LACE: A LIGHT-WEIGHT, CAUSAL MODEL FOR ENHANCING CODED SPEECH THROUGH ADAPTIVE CONVOLUTIONS

Jan Büthe, Jean-Marc Valin, Ahmed Mustafa

Amazon Web Services
Palo Alto, USA
{jbuethe, jmvalin, ahdmust}@amazon.com

ABSTRACT

Classical speech coding uses low-complexity postfilters with zero lookahead to enhance the quality of coded speech, but their effectiveness is limited by their simplicity. Deep Neural Networks (DNNs) can be much more effective, but require high complexity and model size, or added delay. We propose a DNN model that generates classical filter kernels on a per-frame basis with a model of just 300 K parameters and 100 MFLOPS complexity, which is a practical complexity for desktop or mobile device CPUs. The lack of added delay allows it to be integrated into the Opus codec, and we demonstrate that it enables effective wideband encoding for bitrates down to 6 kb/s.

Index Terms— speech enhancement, speech coding, opus

1. INTRODUCTION

Degradation of speech through coding is a persisting problem, especially in communication scenarios where delay and complexity are critical. Classical approaches to coded speech enhancement [1] are typically very low in complexity and do not require lookahead but are also of limited effectiveness due to the simplistic nature of the algorithms. A number of DNN based methods have recently been proposed. [2, 3, 4, 5, 6]. These are more effective but they are also either high in complexity or they require additional lookahead. The reason behind this is the more general observation that time-domain models for signal enhancement, which can operate in a causal manner, tend to be much larger and more complex than frequency domain methods, which in turn require lookahead for overlap-add [7]. This issue becomes even more pressing when the speech codec that is to be enhanced is embedded into a larger coding structure, which is often the case for modern codecs like Opus or EVS.

As a solution to this problem we propose the linear-adaptive coding enhancer (LACE), a light-weight causal model with only 300 K parameters and 100 MFLOPS complexity. The key idea behind LACE is to use adaptive convolutions with filter kernels computed from input features on a per-frame basis at inference time. This eliminates the need to have a large number of channels, which is typical for CNNs. We also base the architecture on classic postfilters [1], providing the model with a large but sparse receptive field where needed, which reduces complexity even further.

We test our model by applying it to the linear-predictive coding mode of the Opus codec, also known as SILK [8, 9][1]. Since LACE does not require lookahead and is essentially phase preserving, it can be directly integrated into the decoder while maintaining the seemless mode-switching capability. The model is bitrate-scalable, and we verify in a P.808 listening test that it significantly improves the baseline at 6, 9 and 12 kb/s. We also provide PESQ scores up to 22 kb/s, which demonstrate that LACE scales to transparency as the codec does. The low complexity and size also allow the model to run on both desktop and mobile device CPUs with insignificant overhead making it a practical and effective method.

Although we chose Opus as a test case for LACE, the model could be easily adapted to any speech codec that provides pitch information at the decoder side. Compared to fully neural codecs [10, 11, 12, 13, 14], this approach has the practical advantage of maintaining backward compatibility, leaving an inexpensive decoding option for low-end devices like microcontrollers.

[1] samples are available at https://282fd5fa7.github.io/LACE

2. LACE

The task of enhancing coded speech is loosely related to a denoising problem, i.e. recovering a clean signal $x(t)$ from a noisy mixture

$$y(t) = x(t) + n(t). \tag{1}$$

However, for a speech codec (or any other perceptual codec) the coding noise $n(t)$ will be closely related to the signal $x(t)$ itself to exploit masking properties. This means that recovering $x(t)$ from $y(t)$ is neither a feasible nor desirable task. In particular, training a model to minimize the mean-square error between enhanced signal and clean signal will result in losing large parts of the speech signal since noise in $x(t)$ is largely replaced by statistically similar noise in $y(t)$. We therefore state the task as a noise-reshaping task. Given the noisy mixture in (1) we want to produce an enhanced signal $\hat{y}(t)$ in which the coding noise is less audible.

Classical approaches [1] identify spectral valleys as main source for audible coding noise. These include both the narrow valleys between harmonics for voiced speech parts as well as wider valleys between formants, the peaks of the spectral envelope. The first task, inter-harmonic noise reduction, is classically addressed by a long term post filter, a comb-filter that makes explicit use of the pitch lag, emphasizing multiples of the fundamental frequency and attenuating frequencies in between. The second task, formant enhancement, is carried out with a short-term-filter, usually derived from the short-term linear prediction coefficients.

Combining these ideas naturally results in time-varying filtering model

$$\hat{y}(t) = \sum_{\tau=0}^{\infty} h(t,\tau)\, y(t-\tau), \tag{2}$$

which we take as starting point for our investigations. By computing these filters from input data we can not only apply formant and pitch enhancement to the coded signal but the model can also address

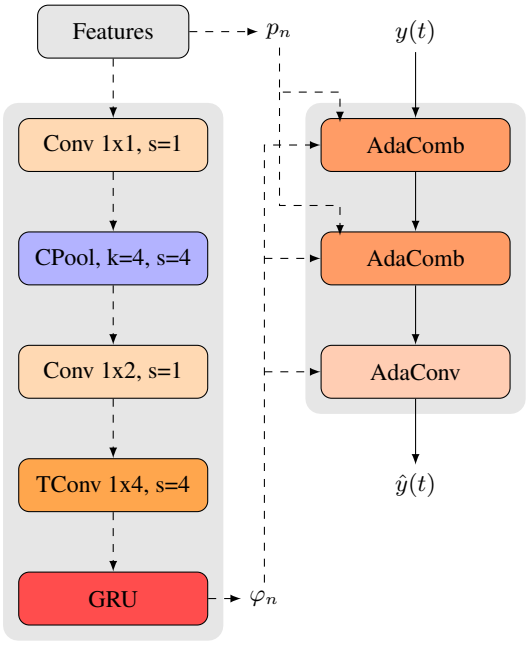

Figure 1: High-level overview of LACE model. The feature encoder on the left produces latent feature vectors φ_n every 5 ms. These feature vectors are used by the adaptive filtering modules in the signal path on the right to calculate convolution kernels. In addition, the comb-filtering modules make direct use of the pitch lags p_n.

temporal artifacts, which is clearly beyond the reach of classical post-filters, which are capable of only limited adaptation. The approach differs significantly from standard non-adaptive CNNs like the time-domain model in [2], which compensate for that lack of adaptation by using a large number of redundant channels, making the models both large and computationally expensive.

2.1. Model Overview

The LACE model implements a finite version of (2) with a varying filter length at most roughly twice the maximal pitch lag, which is depicted as signal path in Fig. 1. Computing the filter coefficients directly would be possible but inefficient. Instead we apply multiple consecutive (comb-)filtering modules which result in sparse filters for large pitch lags. The filtering modules are described in detail in section 2.5 .

The filtering modules in the signal path are steered with features derived from the Opus decoder which are transformed into a sequence of latent feature vectors φ_n by a feature encoder depicted in the left of Fig. 1. These latent feature vectors are calculated at a rate of 200 Hz, which corresponds to the sub-frame rate of the Opus linear-predictive coding mode. The feature processing is causal in the sense that it never uses information beyond the current 20-ms Opus frame.

2.2. Features

To keep complexity low and model size small, we base our model on a set of carefully hand-tuned features. As in [3], these are a mix of (quantized) clean speech features received by the Opus decoder and features calculated from the noisy decoded speech.

For clean speech features we use

1. LPC coefficients converted to a 64-band ERB-scale log-magnitude spectrogram at 10-ms granularity
2. Opus quantization gains in log-domain at 5-ms granularity
3. pitch lags at 5-ms granularity
4. 5 filter taps from the Opus LTP filter at 5-ms granularity

and for noisy speech features we use

1. 18-band cepstrum on 20-ms frames at a granularity of 10 ms following the Opus CELT bands
2. 5 auto-correlation values of $y(t)$ around the Opus pitch lag at a granularity of 5 ms

The features are selected to be largely scale-invariant, the two exceptions being the Opus frame gain and the constant term in the noisy speech cepstrum. The reason for this is that the filtering in (2) as a linear operation should not depend on the signal level. Furthermore, for simplicity all features are upsampled to 200 Hz by repetition.

2.3. Bitrate Signalling

Since the kind and strength of coding artifacts strongly depends on the bitrate, it is essential to provide this information to the LACE model. However, since the linear-predictive Opus mode is a variable bitrate encoder by design, the exact target bitrate specified at the encoder is not known to the decoder. We let the model infer the bitrate from the number of bits of the received Opus frames. Since these values give a very noisy estimate of the bitrate, we provide both the raw number of bits and an exponential moving average with an update rate on 0.1 to the feature encoder.

To make these features more useful, we pass them through an eight-dimensional saturating embedding whose components are given by

$$\sin\left(k\,\frac{2\log(\max\{A,\,\min\{B, n_{bits}\}\}) - \log(A\cdot B)}{\log(B/A)}\right) \qquad (3)$$

for $k = 1, 2, \ldots, 8$. The reasoning behind the saturation is that there is nothing to do for the model at very high bitrates. Hence, it suffices to add a single such high bitrate to the training data to ensure that the model behaves well outside of the intended range of use. We use $A = 50$ and $B = 650$, which correspond to bitrates of 2.5 and 32.5 kb/s.

2.4. Feature Encoding

Feature encoding is done by a sequential model of convolutions and transpose convolutions with $\tanh$ activations followed by a GRU to capture long-term dependencies. The first convolution performs scaling and dimensionality reduction on the 178-dimensional feature space (a 64-dimensional embedding is used for the pitch values), reducing the features to $N_r = 96$ channels. This is followed by a concatenative pooling layer, which combines the information from the four Opus subframes into one feature vector. This is followed by a second convolution, which adjusts the number of channels to the hidden feature dimension N_h and a transpose convolution with N_h channels that performs a factor 4 upsampling back to 200 Hz. Finally, the result is processed by a GRU with N_h hidden units to generate the N_h-dimensional hidden feature vectors φ_n. The pooling and upsampling provides the model with partial lookahead to the Opus frame boundary. The model is therefore causal on the Opus frame level but not causal on the Opus sub-frame level.

2.5. Adaptive Convolutions

The AdaptiveConv module in Fig. 1 filters the input signal with a sequence of FIR filters. It plays the role of formant enhancement in the LACE model.

The FIR filter coefficients are calculated on a per-frame basis from the hidden feature vectors φ_n as

$$h_n(\tau) = g_n \, \kappa_n(\tau), \tag{4}$$

where κ_n is the filter shape calculated as

$$\kappa_n = \frac{W_\kappa \varphi_n + b_\kappa}{\|W_\kappa \varphi_n + b_\kappa\|_2}. \tag{5}$$

and g_n is the filter gain calculated as

$$g_n = \exp(\alpha \tanh(W_g \varphi_n + b_g)) \tag{6}$$

with trainable projection matrices $W_\cdot$ and biases $b_\cdot$. Splitting the filter coefficients into shape and gain is similar to weight normalization [15] for regular convolutional layers but we also use it to limit the maximal amplification of the filters.

To avoid transition artifacts, the filters h_n are interpolated on the first half of the 5-ms frames using a half Hann window.

The AdaptiveComb modules in Fig. 1 are similar to the AdaptiveConv but the FIR filter taps are moved around the pitch delay p_n. Furthermore, they feature a second gain to control the comb-filtering strength, which is calculated as

$$\gamma_n = \exp(\beta - \mathrm{ReLU}(W_\gamma \varphi_n + b_\gamma)). \tag{7}$$

We use a ReLU activation to calculate γ_n since we only need a one-sided limitation on the log-scale to limit the comb-filtering strength. The transfer function on frame n is thus given by

$$H_n(z) = g_n \left(1 + \gamma_n z^{-p_n + \lfloor k/2 \rfloor} \sum_{\ell=0}^{k-1} \kappa_n(\ell) z^{-\ell}\right), \tag{8}$$

where k denotes the filter length. The filters are again interpolated with a half Hann window.

Comb filtering is only useful for voiced speech parts and Opus does not transmit a pitch lag for speech parts classified as unvoiced by the encoder. For such frames we set the pitch lag to $p_n = \lfloor k/2 \rfloor$, essentially turning the AdaptiveComb modules into AdaptiveConv modules. This limits the risk of adding coloration to unvoiced speech parts and increases the spectral shaping capability.

3. TRAINING

3.1. Data

We train our model on 165 hours of clean speech sampled at 16 kHz collected from multiple high-quality TTS datasets [16, 17, 18, 19, 20, 21, 22, 23, 24], which contains more than 900 speakers in 34 languages and dialects. The data is augmented using random scaling and random equalization. We obtain the coded signal with a modified version of libopus, restricting the encoder to wideband and linear-predictive mode only. Furthermore, we change the encoder parameters complexity, packet_loss_percent and bitrate randomly every 249 frames. Since Opus applies a high-pass filter to the clean signal before encoding in linear-predictive mode, we use that high-pass filtered clean signal as the target. Furthermore, we apply pre-emphasis with $P(z) = 1 - 0.85 z^{-1}$ to both the noisy input signal and the clean target signal. We choose hyper parameters $N_r = 96$, $N_h = 128$ and $k = 15$ which results in a model size of 306 K parameters and 99 MFLOPS complexity.

3.2. Loss

We use a mixture of regression losses to train the LACE model, which are tailored to different tasks. One loss is calculated in the time domain and the other two losses are calculated on STFTs with window size equal to DFT size and 50% overlap. The STFT losses are averaged over different resolutions with DFT sizes 2^n, $n = 5, 6, \ldots, 12$. All STFTs are calculated with Hann windows.

The LACE model is not intrinsically phase preserving. As a matter of fact the only zero-phase filter it can implement is the identity. We therefore bias towards phase preservation in periodic signal parts by applying a weighted squared error loss

$$\mathcal{L}_{\mathrm{phase}} = \frac{\|x - \hat{y}\|_2^2}{\|\hat{y}\|_2}. \tag{9}$$

The weighting by $1/\|\hat{y}\|_2$ modifies the L^2 loss behavior on unvoiced signal parts. We pointed out before, that the MSE loss will lead to loss of unvoiced signal parts since $x(t)$ and $y(t)$ will be largely uncorrelated. However, for uncorrelated $x(t)$ and $y(t)$ the weighted version (9) attains its global minimum at $\|x\|_2 = \|\hat{y}\|_2$ (as opposed to $\|\hat{y}\| = 0$ for the unweighted MSE loss) which makes the loss energy preserving in this case.

For envelope reconstruction, we apply a set of perceptually motivated filters to smooth the absolute values of the STFT coefficients. The filters are approximations to the auditory filters as described in [25] following an ERB scale. As envelope loss we use the L1 loss on the resulting smooth spectrograms X_s and $\hat{Y}_s$, i.e.

$$\mathcal{L}_{\mathrm{env}} = \|\log(X_s) - \log(\hat{Y}_s)\|_1. \tag{10}$$

To restore the harmonic structure, we use a modification of the spectral convergence loss, where we replace the Frobenius norm by a cross-correlation based loss, which makes it insensitive to signal scale. The cross-correlation is calculated both over time and frequency, giving

$$\mathcal{L}_{\mathrm{spec}} = 1 - \frac{\sum_{n,k} \left|X(n,k)\right| \left|\hat{Y}(n,k)\right|}{\left(\sum_{n,k} \left|X(n,k)\right|^2 \sum_{n,k} \left|\hat{Y}(n,k)\right|^2\right)^{1/2}}. \tag{11}$$

As total loss we use

$$\mathcal{L}_{\mathrm{total}} = 10\,\mathcal{L}_{\mathrm{phase}} + 2\,\mathcal{L}_{\mathrm{env}} + \mathcal{L}_{\mathrm{spec}} \tag{12}$$

3.3. Training Details

We train the model using the Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ on sequences of 0.5 s length with a mini-batch size of 256 and an initial learning rate $\lambda = 5 \times 10^{-4}$ for 50 epochs or ≈ 230 steps. We use learning rate decay with a factor 2.5×10^{-5}, i.e. in step μ the weights are updated with a learning rate $\lambda/(1 + 2.5 \times 10^{-5}\,\mu)$.

4. EVALUATION

4.1. Frequency Response Analysis

Since the LACE signal path is linear we can study local model behavior by analysing frequency responses. Fig. 2 shows a series of frequency responses for a voiced frame. For low bitrates LACE performs strong comb filtering and for higher bitrates it approaches the identity function, which is the intended behavior.

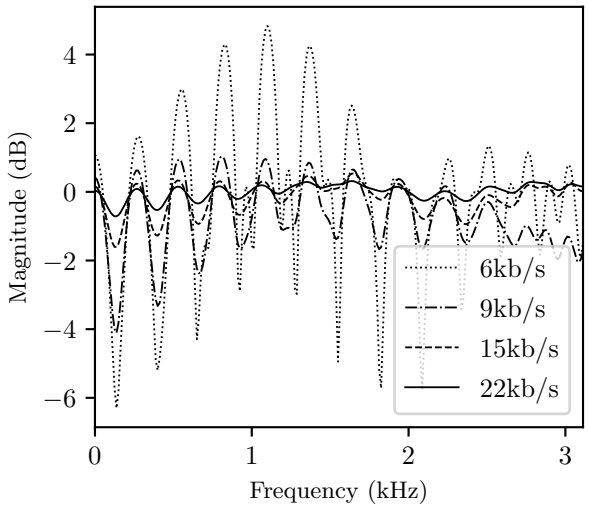

Figure 2: Frequency responses of the LACE signal path for a voiced frame at multiple bitrates

4.2. Listening Test

We evaluated the quality of LACE using 192 clean English speech clips from the NTT Multi-Lingual Speech Database for Telephonometry, which was not included in the training data. Using the crowd-sourcing methodolgy from ITU-R P.808 [19], we tested Opus with and without LACE at 6, 9, and 12 kb/s.

As a benchmark for causal coded speech enhancement, we included the TDCNN time domain method from [2]. Since the model is not designed to operate with multiple bitrates, we restrict training and comparison to 6 kb/s. We adapted the TDCNN training setup and hyper-parameters, increasing its PESQ score by 0.2, and lowering its complexity. We used $N = 15$, $F = 55$, and $L = 320$, and improved the loss function by using (12) instead of MSE.

Although it cannot be used in a real-time Opus implementation due to its 25 ms delay, we also included the non-causal LPCNet resynthesis method from [3] as a comparison point. Such resynthesis methods are also limited to enhancing lower bitrates (6 kb/s in this case), since their output quality is bounded by the quality of the vocoder, which prevents them from scaling to transparency. The exact bitrate threshold, however, will depend on the vocoder.

Results in Fig. 3 show that LACE significantly improves Opus at all tested bitrates. At 6 kb/s LACE outperforms TDCNN by a large margin and achieves about 60% of the quality improvement of the non-causal resynthesis method. LACE has a total complexity of 100 MFLOPS (0.1 GFLOPS), far lower than the 3 GFLOPS required for resynthesis and the 16 GFLOPS of TDCNN.

4.3. Objective Evaluation

We use PESQ to test performance of LACE from 6 to 22 kb/s. As can be seen in Fig. 4 LACE always outperforms the baseline (Opus). Quantitatively, the PESQ improvement is in line with the listening test results at 6, 9 and 12 kb/s.

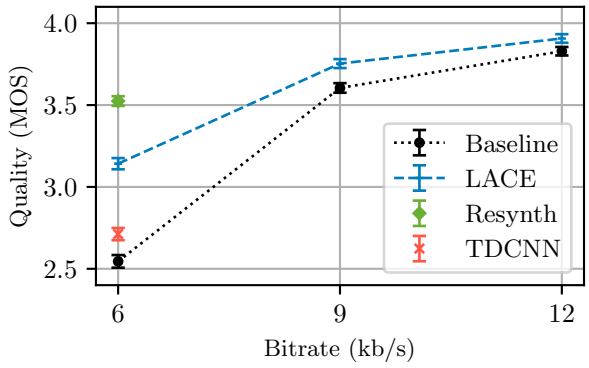

Figure 3: P.808 results. The clean signal has a MOS of 4.01 ± 0.03. LACE consistently outperforms the baseline and TDCNN and achieves about 60% of the MOS improvement of LPCNet resynthesis at 6kb/s which requires 25ms delay at 4x the size and 30x the complexity compared to LACE.

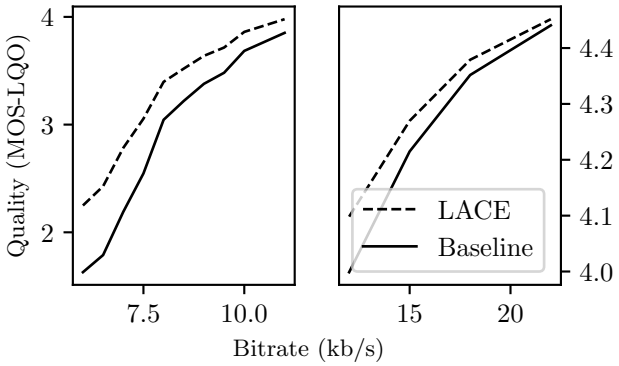

Figure 4: PESQ scores from 6 to 22 kb/s.

5. CONCLUSION

We have demonstrated that adaptive convolutions give rise to a very efficient and effective enhancer for coded speech. LACE consistently improves the test codec over a large bitrate range, demonstrated both by objective and subjective methods, and it outperforms state-of-the-art causal methods by a large margin. Since LACE is both causal and essentially phase preserving, it can be directly integrated into codecs with dedicated speech-coding modes. We believe such a low-complexity enhancement algorithm will be most useful for enhancing the quality of existing classical speech codecs without breaking compatibility.

6. ACKNOWLEDGMENT

The authors would like to thank Timothy Terriberry, Paris Smaragdis and Mike Goodwin for helpful suggestions.

7. REFERENCES

[1] J.-H. Chen and A. Gersho, "Adaptive Postfiltering for Quality Enhancement of Coded Speech," *IEEE Transactions on Speech and Audio Processing*, vol. 3, no. 1, pp. 59–71, 1995.

[2] Z. Zhao, H. Liu, and T. Fingscheidt, "Convolutional Neural Networks to Enhance Coded Speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 4, pp. 663–678, 2019.

[3] J. Skoglund and J.-M. Valin, "Improving Opus Low Bit Rate Quality with Neural Speech Synthesis," in *Proc. INTERSPEECH*, 2019.

[4] K. Gupta, S. Korse, B. Edler, and G. Fuchs, "A DNN Based Post-Filter to Enhance the Quality of Coded Speech in MDCT Domain," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 836–840.

[5] S. Korse, K. Gupta, and G. Fuchs, "Enhancement of Coded Speech Using a Mask-Based Post-Filter," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 6764–6768.

[6] S. Korse, N. Pia, K. Gupta, and G. Fuchs, "PostGAN: A GAN-Based Post-Processor to Enhance the Quality of Coded Speech," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 831–835.

[7] P. Ochieng, "Deep Neural Network Techniques for Monaural Speech Enhancement: State of the Art Analysis," *arXiv:2212.00369*, 2022.

[8] K. Vos, K. Sørensen, S. Jensen, and J.-M. Valin, "Voice Coding with Opus," *135th AES Convention*, pp. 722–731, 2013.

[9] J.-M. Valin, K. Vos, and T. B. Terriberry, "Definition of the Opus Audio Codec," RFC 6716, 2012.

[10] J.-M. Valin and J. Skoglund, "A Real-Time Wideband Neural Vocoder at 1.6kb/s Using LPCNet," in *Proc. INTERSPEECH*, 2019, pp. 3406–3410.

[11] 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," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 6478–6482.

[12] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "SoundStream: An End-to-End Neural Audio Codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2022.

[13] N. Pia, K. Gupta, S. Korse, M. Multrus, and G. Fuchs, "NESC: Robust Neural End-2-End Speech Coding with GANs," in *Proc. INTERSPEECH*, 2022.

[14] T. Jenrungrot, M. Chinen, W. Kleijn, J. Skoglund, Z. Borsos, N. Zeghidour, and M. Tagliasacchi, "Lmcodec: A low bitrate speech codec with causal transformer models," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023.

[15] T. Salimans and D. P. Kingma, "Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks," in *NIPS*, 2016.

[16] I. Demirsahin, O. Kjartansson, A. Gutkin, and C. Rivera, "Open-source Multi-speaker Corpora of the English Accents in the British Isles," in *Proc. LREC*, 2020.

[17] O. Kjartansson, A. Gutkin, A. Butryna, I. Demirsahin, and C. Rivera, "Open-Source High Quality Speech Datasets for Basque, Catalan and Galician," in *Proc. SLTU and CCURL*, 2020.

[18] K. Sodimana, K. Pipatsrisawat, L. Ha, M. Jansche, O. Kjartansson, P. D. Silva, and S. Sarin, "A Step-by-Step Process for Building TTS Voices Using Open Source Data and Framework for Bangla, Javanese, Khmer, Nepali, Sinhala, and Sundanese," in *Proc. SLTU*, 2018.

[19] A. Guevara-Rukoz, I. Demirsahin, F. He, S.-H. C. Chu, S. Sarin, K. Pipatsrisawat, A. Gutkin, A. Butryna, and O. Kjartansson, "Crowdsourcing Latin American Spanish for Low-Resource Text-to-Speech," in *Proc. LREC*, 2020.

[20] F. He, S.-H. C. Chu, O. Kjartansson, C. Rivera, A. Katanova, A. Gutkin, I. Demirsahin, C. Johny, M. Jansche, S. Sarin, and K. Pipatsrisawat, "Open-source Multi-speaker Speech Corpora for Building Gujarati, Kannada, Malayalam, Marathi, Tamil and Telugu Speech Synthesis Systems," in *Proc. LREC*, 2020.

[21] Y. M. Oo, T. Wattanavekin, C. Li, P. De Silva, S. Sarin, K. Pipatsrisawat, M. Jansche, O. Kjartansson, and A. Gutkin, "Burmese Speech Corpus, Finite-State Text Normalization and Pronunciation Grammars with an Application to Text-to-Speech," in *Proc. LREC*, 2020.

[22] D. van Niekerk, C. van Heerden, M. Davel, N. Kleynhans, O. Kjartansson, M. Jansche, and L. Ha, "Rapid development of TTS corpora for four South African languages," in *Proc. INTERSPEECH*, 2017.

[23] A. Gutkin, I. Demirşahin, O. Kjartansson, C. Rivera, and K. Túbọ̀sún, "Developing an Open-Source Corpus of Yoruba Speech," in *Proc. INTERSPEECH*, 2020.

[24] E. Bakhturina, V. Lavrukhin, B. Ginsburg, and Y. Zhang, "Hi-Fi Multi-Speaker English TTS Dataset," in *Proc. INTERSPEECH*, 2021, pp. 2776–2780.

[25] B. Moore, *An Introduction to the Psychology of Hearing*. Brill, 2012.