Table 3. Compared with the baseline, DNN-VQ achieves a 4.6% lower LSD (from 5.44 dB to 5.19 dB).

Finally, we synthesize the time-domain speech signal and qualitatively examine the speech quality using SegSNR and PESQ. The results are shown in the third and fourth columns of Table 3. It can be clearly seen that, in comparison with the baseline, DNN-VQ achieves about 7.4% relative gain in terms of SegSNR (from 6.84 dB to 7.35 dB) and about 5.5% relative gain in terms of PESQ score (from 3.08 to 3.25).

5.2. Quantization of the acoustic context information using the DNN-VQ system

In this evaluation, we demonstrate the capability of the proposed method in quantizing acoustic context information. That is, more than one frame of speech signal is coded using the DNN-VQ system. The baseline of this evaluation is a 9-bit vector quantizer based on *k*-means. Since it is feasible to implement *k*-means with 2^9 clusters for 121-dimensional vector, the integral speech spectrum is used for clustering. With this setting, the distortion caused by the vector partition, which is illustrated in Fig. 6, can be avoided.

Table 4
Average LSD (dB), SegSNR (dB) and PESQ results using different number of frames with constant compression ratio.

VQ system	Vector dimension/ Coding bits	LSD	SegSNR	PESQ
DNN-VQ	121/9	6.88	4.18	2.33
	242/18	6.70	4.56	2.45
	484/36	**6.62**	**4.64**	2.48
	968/72	6.70	4.57	**2.50**
k-means-VQ (baseline)	121/9	6.89	4.18	2.33

Using DNN's layer size to denote each layer, the DAE architecture for this evaluation is $121 * N$-2048-2048-2048-9 $* N$-2048-2048-2048-$121 * N$, where $N = \{1, 2, 4, 8\}$ is the number of frames. That is, DAE's the input/output layer's size is $121 * N$. The hidden layers' size are 2048-2048-2048-9 $* N$-2048-2048-2048, in which the coding layer's size is $9*N$. For example, when N is 8, the input vector dimension is 968, and the coding layer's size is 72. This makes sure the compression ratio is constant. The input of the decoder network is obtained from the 9-bit vector quantizer frame by frame. The data sets of this evaluation are the same as Section 5.2, but it should be noticed that the number of samples is divided by N for the N frames acoustic context. For instance, there are more than 6 million training samples when N is 1, but there are only about 0.78 million training samples when N is 8.

The results of quantizing multiple frames with constant compression ratio on the test set are shown in Table 4. Compared with the baseline, DNN-VQ has distinctly better performance, e.g., a 3.92% lower LSD (from 6.89 dB to 6.62 dB), an 11% higher SegSNR (from 4.18 dB to 4.64 dB), and a 7.3% higher PESQ (from 2.33 to 2.50). The performance of DNN-VQ improves as the number of frame increases, except for the case of 8 frames. This is because there are more than 10 million parameters in the DNN model, but only 0.78 million training samples are available when the number of frames is 8. It can be expected that better performance can be attained when more training data are used.

6. Conclusion and future work

This paper proposed a novel DNN-based VQ method for coding high-dimensional vector with large codebook sizes. This method is derived via combining the traditional VQ technique and a DNN-based binary coding procedure. This combination ensures that the input of the decoder network is exactly binary and avoids the distortions induced by the binary coding layer in DAE. This method also has the capability to reduce the coding error of the traditional VQ system. Evaluation results on speech signal showed that, the proposed method can attain significantly better performance than the traditional k-means based method.

The proposed method would be preferred in many practical downstream applications, especially for ultra-low bit-rate speech coding [19], where coding high-dimensional vector with large bits is the exact requirement. We would like to integrate this technology into the mixed excitation linear prediction (MELP) coder [20] to improve the speech quality and obtain a further lower bit-rate coder.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grants 61401501, 61304225, and 61573242.

References

[1] MacQueen J. Some methods for classification and analysis of multivariate observations. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, Oakland, CA, USA, vol. 1. p. 281–97.
[2] Linde Y, Buzo A, Gray R. An algorithm for vector quantizer design. IEEE Trans Commun 1980;28(1):84–95.
[3] Gray R. Vector quantization. IEEE ASSP Magaz 1984;1(2):4–29.
[4] Gersho A, Gray R. Vector quantization and signal compression. Springer Science & Business Media; 1992.
[5] Hinton G, Salakhutdinov R. Reducing the dimensionality of data with neural networks. Science 2006;313(5786):504–7. http://dx.doi.org/10.1126/science.1127647.
[6] Hinton G, Osindero S, Teh Y. A fast learning algorithm for deep belief nets. Neural Comput 2006;18(7):1527–54. http://dx.doi.org/10.1162/neco.2006.18.7.1527.
[7] Deng L, Seltzer M, Yu D, Acero A, et al. Binary coding of speech spectrograms using a deep auto-encoder. In: Interspeech, Citeseer. p. 1692–5.
[8] Salakhutdinov R, Hinton G. Semantic hashing. Int J Approx Reason 2009;50(7):969–78. http://dx.doi.org/10.1016/j.ijar.2008.11.006.
[9] Krizhevsky A, Hinton G. Using very deep autoencoders for content-based image retrieval. In: ESANN, Citeseer.
[10] Huang Z, Weng C, Li K, et al. Deep learning vector quantization for acoustic information retrieval. In: 2014 IEEE international conference on acoustics, speech and signal processing (ICASSP). IEEE; 2014. p. 1350–4. http://dx.doi.org/10.1109/ICASSP.2014.6853817.
[11] Bottou L, Bengio Y. Convergence properties of the k-means algorithms. In: Advances in neural information processing systems 7, [NIPS Conference, Denver, Colorado, USA, 1994]. p. 585–92.
[12] Sculley D. Web-scale k-means clustering. In: Proceedings of the 19th international conference on World Wide Web. ACM; 2010. p. 1177–8. http://dx.doi.org/10.1145/1772690.1772862.
[13] Wu J, Hong B. An efficient k-means algorithm on CUDA. In: 2011 IEEE international symposium on parallel and distributed processing workshops and Phd forum (IPDPSW). IEEE; 2011. p. 1740–9. http://dx.doi.org/10.1109/IPDPS.2011.331.
[14] Arthur D, Vassilvitskii S. k-means++: the advantages of careful seeding. In: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. Society for Industrial and Applied Mathematics; 2007. p. 1027–35.
[15] Bengio Y. Learning deep architectures for ai. Found Trends® Mach Learn 2009;2(1):1–127. http://dx.doi.org/10.1561/2200000006.
[16] Bengio Y, Lamblin P, Popovici D, et al. Greedy layer-wise training of deep networks. Adv Neural Inform Process Syst 2007;19:153.
[17] Recommendation I. Perceptual evaluation of speech quality (PESQ), an objective method for end-to-end speech quality assessment of narrowband telephone networks and speech codecs. ITU-T Recommend 2001:862.
[18] Price P, Fisher W, Bernstein J, Pallett D. Resource management RM2 2.0 LDC93S3C. Philadelphia: Linguistic Data Consortium.
[19] Boucheron L, De Leon PL, Sandoval S. Low bit-rate speech coding through quantization of Mel-frequency cepstral coefficients. IEEE Trans Audio, Speech, Lang Process 2012;20(2):610–9. http://dx.doi.org/10.1109/TASL.2011.2162407.
[20] Guilmin G, Capman F, Ravera B, et al. New NATO STANAG narrow band voice coder at 600 bits/s. In: 2006 IEEE international conference on acoustics, speech and signal processing, 2006. ICASSP 2006 Proceedings. IEEE; 2006. http://dx.doi.org/10.1109/ICASSP.2006.1660114.