INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

End-to-end Optimized Multi-stage Vector Quantization of Spectral Envelopes for Speech and Audio Coding

Mohammad Hassan Vali, Tom Bäckström

Department of Signal Processing and Acoustics, Aalto University, Finland
mohammad.vali@aalto.fi, tom.backstrom@aalto.fi

Abstract

Spectral envelope modeling is an instrumental part of speech and audio codecs, which can be used to enable efficient entropy coding of spectral components. Overall optimization of codecs, including envelope models, has however been difficult due to the complicated interactions between different modules of the codec. In this paper, we study an end-to-end optimization methodology to optimize all modules in a codec integrally with respect to each other while capturing all these complex interactions with a global loss function. For the quantization of the spectral envelope parameters with a fixed bitrate, we use multi-stage vector quantization which gives high quality, but yet has a computational complexity which can be realistically applied in embedded devices. The obtained results demonstrate benefits in terms of PESQ and PSNR in comparison to the 3GPP EVS, as well as our recently proposed PyAWNeS codecs.

Index Terms: speech and audio coding, spectral envelope modeling, multi-stage vector quantization, end-to-end optimization

1. Introduction

Speech and audio coding is the most widely used application of speech processing with more than 8 billion mobile phones in use [1]. The main objective of such lossy coding is to compress a signal into a given bitrate with the best possible quality. We furthermore expect that coding will, in the near future, also be used in Internet-of-things (IoT) devices, such that the number of devices supporting speech and audio coding will increase greatly. It is therefore important to develop methods to enhance efficiency and quality, since any improvements would lead to better performance for a large number of devices. Concurrently, the tremendous success of machine learning methods in other areas of speech processing, such as speech recognition, speech enhancement and classification applications [2,3], raises hopes that there is a lot to be gained also in speech and audio codecs in terms of quality, bitrate and computational efficiency.

Spectral envelopes model the smoothed shape of the magnitude spectrum. Linear predictive coding (LPC) is one of the commonly used methods to model the spectral envelope in an autoregressive way, which is embodied in the code-excited linear prediction (CELP) speech coding algorithm [4]. LPC parameters are usually represented by line spectral frequencies (LSF) which is more effective and robust representation compared to other domains [5–7]. LSFs are usually quantized with vector quantization, since experiments have shown that it gives the best coding efficiency [8–10]. LPC models are not, however, a good fit for frequency domain codecs, since the LPC model is applied in the time-domain and switching between the time- and frequency-domains leads to added computational costs. Frequency-domain codecs such as MPEG USAC, on the other hand, use piece-wise constant envelope models known as scale factor bands [11]. This representation is however known to be less efficient than modeling with an LPC. In our earlier work, we have tried to remedy the difference with distribution quantization (DQ), which estimates the energy proportion among different sections of the spectrum [12, 13]. The DQ parameters are approximately uncorrelated, such that they can be quantized with scalar quantization. Another approach is to improve the entropy coding of scale factor bands with Gaussian mixture modeling [14].

A completely different approach for improving speech coding is to take advantage of machine learning models. In particular, generative models can be used to create speech signal samples using conditional probability distribution which is conditioned on previous signal samples. For example WaveNet is a generative deep neural network, which has been shown to be highly efficient [15]. Though this method provides high efficiency compared to traditional coding methods, it comes at the cost of a very high computational complexity that makes it infeasible to be applied for typical devices. To reduce complexity, WaveNet has been later improved by for example an autoregressive generative model which is used to predict the distribution of speech samples using WaveGRU [16, 17].

In this paper, we explore improvements to the recently presented Python acoustic wireless network of sensors (PyAWNeS) -codec [18]. This codec is a frequency-domain codec related to the TCX-mode of 3GPP EVS which was designed for ad-hoc wireless sensor networks. The original version uses a scale-factor-band type envelope model encoded with a trivial entropy coder. In this work, we improve the efficiency of the entropy coder by applying a multi-stage vector quantization (MSVQ) [19]. A key novelty of the proposed method is that we optimize the MSVQ codebooks together with all other modules in the codec with a global loss function. In other words, we propose an end-to-end optimized coding method using machine learning optimization algorithms in order to achieve the maximum potential coding efficiency. With end-to-end optimization we avoid potentially detrimental effects of complicated interactions between different modules of the codec [20]. In addition, we present a method for automatic evaluation of a vector quantization approach to determine whether its bitrate is optimal. Our coding method also has low computational cost in a way that it is applicable to low-resource devices. Experimental results demonstrate that the proposed method can be used to improve the quality of the PyAWNeS and 3GPP EVS codecs.

2. Multi-Stage Vector Quantization (MSVQ)

Vector quantization (VQ) is a method for compressing the distribution of a signal with a fixed number of bits. VQ is especially effective in cases where the data has complicated structures which are not known in advance [4]. Specifically, suppose $x \in \mathbb{R}^{M\times 1}$ is a vector of M envelope parameters to be

http://dx.doi.org/10.21437/Interspeech.2021-867

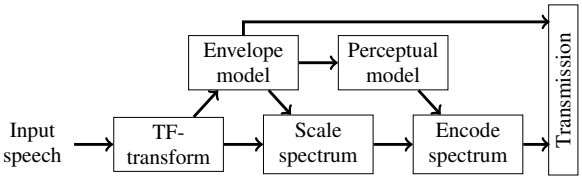

Figure 1: *Overall structure of encoder (identical to [18]).*

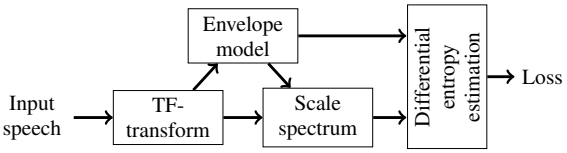

Figure 2: *End to end training of source model, optimizing all parameters of the envelope and entropy models.*

quantized and vector $c_k \in \mathbb{R}^{M\times 1}$ is the kth codebook vector, $0 \leq k < N$. The index of the best-matching codebook vector is defined as

$$k_{min} = \arg\min_k \ d(x, c_k), \tag{1}$$

where $d(.,.)$ refers to a distance measure, like Euclidean distance and we define the quantized vector as $\hat{x} = c_{k_{min}}$. The main objective in design of a vector quantizer is then to find the codebook vectors to minimize the expected distance

$$\mathbf{E}[\arg\min_k \ d(x_i, c_k)] \approx \frac{1}{P}\sum_{i=0}^{P-1} \arg\min_k \ d(x_i, c_k), \tag{2}$$

where $\mathbf{E}[.]$ refers to the expectation operand which is calculated over all P available x_i vectors. A frequently used algorithm for determining the codebook vectors is the expectation maximization (EM) algorithm, which is an iterative approach that is guaranteed to improve the codebook vectors at each iteration.

The computational complexity of vector quantization increases exponentially with the bitrate. Specifically, in Eq. 1, we need to evaluate the distance between the target vector x and all N codebook vectors. The codebook size in turn is related to the bitrate B as $N = 2^B$ and the number of envelope parameters M. The algorithmic complexity is then $\mathcal{O}(M2^B)$. Clearly complexity thus becomes astronomically high at high bitrates B. One approach for reducing complexity is to use multi-stage vector quantization (MSVQ), where each stage quantizes the residual of the previous stage with small number of bits [21,22]. The input to the first stage of MSVQ is the original signal $y_0 := x$, which will be quantized to $\hat{y}_0$ with the assigned number of bits for this stage. The following stages, use the residual $y_{k+1} := y_k - \hat{y}_k$ as input and this process will be continued until the last stage of MSVQ. The quantization of the original vector x is then the sum of all intermediate quantizations, $\hat{x} := \sum_k \hat{y}_k$. The computational complexity of MSVQ is the sum of its parts. In particular, for example, if we have L layers of equal size, then the complexity is $\mathcal{O}(ML2^{B/L})$ which is considerably smaller than that of conventional vector quantization.

3. Codec Structure

The proposed codec is built on top of the recently proposed PyAWNeS codec [18], which is a frequency-domain codec similar to the TCX-mode of the 3GPP EVS standard [23], but designed for ad-hoc sensor networks and implemented using Python and PyTorch. In this work we use only the single-channel mode of PyAWNeS at a sampling rate of 16 kHz, though the proposed methods are also directly applicable to multi-channel and -device scenarios. The overall structure of the encoder is illustrated in Fig. 1.

As a time-frequency transform, the codec uses the MDCT to obtain a perfectly reconstructing and critically sampled representation of the signal [4, 24]. The window is half-sine but with a flat-top, of length 30 ms and with a 20 ms step between windows. The spectral envelope is modelled by $M = 16$ equal-width spectral bands, where the spectral energy ν_k of the kth band is estimated from the energy-vector x_k with a vector a_k as $\nu_k = a_k^T x_k$. The spectral power envelope describes the variance of the signal and the variance vector for band k is defined with a reconstruction vector b_k as $s_k = \nu_k b_k$. Equivalently, in matrix-form, we write $y = Ax$ and $s = By$, where $x \in \mathbb{R}^{N\times 1}$, $y \in \mathbb{R}^{M\times 1}$ and $s \in \mathbb{R}^{N\times 1}$ are respectively, the input spectral energy, envelope parameters and spectral power envelope vectors. Scalars N and M correspond to the number of parameters in the spectrum and envelope and the transform matrices are $A \in \mathbb{R}^{M\times N}$ and $B \in \mathbb{R}^{N\times M}$. Observe that with this construction, in difference to EVS, the envelope model also encodes the signal energy. In the PyAWNeS codec, the log-values of the envelope parameters y are quantized with a uniform scalar quantizer and coded with a scalar arithmetic coder based on univariate normal distributions [4, 18, 25].

The square root of the spectral power envelope is used to scale the spectrum such that it becomes approximately unit-variance. The distribution of the scaled spectrum is modelled with a logistic mixture model, such that it can be efficiently coded with arithmetic coding. The quantization accuracy is chosen based on a perceptual model similar, which is scaled to achieve the desired overall bitrate [26]. For quantization, we use dithered quantization to avoid vanishing components at low bitrates [27]. The perceptual model is adopted from 3GPP EVS.

A central idea of the PyAWNeS codec is that all parameters of the source model can be optimized end-to-end, including the transform matrices A, B, quantization accuracy of the envelope parameters, as well as the probability distributions of the spectral components [18, 20]. The end-to-end training is illustrated in Fig. 2. In this work, our objective is to improve envelope coding by replacing the uniform quantization and simple entropy coding with a multi-stage vector quantizer. We use the vector quantizer to encode the information of logarithmic envelope parameters and optimize all source model parameters end-to-end, including the codebooks of vector quantization.

4. Implementation

We have implemented the codec and optimize all modules end-to-end with the PyTorch machine learning library. The transform matrices A, B, codebook vectors for MSVQ and all logistic mixture distributions for spectral elements modeling are optimized with the Adam optimizer with learning rate of 10^{-2}. As our training dataset, we chose the train-clean-100 set from LibriSpeech corpus [28], which contains 100 hours clean English speech. We train the codec over 5 epochs, such that each file of the dataset represents one batch.

With vector quantization, the bitrate for spectral envelope modeling is fixed, such that the loss function for training only includes negative log likelihood of the joint distribution for logistic mixture elements, which is relative to the bitrate of the

spectrum

$$Loss\ Function = -\sum_k \log_2(f_k(y_k)), \tag{3}$$

where y_k is the kth spectral element and $f_k()$ refers to its probability distribution.

Another important point about the training process is related to the optimization of codebook vectors. Finding the index of the nearest codebook vector and assigning that vector to the corresponding envelope parameters is not differentiable. Therefore it is not possible to apply traditional hard version of vector quantization (VQ) in our implementation. In order to make the gradients go through our computation graphs in PyTorch while using VQ method, we use the built-in softmin function which gives a probability for each codebook vector with regard to its closeness to each frame of envelope parameters. To allow adjusting the "hardness" of the operation, we further multiplied the exponent part of the softmin with a scalar $\beta > 0$ as

$$Softmin(x_i) = \frac{e^{-\beta x_i}}{\sum_{j=0}^{N-1} e^{-\beta x_j}}, \tag{4}$$

where x_i is the distance of the desired frame of envelope parameters to the ith codebook vector and N is the total number of codebook vectors. In other words, a larger value for β leads to a more concentrated probability around the nearest codebook vector index. Based on informal experiments, we chose $\beta = 0.1$, because this is the largest value with which the transform matrices A, B converge to the same shape as with the hard minimum.

5. Bitrate Adaptation

A central issue in design of vector quantization is the choice of bitrate. An often-used approach is to experiment with many different configurations to find the best system structure. It is a time-consuming and heuristic approach, where it cannot be guaranteed that the best configuration is found. In contrast, the PyAWNeS codec is especially designed to allow joint optimization of all parameters, including bit allocation. Our objective is thus to develop an objective method for choosing the bitrate of vector quantization.

Changing dynamically the size of the codebook during training is impractical since the number of codebook vectors is constrained to powers of two and because changing the number of vectors would necessarily require the re-training of all vectors. Instead, we therefore simulate a change in the codebook size by dynamically adjusting the accuracy of the codebook. Specifically, suppose that we quantize x to c_k using Eq. 1, such that the quantization error is $e = x - c_k$. We can then scale the error by a scalar $\gamma > 0$ as $e_\gamma = \gamma e$ to obtain a new quantization

$$\hat{x}_\gamma := x - e_\gamma = x - \gamma e = (1-\gamma)x + \gamma c_k. \tag{5}$$

This scaled-error quantizer is thus an interpolation between the original vector x and its codebook vector for $0 \leq \gamma \leq 1$ and extrapolation for $\gamma > 1$.

When quantizing a scalar at high rates, reducing the quantization error in half requires approximately one bit, such that the error ϵ is related to the change in bitrate ΔB as $\epsilon = \epsilon_0 2^{-\Delta B}$, where ϵ_0 is the error when $\Delta B = 0$. For an M-dimensional vector $e \in \mathbb{R}^{M\times 1}$ of uncorrelated components with equal variance, correspondingly, we need M bits to reduce the error in half. The penalty in bitrate ΔB, corresponding to the scaled-error quantizer above, is thus $\Delta B = -M \log_2 \gamma$. To optimize the balance in bitrate between different components of the codec, we should thus add this bitrate-penalty term ΔB to the loss function during training.

It is important to note that the scaled-error quantizer is not a realizable quantizer in the sense that the decoder does not have access to the original x and thus cannot reconstruct the quantized signal according to Eq. 5. However, this quantizer can be used during training to determine whether the bitrate of the vector quantizer is appropriate. That is, if after training we have $\gamma \approx 1$ such that $\Delta B \approx 0$, then the bitrate is optimal and otherwise, ΔB will indicate approximately how much more or less bits should be assigned to the vector quantizer.

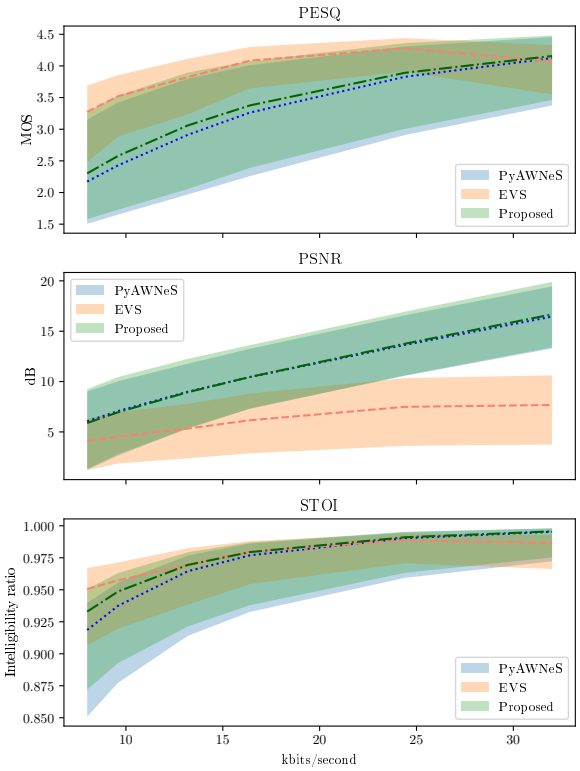

Figure 3: *Performance of the codecs at bitrates 8, 9.6, 13.2, 16.4, 24.4 and 32 kbit/s; dotted, dashed and dash-dotted lines refer to the median values of, respectively, the PyAWNeS, 3GPP EVS and proposed codecs, and the corresponding filled areas refer to their 95 % quantiles.*

6. Results

To analyze the performance of our proposed method, we use the PyAWNeS and 3GPP EVS codecs as references. We evaluate objective quality using the perceptual evaluation of speech quality (PESQ) [29], perceptually weighted signal to noise ratio (PSNR) and short-time objective intelligibility (STOI) [30]. Observe that codecs internally optimize PSNR as a measure of quality [4]. Quality is evaluated at the bitrates 8, 9.6, 13.2, 16.4, 24.4, 32 kbit/s, corresponding to operating modes of EVS. The experiments are performed over the test set of LibriSpeech corpus [28], and we chose linear predictive coding as the perceptual model for calculating PSNR metric for both proposed and

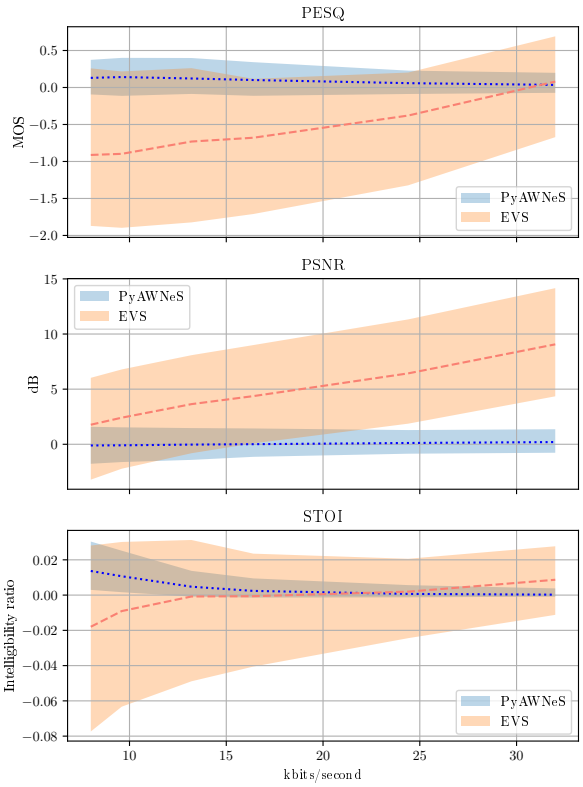

Figure 4: *The difference in objective scores between the proposed codec and PyAWNeS, and between the proposed codec and 3GPP EVS.*

PyAWNeS codecs.

The objective results are illustrated in Fig. 3. We observe that the PESQ median values for the proposed method grow as a function of the bitrate. Importantly, the proposed codec performs better than 3GPP EVS at 32 kbit/s and the trend suggests that it would be even better for higher bitrates as well. The proposed method also improves PESQ scores in comparison to PyAWNeS in 86 % of cases.

The PSNR median values of the proposed codec are clearly higher than the 3GPP EVS codec for the whole range of bitrates. For both the proposed method and the PyAWNeS codec, PSNR results increase linearly with the increase in bitrate, which confirms that the codec behaves consistently with respect to changes in bitrate. The proposed codec is better in terms of PSNR but not consistently better for PESQ, indicating that the perceptual model in the proposed method is the reason for subpar PESQ results.

In terms of STOI, our proposed method has higher scores than 3GPP EVS above 20 kbit/s bitrate, though all methods are close to saturation at higher bitrates. In addition, the proposed method has higher STOI median values than PyAWNeS codec for low bitrates, while they perform almost the same for bitrates above 24 kbit/s. All in all, the large overlap in 95 % quantiles for all three codecs under PESQ and STOI metrics shows that all coding methods have small difference in quality.

To better differentiate between the original PyAWNeS and the proposed method, Fig. 4 illustrates the differential scores, where we use the proposed method as reference and calculate the difference in performance to PyAWNeS and EVS. We can

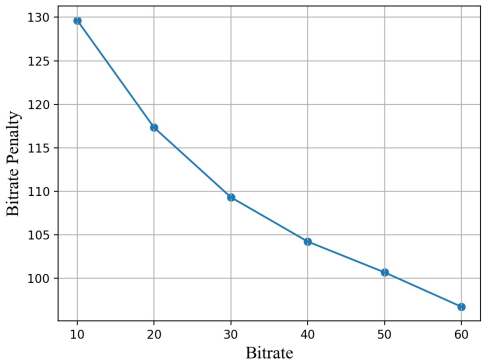

Figure 5: *Bitrate penalty ΔB of the proposed coding method for different bitrates.*

see that at 8 kbit/s, the PESQ score of the proposed method is approximately 0.15 MOS better than PyAWNeS, but −0.9 MOS worse than EVS. At 32 kbit/s, all three are approximately even, though the variance of the difference is much larger with EVS. With PSNR, there is hardly any difference between the proposed method and PyAWNeS, while the improvement over EVS is approximately 9 dB. With STOI, the results are inconclusive.

Finally, Fig. 5 illustrates the bitrate penalty ΔB at different sizes of codebook for the MSVQ. The MSVQs for the test were implemented at 10 bit steps such that each level in the multi-stage architecture had 10 bits. From 10 bit to 30 bit we see that the bitrate penalty ΔB decreases with steps of approximately 10 bit, confirming that our model captures the desired behaviour. In other words, since the penalty term is approximately 130 bit when the bitrate of the MSVQ is 10 bit, we conclude that the theoretically optimal bitrate for the MSVQ would be $130 + 10 = 140$ bit. However, at higher bitrates, the penalty term does not quite reach 10 bit steps, suggesting that the MSVQ is not theoretically optimal. This is expected, since MSVQ is a low-complexity approximation of a full vector quantizer at the same bitrate.

7. Conclusions

Modeling speech envelopes are central components of speech and audio codecs since they enable high coding efficiency. In this paper we study quantization of the envelope parameters using multi-stage vector quantization (MSVQ). We used end-to-end optimization to train all modules of the codec including the MSVQ codebook vectors for best coding efficiency. We defined a global loss function which was optimized using the PyTorch machine learning library. Our informal estimation is that the computational complexity of the proposed method, in use, is similar or lower than the EVS codec but higher than PyAWNeS, making it feasible to use in low-resource devices. The experimental results indicate that our proposed method performs better than PyAWNeS [18] codec in terms of PESQ and better than the 3GPP EVS [23] codec in terms of PSNR. We further presented a novel method to estimate the best bitrate for envelope modeling, which turns out to be approximately 140 bit/frame in our codec, though that would not be realizable due to the high computational complexity. These results can be used to improve the design process of future codecs, to determine the optimal configuration such that quality and efficiency maximized.

8. References

[1] World Bank, "Mobile cellular subscriptions," 2021. [Online]. Available: data.worldbank.org/indicator/IT.CEL.SETS

[2] T. N. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *Proc. ICASSP*, 2013, pp. 8614–8618.

[3] A. B. Nassif, I. Shahin, I. Attili, M. Azzeh, and K. Shaalan, "Speech recognition using deep neural networks: A systematic review," *IEEE Access*, vol. 7, pp. 19 143–19 165, 2019.

[4] T. Bäckström, *Speech Coding with Code-Excited Linear Prediction*. Springer, 2017.

[5] T. Bäckström and C. Magi, "Properties of line spectrum pair polynomials – a review," *Signal Processing*, vol. 86, no. 11, pp. 3286–3298, Nov. 2006.

[6] F. Itakura, "Line spectrum representation of linear predictor coefficients of speech signals," *The Journal of the Acoustical Society of America*, vol. 57, p. S35, 1975.

[7] F. Soong and B. Juang, "Line spectrum pair (LSP) and speech data compression," in *Proc. ICASSP*, vol. 9, 1984, pp. 37–40.

[8] K. K. Paliwal and B. S. Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame," *IEEE Trans. Speech Audio Process.*, vol. 1, no. 1, pp. 3–14, 1993.

[9] A. Gersho and R. M. Gray, *Vector quantization and signal compression*. Springer, 1992.

[10] R. Gray, "Vector quantization," *IEEE ASSP Magazine*, vol. 1, no. 2, pp. 4–29, 1984.

[11] ISO/IEC 23003–3:2012, "MPEG-D (MPEG audio technologies), Part 3: Unified speech and audio coding," 2012.

[12] T. Jähnel, T. Bäckström, and B. Schubert, "Envelope modeling for speech and audio processing using distribution quantization," in *Proc. EUSIPCO*, 2015.

[13] S. Korse, T. Jähnel, and T. Bäckström, "Entropy coding of spectral envelopes for speech and audio coding using distribution quantization," in *Proc. Interspeech*, 2016.

[14] S. Korse, G. Fuchs, and T. Bäckström, "GMM-based iterative entropy coding for spectral envelopes of speech and audio," in *Proc. ICASSP*, 2018.

[15] W. B. Kleijn, F. S. Lim, A. Luebs, J. Skoglund, F. Stimberg, Q. Wang, and T. C. Walters, "Wavenet based low rate speech coding," in *Proc. ICASSP*. IEEE, 2018, pp. 676–680.

[16] W. B. Kleijn, A. Storus, M. Chinen, T. Denton, F. S. C. Lim, A. Luebs, J. Skoglund, and H. Yeh, "Generative speech coding with predictive variance regularization," 2021.

[17] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014.

[18] T. Bäckström, M. B. Mansali, P. P. Zarazaga, M. Ranjit, S. Das, and Z. Lachiri, "Pyawnes-codec: Speech and audio codec for adhoc acoustic wireless sensor networks," in *Proc. EUSIPCO*, 2021.

[19] W. P. LeBlanc, B. Bhattacharya, S. A. Mahmoud, and V. Cuperman, "Efficient search and design procedures for robust multistage vq of lpc parameters for 4 kb/s speech coding," *IEEE Trans. Speech Audio Process.*, vol. 1, no. 4, pp. 373–385, 1993.

[20] T. Bäckström, "End-to-end optimization of source models for speech and audio coding using a machine learning framework," in *Proc. Interspeech*, 2019. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-128

[21] B. H. Juang and A. Gray, "Multiple stage vector quantization for speech coding," in *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 7, 1982, pp. 597–600.

[22] W. P. LeBlanc, S. A. Mahmoud, and V. Cuperman, *Joint Design of Multi-Stage VQ Codebooks for LSP Quantization with Applications to 4 kbit/s Speech Coding*. Springer, 1993.

[23] *TS 26.445, EVS Codec Detailed Algorithmic Description; 3GPP Technical Specification (Release 12)*, 2014.

[24] M. Bosi and R. E. Goldberg, *Introduction to Digital Audio Coding and Standards*. Kluwer Academic Publs., 2003.

[25] J. Rissanen and G. G. Langdon, "Arithmetic coding," *IBM Journal of research and development*, vol. 23, no. 2, pp. 149–162, 1979.

[26] T. Bäckström, G. Fuchs, M. Multrus, and M. Dietz, "Linear prediction based audio coding using improved probability distribution estimation," US Provisional Patent US 61/665 485, Jun., 2013.

[27] T. Bäckström, J. Fischer, and S. Das, "Dithered quantization for frequency-domain speech and audio coding," in *Proc. Interspeech*, 2018, pp. 3533–3537.

[28] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *Proc. ICASSP*, 2015, pp. 5206–5210.

[29] *Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs*, ITU-T Recommendation P.862, 2001. [Online]. Available: http://www.itu.int/rec/T-REC-P.862/en

[30] C. H. Taal, R. C. Hendriks, R. Heusdens, and J. Jensen, "An algorithm for intelligibility prediction of time–frequency weighted noisy speech," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 19 (7), no. 7, pp. 2125–2136, 2011.