Interspeech 2018
2-6 September 2018, Hyderabad

Multi-frame Quantization of LSF Parameters using a Deep Autoencoder and Pyramid Vector Quantizer

Yaxing Li[1], *Eshete Derb Emiru*[1], *Shengwu Xiong*[1,*], *Anna Zhu*[1], *Pengfei Duan*[1], *Yichang Li*[2]

[1]School of Computer Science and Technology, Wuhan University of Technology, China
[2]School of Computer Science and Engineering, Nanjing University of Science & Technology, China

yaxing.li@whut.edu.cn, xiongsw@whut.edu.cn

Abstract

This paper presents a multi-frame quantization of line spectral frequency (LSF) parameters using a deep autoencoder (DAE) and pyramid vector quantizer (PVQ). The object is to provide sophisticated LSF quantization for the ultra-low bit rate speech coders with moderate delay. For the compression and de-correlation of multiple LSF frames, a DAE possessing linear coder-layer units with Gaussian noise is used. The DAE demonstrates a high degree of modelling flexibility for multiple LSF frames. To quantize the coder-layer vector effectively, a PVQ is considered. Comparing the discrete cosine model (DCM), the DAE-based compression shows better modelling accuracy of multi-frame LSF parameters and possesses an advantage in that the coder-layer dimensions could be any value. The compressed coder-layer dimensions of the DAE govern the trade-off between the modelling distortion and the coder-layer quantization distortion. The experimental results show that the proposed algorithm with determined optimal coder-layer dimension outperforms the DCM-based multi-frame LSF quantization approach in terms of spectral distortion (SD) performance and robustness across different speech segments.

Index Terms: Line spectral frequency (LSF), quantization, deep autoencoder (DAE), pyramid vector quantizer (PVQ)

1. Introduction

With regard to linear predictive coding (LPC) based speech coders, the quantization of line spectral frequency (LSF) parameters is a major issue [1]. LSF parameters are mathematically equivalent to LPC parameters, providing very good performance with regard to quantization and interpolation [1, 2]. LSF parameters are usually extracted, quantized, and transmitted on a frame-by-frame basis in speech coders. High correlation between consecutive LPC parameters has been evidenced and it can be exploited for LSF quantization [2]. In order to reduce the encoding rate, the multi-frame coding of LSF parameters has been extensively applied to ultra-low bit rate speech coders with moderate delay, such as enhanced mixed excitation linear prediction (MELPe) and its variations [3–6]. The block-constrained trellis coded vector quantization schemes were recently developed for quantization of multiple frames LSF parameters [7].

An efficient discrete cosine model (DCM) [2] was introduced to model the LSF time-trajectory, and consecutive LSF frames were jointly quantized to exploit their high inter-frame correlation. At the encoder, the DCM coefficients of the time trajectory of multiple LSF frames are calculated and switched to a reduced set of LSF vectors. Then, the reduced set of LSF vectors are quantized via a multi stage vector quantization (MSVQ). At the decoder, the corresponding quantized DCM coefficients are recalculated from the quantized version of the reduced LSF vectors. Finally, the quantized multiple LSF vectors are derived from the quantized DCM coefficients. As shown in [2], the DCM-based approach exhibits better performance than the frame-by-frame and other multi-frame quantization methods.

In the DCM-based approach, inter-frame correlation between consecutive LSF frames is exploited and the reduced LSF set still shows a correlation. In this paper, we propose the combination of a deep autoencoder (DAE) [8] and pyramid vector quantizer (PVQ) [9] for the quantization of multiple LSF frames. Inspired by the successful usage of DAE for dimensionality reductions [8], we train a DAE whose coder-layer possesses linear units with Gaussian noise to model multiple LSF frames. The DAE demonstrates a high degree of modelling flexibility for multiple LSF frames and it possesses better modelling accuracy than the DCM. In the encoder, the lower layers of the DAE are used to compress the multiple LSF frames to a coder-layer vector with reduced dimensions. Both inter-frame and intra-frame correlations of consecutive LSF frames have been exploited by the DAE since the compressed coder-layer units are found to be uncorrelated. The PVQ proposed by Fischer provides better performance than the optimal scalar quantizer for a memoryless Gaussian source [9]. The low-dimensional coder-layer vector is then quantized by the PVQ. In the decoder, the quantized coder-layer vector is applied to the upper layers of the DAE to reconstruct the quantized multiple LSF frames. The experimental results show that the proposed algorithm produces better spectral distortion (SD) performance than the combination of a DCM and MSVQ.

2. Multiple LSF Frames quantization using a deep autoencoder and PVQ

2.1. Long-term quantizer for multiple LSF frames

The proposed long-term quantization block diagram for multiple LSF frames is depicted in Figure 1. At the encoder, N consecutive LSF frames $\boldsymbol{f}_{1,N}$ are provided as the DAE input. Two lower layers of the DAE are used to compute the coder-layer vector $\boldsymbol{h}_2$. The coder-layer vector $\boldsymbol{h}_2$ has a Gaussian distribution and each unit of the vector is found to be uncorrelated. The PVQ is a type of lattice quantizer, with the codewords selected as the cubic lattice points which lie on the surface of a pyramid [9]. Thus, PVQ does not require memory for codebook storage and it could be used as an efficient

10.21437/Interspeech.2018-2577

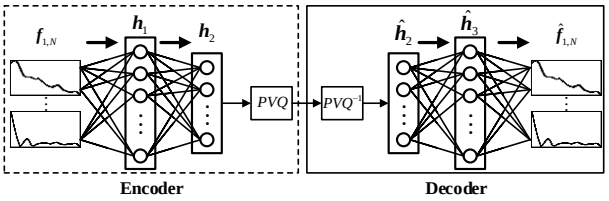


Figure 1: *Block diagram of the proposed long-term quantizer for multiple LSF frames*.

vector quantizer for $\boldsymbol{h}_2$. At the decoder, the quantized coder-layer vector $\hat{\boldsymbol{h}}_2$ is then used to reconstruct the N consecutive LSF frames $\hat{\boldsymbol{f}}_{1,N}$ using the two upper layers of the DAE.

2.2. Deep autoencoder for multiple LSF frames

It was introduced in [10] that restricted Boltzmann machines (RBMs) are good at modelling acoustic features with cross-dimensional correlations. The conventional RBMs, assuming that both the visible and hidden units are binary data, are referred to as Bernoulli-Bernoulli RBMs [11]. Hinton formulated an RBM while assuming the input to be a Gaussian distribution to apply real-valued data such as LSF, referred to as the Gaussian-Bernoulli RBM [8, 11]. It was shown in [12, 13] that LSF could be well modelled by deep neural network (DNN) with multiple RBMs. The DAE is trained to make the output layer vectors as similar possible to input vectors and it is considered as special type of DNN.

The unsupervised pre-training algorithm [14] is a useful technique to initialize the DNN, and its potential benefits have been previously discussed. This algorithm guides learning to reduce the generalization error; hence, pre-trained neural networks usually perform better than conventional neural networks in terms of training error. As presented in the upper part of Figure 2, a stack of multiple restricted Boltzmann machines (RBMs) [15] are pre-trained to learn a deep generative model of N consecutive LSF frames, $\boldsymbol{f}_{1,N}$. The popular greedy learning algorithm is used to train the deep generative model in a layer-by-layer fashion [15]. RBM1, which is needed to create a higher-level representation of the visible training data, is estimated first. RBM1 should have been the Gaussian-Bernoulli RBM with linear visible variables and binary hidden units, because the DAE is fed with the real-valued N consecutive LSF vectors. The first layer parameters are then frozen, and the reconstructed samples from RBM1 are

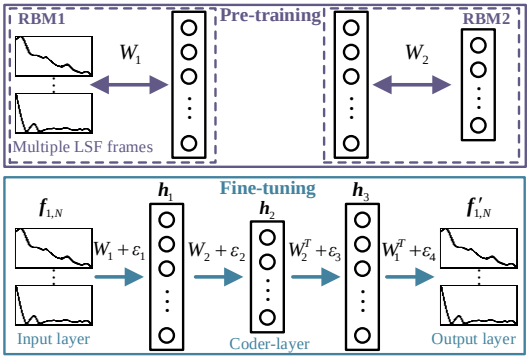


Figure 2: *Deep autoencoder architecture and training procedure.*

used to train RBM2. RBM2 is the Bernoulli-Gaussian RBM with binary visible and linear hidden units. During the layer-wise training, the parameters of each RBM are updated through efficient contrastive divergence learning [15]. After the layer-by-layer pre-training, the RBMs are unfolded by using its weight matrices to create a deep five-layer autoencoder network [8], as seen in the lower part of Figure 2. The lower layers of the DAE uses the matrices to encode the input N successive LSF vectors and the upper layers use the matrices in the reverse order to decode the input. The DAE is then fine-tuned to minimize the reconstruction error. The N consecutive LSF frames reconstruction process is defined as follows [16]:

$$\boldsymbol{h}_l = \mathrm{f}_l(\boldsymbol{W}_l \boldsymbol{h}_{l-1} + \boldsymbol{b}_l), \quad 1 \le l \le L+1, \tag{1}$$

where $\boldsymbol{h}_l$ represents the output of the l^{th} layer, $(\boldsymbol{W}_l,\boldsymbol{b}_l)$ denotes the weight and bias parameters of the l^{th} layer, respectively, L denotes the number of hidden layers, and $\mathrm{f}_l(\bullet)$ signifies the activation function. $\boldsymbol{h}_0$ represents the input layer features and is initialized as the N consecutive LSF vectors, $\boldsymbol{f}_{1,N}$. The estimated N consecutive LSF frames $\boldsymbol{f}'_{1,N}$ equals to the output layer vector $\boldsymbol{h}_{L+1}$. The activation function of the first and third hidden layers are sigmoidal, while the activation function for second hidden layer and output layer are linear. The fine-tuning procedures can be seen in the lower part of Figure 2. The mean square error (MSE) criterion between the original and estimated N consecutive LSF vectors is used to optimize the DAE [8, 15]. The MSE is minimized as follows:

$$\min \frac{1}{M} \sum_{m=1}^{M} \left\| \boldsymbol{f}_{1,N}(m) - \boldsymbol{f}'_{1,N}(m) \right\|_2^2, \tag{2}$$

where $\boldsymbol{f}_{1,N}(m)$ and $\boldsymbol{f}'_{1,N}(m)$ represent the original and reconstructed N consecutive LSF vectors for the sample index m, respectively, with M denoting the size of the mini-batch. The hidden units of RBM2 are referred to as the coder-layer of the DAE. Modelling error is defined as the difference between the reconstructed and original LSF vectors. We train the RBM2 linear hidden units with Gaussian noise because our study determines that a fine-tuned DAE with linear coder-layer units demonstrates better modelling accuracy than with binary units.

2.3. Quantization of the coder-layer units

The vector quantizer for $\boldsymbol{h}_2$ is designed on the pyramid given by [9]:

$$S(L,r) = \left\{ \boldsymbol{h}_2 : \sum_{i=1}^{L_p} |h_2(i)| = r \right\}, \tag{3}$$

where r is the radial parameter that indexes the pyramid $S(L_p,r)$ and L_p represents the dimension of the coder-layer vector $\boldsymbol{h}_2$. For moderate size dimensions, the relative variance of r maybe appreciable and a product code PVQ [9] can be used to reduce the quantization distortion significantly. The product code pyramid possesses identical relative orientations of the output vectors on each pyramid, and the pyramids are indexed by the quantized versions of r. The quantity and location of output vectors on each pyramid are selected to minimize the average distortion. In the product code PVQ, a single pyramid VQ with 2^{RpLp} output levels is designed for the output vectors, and a scalar quantizer (Max quantizer [17]) with 2^{Rr} outputs levels is designed for r. The average rate per dimension R_p and

the scalar quantizer R_r are constrained to satisfy the following equation:

$$R_p L_p + R_r = RL_p, \quad (4)$$

where R is the total average rate per dimension. Assuming K to be a positive integer, then the pyramid $S(L_p,K)$ possesses many cubic lattice points (corresponding to codewords) on its surface. If we denote $N(L_p,K)$ as the number of codewords on pyramid $S(L_p,K)$, then K must be chosen as the largest integer such that:

$$N(L_p,K) \le 2^{R_p L_p}. \quad (5)$$

For quantizing the coder-layer vector $\boldsymbol{h}_2$, we design the product PVQ, which possesses $R*L_p$ bits, and among them, R_r bits are used for the scalar quantizer of r. The values of R_p and integer K can be calculated from (4) and (5), respectively. Then, the PVQ encoding algorithm for the L_p dimensional coder-layer vector $\boldsymbol{h}_2$ can be designed with a set value of R_r and calculated values of R_p and K. With the specified values of R_r, R_p and K, the PVQ encoding algorithm for the coder-layer vector $\boldsymbol{h}_2$ is given as:

Step 1: Quantize r via a Max quantizer with 2^{Rr} outputs levels and get the quantizer output $\hat{r}$.

Step 2: For each coder-layer vector $\boldsymbol{h}_2$, choose the nearest $\tilde{\boldsymbol{h}}_2 \in S(L_p,\hat{r})$.

Step 3: Scale $\tilde{\boldsymbol{h}}_2$ by $K/\|\tilde{\boldsymbol{h}}_2\|_1$ such that the resulting vector $\hat{\boldsymbol{v}}$ is on $S(L_p,K)$.

Step 4: Find the integer point codeword $\hat{\boldsymbol{v}} \in S(L,K)$, nearest to $\hat{\boldsymbol{v}}$. This can be performed as follows:

(*a*) Round each element of $\hat{\boldsymbol{v}}$ to the nearest integer to obtain the resulting vector $\hat{\boldsymbol{v}}$.

(*b*) Calculate $\|\hat{\boldsymbol{v}}\|_1$. If $\|\hat{\boldsymbol{v}}\|_1 = K$, then go to step 5. If $\|\hat{\boldsymbol{v}}\|_1 > K$, then decrease by one in magnitude the $(\|\hat{\boldsymbol{v}}\|_1 - K)$ nonzero elements of $\hat{\boldsymbol{v}}$ that contribute the largest error and were previously rounded up. If $\|\hat{\boldsymbol{v}}\|_1 < K$, then increase by one in magnitude the $(K - \|\hat{\boldsymbol{v}}\|_1)$ elements of $\hat{\boldsymbol{v}}$ that contributes the largest error and were previously rounded down.

Step 5: Scale $\hat{\boldsymbol{v}}$ by $\|\tilde{\boldsymbol{h}}_2\|_1 / K$ to produce the PVQ output vector $\hat{\boldsymbol{h}}_2'$. The product PVQ output is $\hat{\boldsymbol{h}}_2 = \hat{r}\hat{\boldsymbol{h}}_2'$.

Different R_r values are applied for experiment and the optimal R_r which produces the best quantization performance is used in the final PVQ.

3. Experiments and results

3.1. Experiment setup

Experiments are conducted using the TIMIT database [18] and all of the 4200 speech utterances are down sampled to 8 kHz. A total of 3800 sentences are selected for training and the remaining 400 sentences are used for testing. This results in a connection of 196.29 minutes training speech and 20.41 minutes testing speech. The 10th order LSF vectors are calculated for every 10 ms frame with a 30 ms Hamming window. For pre-training each RBM, all training vectors are subdivided to mini-batches [8, 15], with each containing 100 training vectors; the weights are updated after each mini-batch. The RBMs are pre-trained with 50 epochs and the learning rate is 0.001. A momentum of 0.9 and a small weight-cost of 0.0002 are utilized during the pre-training. For fine-tuning, a conjugate gradient method with 200 epochs is employed [8]. The input and output features are normalized to zero mean and unit variance, and a reverse step is processed on the output.

3.2. Results and discussions

3.2.1. Modelling accuracy comparison

Performance comparisons are first made between the modelling error of the DCM and DAE. In the DCM, N consecutive LSF vectors are compressed into a reduced LSF set with P+1 vectors, where P is a positive integer defining the order of the model [2]. We train the DAE networks with the coder-layer vector dimension, identical to the DCM compressed vector dimension, that is, $L_p=(P+1)\times 10$. The quantity of the first and third hidden nodes is 100. To measure the performance of the reconstructed LSF parameters, we employ the spectral distance (SD) [1], which is defined as follows:

$$SD = \sqrt{\frac{1}{2\pi}\int_{-\pi}^{\pi} 10\log_{10} P(\omega) - 10\log_{10} \hat{P}(\omega)^{\,2}\, d\omega}, \quad dB, \quad (6)$$

where $P(\omega)$ and $\hat{P}(\omega)$ are the LPC power spectra of the original and quantized LSF vectors, respectively.

Table 1 presents the modelling error performance of the DCM and DAE for 5 consecutive LSF vectors, as a function of the compressed vector dimension. The results show that the DAE possesses better modelling accuracy than the DCM when their compressed vectors has the same dimensions.

Table 1: *Modelling error comparison of DCM and DAE for five consecutive LSF vectors*.

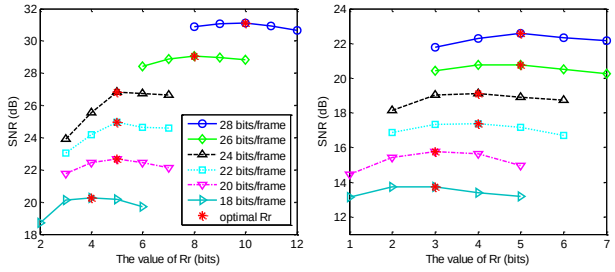
Compressed vector dimension	Avg. SD (dB)		2-4 dB outliers (%)	
	DCM	DAE	DCM	DAE
20	0.90	0.79	7.81	1.59
30	0.67	0.53	3.47	0.13
40	0.63	0.48	2.5	0.06

The proposed long-term quantizer, which combines the DAE and PVQ, is referred to as the DAE-PVQ, and the combination of the DCM and MSVQ in [2] is denoted as the DCM-MSVQ. The overall quantization performance of the DCM-MSVQ and DAE-PVQ is evaluated. Considering a constrained coding delay in telephony applications, we concatenate 3 consecutive frames for the experiment, i.e., N is set to 3. For the DCM, the only choice for P is 1 for 3 LSF frames, meaning that the dimension of the reduced LSF set is 20 (= $(P+1)\times 10$). The DCM-based compression possesses the constraint that the dimensions of the reduced set needs to be a multiple of the LSF vector, but the DAE-based compression possesses an advantage in that the coder-layer dimensions could be any value.

3.2.2. Rate-distortion performance comparison

We train several DAE networks with the coder-layer vector dimension L_p varying from 14 to 20 in increments of 2. The PVQ totally uses $R*L_p$ bits to quantize the coder-layer vector,

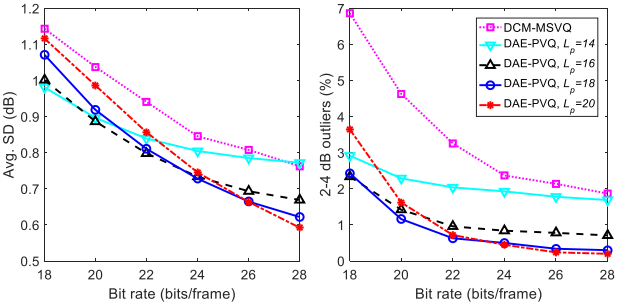
and the corresponding bit rate for LSF quantization is RL_p/N bits per frame. For a specific bit rate and coder-layer dimension, we need to determine the optimal R_r, producing the best quantization performance for the PVQ. Figure 3 presents the SNR values of PVQ quantized coder-layer vectors as a function of R_r. The selected optimal R_r values are used in the final PVQ to evaluate the LSF quantization performance.



(*a*) $L_p = 14$ (*b*) $L_p = 20$

Figure 3: *SNR values of the L_p-dimensional PVQ for different bit rates of R_r.*

Figure 4 presents a rate-distortion performance comparison between the DCM-MSVQ and DAE-PVQ. The average SD curves of DAE-PVQ are situated on the left of the DCM-MSVQ. Thus, DAE-PVQ yields a performance gain with regard to both the average SD and 2-4 dB outliers, compared to DCM-MSVQ at the same bit rate. For small values of L_p, large performance gains at low bit rates benefit from the coder-layer dimension of DAE being allowed to take any value. For increased values of L_p, large performance gains are achieved at high bit rates due to the improved modelling accuracy of DAE. Figure 4 also illustrates the trade-off between modelling accuracy of the DAE and the quantization accuracy of PVQ. This trade-off is influenced by the coder-layer dimension; a proper coder-layer dimension producing the best performance can be selected depending on the actual bit rate.



(*a*) Average SD (*b*) 2-4 dB outliers

Figure 4: *Rate-distortion performance comparison between DCM-MSVQ and DAE-PVQ.*

3.2.3. Implement into ultra-low bit rate speech coders

For the MELPe coder at 1200 bps mode, the three consecutive LSF frames are quantized using MSVQ. The DCM-MSVQ and proposed DAE-PVQ algorithm are implemented to the MELPe standard codec at 1200 bps mode, in which six unvoiced (U) or voiced (V) frame combinations or voicing classes are considered. For each voicing class, the DCM-MSVQ and the DAE-PVQ algorithms are designed at the same LSF parameters encoding rate as the standard codec.

Table 2 shows the average SD performance of MSVQ, DCM-MSVQ and DAE-PVQ with fixed coder-layer dimensions (L_p=20) and evaluated optimal coder-layer dimensions. As shown in Table 2, DAE-PVQ with the optimal coder-layer dimensions produces better SD performance than that of the DCM-MSVQ, since the DAE-PVQ method can determine the optimal coder-layer dimension based on the actual encoding rate. Comparing the MSVQ used in MELPe, the DAE-PVQ produces lower SD values in each voicing class, but the DCM-MSVQ demonstrates worse performance for the VVU voicing class. Thus, the DAE-PVQ shows better robustness than the DCM-MSVQ across different speech segments or different voicing classes.

Table 2: *Average SD performance of DCM-MSVQ and DAE-PVQ for each voicing class in MELPe coder at 1200 bps mode*.

Methods	Voicing classes		
	UUU (27bits)	VUU (42bits)	UVU (42bits)
MELPe (MSVQ)	3.15	2.64	2.51
DCM-MSVQ (20 dimensions)	2.31	2.33	2.30
DAE-PVQ (L_p =20)	2.78	2.07	2.08
DAE-PVQ (optimal L_p)	2.14 (L_p =12)	1.99 (L_p =16)	1.96 (L_p =16)
Methods	Voicing classes		
	UUV (42bits)	VVU (39bits)	Others (42bits)
MELPe (MSVQ)	2.65	2.74	2.65
DCM-MSVQ (20 dimensions)	2.41	2.99	2.51
DAE-PVQ (L_p =20)	2.10	2.50	2.52
DAE-PVQ (optimal L_p)	2.01 (L_p =15)	2.30 (L_p =15)	2.24 (L_p =14)

4. Conclusions

The consecutive line spectral frequency (LSF) parameters, showing a high inter-frame and intra-frame correlation, are usually jointly quantized to reduce the encoding rate of the ultra-low bit rate speech coders. This paper presents a multi-frame quantization of LSF parameters using a deep autoencoder (DAE) and pyramid vector quantizer (PVQ). To compress and de-correlate multiple LSF frames, we apply a deep autoencoder whose coder-layer units are linear with Gaussian noise. To quantize the coder-layer vector, we apply PVQ. The experiment results show that the proposed DAE-PVQ algorithm with determined optimal coder-layer dimension outperforms the DCM-MSVQ method in terms of spectral distortion (SD) performance and robustness across different speech segments.

5. Acknowledgements

The authors acknowledge the contribution of Prof. Kang Sangwon for this work and it is with sadness to note that Prof. Kang Sangwon passed away on May 4, 2018.

This work was partially supported by National Key R&D Program of China (No. 2017YFB1402203, No. 2016YFD0101903), and funded by the Fundamental Research Funds for the Central Universities (WUT: 2018IVA028).

6. References

[1] A. M. Kondoz, *Digital Speech: Coding for low bit rate communication systems, 2nd Edition*, Chichester, West Sussex, England: John Wiley & Sons Ltd, 2004.

[2] L. Girin, "Long-term quantization of speech LSF parameters," in *Proc. Int. Conf. on Acoustics, Speech, and Signal Processing*, Apr. 2007, pp. IV-845–IV-848.

[3] The 600 bit/s, 1200 bit/s and 2400 bit/s NATO interoperable narrow band voice coder, STANAG 4591, 2008.

[4] L. Zhu and Q. Li, "A 600bps Vocoder Algorithm Based on MELP," *in 2017 2nd International Conference on Electrical and Electronics*: *Techniques and Applications*, 2017, pp. 306–311.

[5] X. Ma, Y. Li, J. Jiang, et al, "400bps High-Quality Speech Coding Algorithm," *in 2016 International Symposium on Computer, Consumer and Control*, 2016, pp. 256–259.

[6] Y. Li, Q. Hao, P. Zhang, et al, "A variable-bit-rate speech coding algorithm based on enhanced mixed excitation linear prediction," *in 2016 9th International Congress on Image and Signal Processing, Biomedical Engineering and Informatics*, 2016, pp. 915-919.

[7] Y. Li, S. Xu, S. Xiong, A. Zhu, P. Duan and Y. Ding, "Multi-frame coding of LSF parameters using block-constrained trellis coded vector quantization," *in Proc. INTERSPECCH* (to appear), 2018.

[8] G. E. Hinton, R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[9] T. R. Fisher, "A pyramid vector quantizer," *IEEE Trans. Inform. Theory*, vol. IT-32, no. 4, pp. 568–583, Jul. 1986.

[10] Z. Ling, S. Kang, H. Zen, A. Senior, M. Schuster, X. Qian, H. Meng and L. Deng, "Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends," *IEEE Signal Process. Mag.*, vol. 32, no. 3, pp. 35–52, Apr. 2015.

[11] T. Yamashita, M. Tanaka, E. Yoshida, Y. Yamauchi and H. Fujiyoshii, "To be Bernoulli or to be Gaussian, for a restricted Boltzmann machine," in *Proc. 22nd Int. Conf. on Pattern Recognition*, Aug. 2014, pp. 1520–1525.

[12] Y. Li and S. Kang, "Artificial bandwidth extension using deep neural network-based spectral envelope estimation and enhanced excitation estimation," *IET Signal Processing*, vol. 10, no. 4, pp. 422–427, Jun. 2016.

[13] Y. Li and S. Kang, "Deep neural network-based linear predictive parameter estimations for speech enhancement," *IET Signal Processing*, vol. 11, no. 4, pp. 469–476, Jun. 2017.

[14] D. Erhan, Y. Bengio, A. Courville and P. A. Vincent, "Why does unsupervised pre-training help deep learning?" *Journal of Machine Learning Research*, vol. 11, pp. 625–660, Feb., 2010.

[15] G. E. Hinton, S. Osindero, and Y. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput*., vol. 18, no. 7, pp. 1527–1554, 2006.

[16] D. Liu, P. Smaragdis, M. Kim, "Experiments on deep learning for speech denoising," in *Proc. INTERSPEECH*, Sep. 2014.

[17] J. Max, "Quantizing for minimum distortion," *IRE Trans. Inf. Theory*, vol. IT-6, no. 1, pp. 7–12, 1960.

[18] J. S. Garofolo, L. Lamel, W. M. Fisher and D. S. Pallett, "Getting started with the DARPA TIMIT CD-ROM: An acoustic phonetic continuous speech database," National Institute of Standards and Technology (NIST), Gaithersburgh, MD, 1988.