arXiv:2103.14776v1 [eess.AS] 27 Mar 2021

Scalable and Efficient Neural Speech Coding

Kai Zhen, *Student Member, IEEE,* Jongmo Sung, Mi Suk Lee, Seungkwon Beak, Minje Kim, *Senior Member, IEEE,*

***Abstract*—This work presents a scalable and efficient neural waveform codec (NWC) for speech compression. We formulate the speech coding problem as an autoencoding task, where a convolutional neural network (CNN) performs encoding and decoding as its feedforward routine. The proposed CNN autoencoder also defines quantization and entropy coding as a trainable module, so the coding artifacts and bitrate control are handled during the optimization process. We achieve efficiency by introducing compact model architectures to our fully convolutional network model, such as gated residual networks and depthwise separable convolution. Furthermore, the proposed models are with a scalable architecture, cross-module residual learning (CMRL), to cover a wide range of bitrates. To this end, we employ the residual coding concept to concatenate multiple NWC autoencoding modules, where an NWC module performs residual coding to restore any reconstruction loss that its preceding modules have created. CMRL can scale down to cover lower bitrates as well, for which it employs linear predictive coding (LPC) module as its first autoencoder. Once again, instead of a mere concatenation of LPC and NWC, we redefine LPC's quantization as a trainable module to enhance the bit allocation tradeoff between LPC and its following NWC modules. Compared to the other autoregressive decoder-based neural speech coders, our decoder has significantly smaller architecture, e.g., with only 0.12 million parameters, more than 100 times smaller than a WaveNet decoder. Compared to the LPCNet-based speech codec, which leverages the speech production model to reduce the network complexity in low bitrates, ours can scale up to higher bitrates to achieve transparent performance. Our lightweight neural speech coding model achieves comparable subjective scores against AMR-WB at the low bitrate range and provides transparent coding quality at 32 kbps.**

***Index Terms*—Neural speech coding, waveform coding, representation learning, model complexity**

I. Introduction

SPEECH coding can be implemented as an encoder-decoder system, whose goal is to compress input speech signals into the compact bitstream (encoder) and then to reconstruct the original speech from the code with the least possible quality degradation. Speech coding facilitates telecommunication and saves data storage among many other applications. There is a typical trade-off a speech codec must handle: the more the system reduces the amount of bits per second (bitrate), the worse the perceptual similarity between the original and recovered signals is likely to be perceived. In addition, the speech coding systems are often required to maintain an affordable computational complexity when the hardware resource is at a premium.

For decades, speech coding has been intensively studied yielding various standardized codecs that can be categorized into two types: the vocoders and waveform codecs. A vocoder, also referred to as parametric speech coding, distills a set of physiologically salient features, such as the spectral envelope (equivalent to vocal tract responses including the the contribution from mouth shape, tongue position and nasal cavity), fundamental frequencies, and gain (voicing level), from which the decoder *synthesizes* the speech. Typically, a vocoder is computationally efficient, but it usually operates in the narrowband mode due to its limited performance [1][2]. A waveform codec aims to perfectly reconstruct the speech signal, which features up-to-transparent quality with a higher bitrate range. The latter can be generalized to non-speech audio signal compression as it is not restricted to those speech production priors, although waveform coding and parametric coding can be coupled for a hybrid design [3][4][5].

Under the notion of unsupervised speech representation learning, deep neural network (DNN)-based codecs have revitalized the speech coding problem and provided different perspectives. The major motivation of employing neural networks to speech coding is twofold: to fill the performance gap between vocoders and waveform codecs towards a near-transparent speech synthesis quality; to use its trainable encoder and learn latent representations which may benefit other DNN-implemented downstream applications, such as speech enhancement [6][7], speaker identification [8] and automatic speech recognition [9][10]. Having that, a neural codec can serve as a trainable acoustic unit integrated in future digital signal processing engines [11].

Recently proposed neural speech codecs have achieved high coding gain and reasonable quality by employing deep autoregressive models. The superior speech synthesis performance achieved in WaveNet-based models [12] has successfully transferred to neural speech coding systems, such as in [13], where WaveNet serves as a decoder synthesizing wideband speech samples from a conventional non-trainable encoder at 2.4 kbps. Although its reconstruction quality is comparable to waveform codecs at higher bitrates, the computational cost is significant due to the model size of over 20 million parameters.

Meanwhile, VQ-VAE [14] integrates a trainable vector quantization scheme into the variational autoencoder (VAE)

This work was supported by the Institute for Information and Communications Technology Promotion (IITP) funded by the Korea government (MSIT) under Grant 2017-0-00072 (Development of Audio/Video Coding and Light Field Media Fundamental Technologies for Ultra Realistic Tera-Media). Kai Zhen is with the Department of Computer Science and Cognitive Science Program at Indiana University, Bloomington, IN 47408 USA. Jongmo Sung, Mi Suk Lee, and Seungkwon Beack are with Electronics and Telecommunications Research Institute, Daejeon, Korea 34129. Minje Kim is with the Dept. of Intelligent Systems Engineering at Indiana University (e-mails: zhenk@iu.edu, lms@etri.re.kr, jmseong@etri.re.kr, skbeack@etri.re.kr, minje@indiana.edu).

Manuscript received March XX, 2021; revised YYY ZZZ, 2021.

TABLE I: Categorical summary of recently proposed neural speech coding systems. ✔means the system features the characteristic. ✗means not and ●means not reported.

	WaveNet [13]	VQ-VAE [16]	LPCNet [17]	Proposed
Transparent coding	✔	●	✗	✔
Less than 1M parameters	✗	✗	✔	✔
Real time communications	✗	✗	✔	✔
Encoder trainable	✔	✔	✗	✔

[15] for discrete speech representation learning. While the bitrate can be lowered by reducing the sampling rate 64 times, the downside for VQ-VAE is that the prosody can be significantly altered. Although [16] provides a scheme to pass the pitch and timing information to the decoder as a remedy, it does not generalize to non-speech signals. More importantly, VQ-VAE as a vocoder does not address the complexity issue since it uses WaveNet as the decoder. Although these neural speech synthesis systems noticeably improve the speech quality at low bitrates, they are not feasible for real-time speech coding on the hardware with limited memory and bandwidth.

LPCNet [17] focuses on efficient neural speech coding via a WaveRNN [18] decoder by leveraging the traditional linear predictive coding (LPC) techniques. The input of the LPCNet is formed by 20 parameters (18 Bark scaled cepstral coefficients and 2 additional parameters for the pitch information) for every 10 millisecond frame. All these parameters are extracted from the non-trainable encoder, and vector-quantized with a fixed codebook. As discussed previously, since LPCNet functions as a vocoder, the decoded speech quality is not considered transparent [19].

In this paper, we propose a novel neural waveform coding model, serving as a trainable acoustic processing unit with a lightweight design and scalable performance. In Sec. II, we introduce our basic model: a compact neural waveform codec with only 0.35 million parameters, much more lightweight than WaveNet, VQ-VAE and our previous neural codec [20]. Based on this neural codec, we introduce two mechanisms to integrate speech production theory and residual coding techniques in Sec. III. First, benefited from the residual-excited linear prediction (RELP) [21], we conduct LPC and apply the neural waveform codec to the excitation signal, which is illustrated in Sec.III-A. In this integration, a trainable soft-to-hard quantizer bridges the encoding of linear spectral pairs and the corresponding LPC residual, making the entire quantization and entropy coding modules trainable. Second, to scale up the performance for high bitrates, we propose cross-module residual learning (CMRL), a cascaded model architecture that adds up neural codecs for residual coding, sequentially (Sec.III-B). As a result, the proposed neural speech coding systems have following characteristics:

- Scalability: Similar to LPCNet [17], the proposed scheme is compatible with conventional spectral envelope estimation techniques. However, ours operates at a much wider bitrate range with comparable or superior speech quality to standardized waveform codecs.
- Compactness: Having achieved the superior speech quality, the model is with a much lower complexity than WaveNet [12] and VQ-VAE [14] based codecs. Our decoder contains only 0.12 million parameters which is 100 times more compact than a WaveNet counterpart. The execution time to encode and decode a signal is only 42.44% of its duration on a single-core CPU, which facilitates real-time communications.
- Trainability: Our method is with a trainable encoder as in VQ-VAE, which can be integrated into other DNNs for acoustic signal processing. Besides, it is not constrained to speech, and can be generalized to audio coding with minimal effort as shown in [22].

TABLE I highlights the comparison.

In Sec.IV, both objective comparisons and subjective listening tests are conducted for model evaluation. With a trainable quantizer for the LPC coefficients, the neural codec compresses the residual signal, showing noticeable performance gain, which outperforms Opus and is on a par with AMR-WB at lower bitrates; our codec is slightly superior to AMR-WB and Opus at higher bitrates when operating at 20 kbps; at 32 kbps, our codec is capable of scaling up to near transparency when residual coding among neural codecs is enabled in CMRL. Additionally, we investigate the effect of various blending ratios of loss terms and bit allocation schemes on the experimental result via the ablation analysis. The execution time and delay analysis is given under 4 hardware specifications, too. We conclude in Sec. V.

II. End-to-End Neural Waveform Codec (NWC)

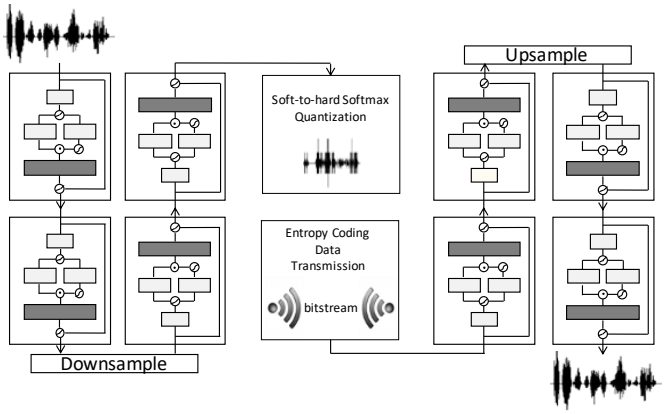
The neural waveform codec (NWC), is an end-to-end autoencoder that forms the base of our proposed coding systems. It directly encodes and quantizes the input waveform $\boldsymbol{x} \in \mathbb{R}^T$ into the bitstream $\tilde{\boldsymbol{h}} \in \mathbb{R}^N$ using a convolutional neural network (CNN) encoder module, and then reconstructs the output waveform using a decoder with a similar topology:

$$\boldsymbol{x} \approx \hat{\boldsymbol{x}} \leftarrow \mathcal{F}_{\text{dec}}(\tilde{\boldsymbol{h}}), \quad \tilde{\boldsymbol{h}} \leftarrow \mathcal{Q}(\boldsymbol{h}), \quad \boldsymbol{h} \leftarrow \mathcal{F}_{\text{enc}}(\boldsymbol{x}). \quad (1)$$

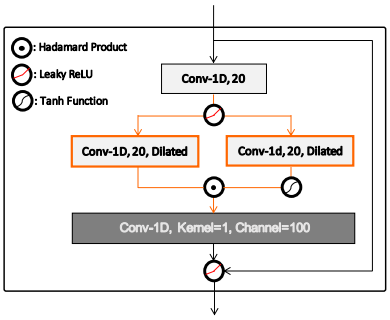
Fig. 1 (a) depicts NWC's overall system architecture. The structure is detailed in TABLE II. It serves as a basic component in the proposed speech coding system. In Sec. III-A and III-B we introduce our scaling mechanism using NWC as the building block of the CMRL framework.

NWC is defined with a compact architecture while achieving high reconstruction quality. The codec is a fully convolutional network (FCN) defined with 1-D convolutional layers. Both encoder and decoder adopt gated linear units (GLU) [23], which is based on the ResNet's cross-layer residual learning [24] with dilation [25] to achieve a compact model architecture and to expand the receptive field in the time domain. The gating mechanism (Fig. 1 (b)) boosts the gradient flow with superior performance as evidenced in [26].

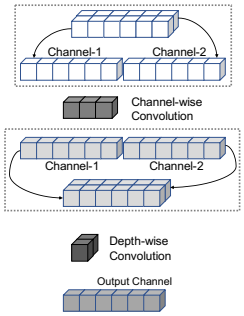
The depthwise separable convolution [27] is used to further save up the computational cost in one of the decoder layers (Fig. 1 (c)). For example, to transform a feature map of size



(a) The high-level structure of proposed neural waveform codec



(b) Dilated gated linear unit (GLU)



(c) Separable convolution

Fig. 1: Proposed lightweight neural waveform codec.

TABLE II: Architecture of the neural waveform codec: input and output tensors are shaped as (sample, channel), while the kernel is represented as (kernel size, in channel, out channel).

Layer	Input shape	Kernel shape	Output shape
Channel expansion	(512, 1)	(55, 1, 100)	(512, 100)
Gated linear unit	(512, 100)	[(1, 100, 20), (15, 20, 20)†, (15, 20, 20)†, (9, 20, 100)] ×2	(512, 100)
Downsampling	(512, 100)	(9, 100, 100)	(256, 100)
Gated linear unit	(512, 100)	[(1, 100, 20), (15, 20, 20)†, (15, 20, 20)†, (9, 20, 100)] ×2	(512, 100)
Channel reduction	(256, 100)	(9, 100, 1)	(256, 1)
Channel expansion	(256, 1)	(9, 1, 100)	(256, 100)
Gated linear unit	(256, 100)	[(1, 100, 20), (15, 20, 20)†, (15, 20, 20)†, (9, 20, 100)] ×2	(256, 100)
Upsampling	(256, 100)	(9, 100, 1), (1, 100, 100)	(512, 50)
Gated linear unit	(512, 50)	[(1, 50, 20), (15, 20, 20)†, (15, 20, 20)†, (9, 20, 50)] ×2	(512, 50)
Channel reduction	(512, 50)	(55, 50, 1)	(512, 1)

256×100 (features, channels) into the same-sized tensor, we need a kernel of size $c \times 100 \times 100$ (features, input channels, output channels). With the depthwise separable convolution, it first conducts channel-wise convolution with a kernel shape of $c \times 1 \times 1$ on every input channel, respectively, thus requiring 100 such kernels. It results in a 256×100 tensor. A depthwise Hamdard product with the kernel shape of $1 \times 100 \times 100$ follows. It is easy to show that $c \times 100 \times 100 > c \times 100 + 100 \times 100$ for a small c.

The proposed NWC achieves compression through two strategies: feature map compression and trainable quantization.

A. Feature Map Compression

One way to compress the input signal in the proposed encoder architecture is to reduce the data rate. The CNN encoder function takes an input frame, $\boldsymbol{x} \in \mathbb{R}^T$, and converts it into a feature map $\boldsymbol{h} \in \mathbb{R}^N$,

$$\boldsymbol{h} \leftarrow \mathcal{F}_{\text{enc}}(\boldsymbol{x}), \tag{2}$$

which then goes through quantization, transmission, and decoding to recover the input as shown in Fig. 1 (a). During the encoding process, we introduce a *downsampling* operation, reducing the dimension of the code vector $\boldsymbol{h}$. We employ a dedicated downsampling layer by setting up the stride value to be 2 during its convolution, reducing the data rate by 50%, i.e., $N = T/2$. Accordingly, the decoder needs a corresponding upsampling operation to recover the original sampling rate. We use subpixel CNN layer proposed in [28] to recover the original sampling rate. Concretely, the subpixel upsampling involves a feature transformation implemented in depthwise convolution, and a shuffle operation that interlaces features from two channels into a single channel, as shown in Eq. (3), where the input feature of the shuffle operation is shaped as $(N, 2)$ and the output is shaped as $(2N, 1)$.

$$\begin{aligned}&[h_{11}, h_{21}, h_{12}, h_{22}, \ldots, h_{1N}, h_{2N}] \\ &\leftarrow \text{Upsampling}([h_{11}, h_{12}, \ldots, h_{1N}; h_{21}, h_{22}, \ldots, h_{2N}]\end{aligned} \tag{3}$$

B. The Trainable Quantizer for Bit Depth Reduction

The dimension-reduced feature map can be further compressed via bit depth reduction. Hence, the floating-point code $\boldsymbol{h}$ goes through quantization and entropy coding, which will finalize the bitrate based on the entropy of the code value distribution. Typically, a bit depth reduction procedure lowers the average amount of bits to represent each sample. In our case, we could employ a quantization process that assigns the output of the encoder to one of the pre-defined quantization bins. If there are $2^5 = 32$ quantization bins, for example, a single-precision floating-point value's bit depth reduces from 32 to 5. In addition, various entropy coding techniques, such as Haffman coding, can be further employed to losslessly reduce the bit depth. While the quantization could be done in a traditional way, e.g., using Lloyds-Max quantization [29] *after* the neural codec is fully trained, we encompass the quantization step as a trainable part of the neural network as

Algorithm 1 Trainable Softmax quantization, $\mathcal{Q}(\boldsymbol{h}, \alpha, \boldsymbol{\beta})$

Input: the code, e.g., the encoder output, $\boldsymbol{h} = \mathcal{F}_{\text{enc}}(\boldsymbol{x})$
the Softmax scaling factor, α
the centroid vector, $\boldsymbol{\beta} \in \mathbb{R}^K$
Output: the quantized code, $\hat{\boldsymbol{h}}$ (training) or $\tilde{\boldsymbol{h}}$ (testing)
Compute the dissimilarity matrix: $\boldsymbol{D}_{nk} \leftarrow \ell_2(h_n || \beta_k)$
Softmax conversion: $\boldsymbol{A}^{(\text{soft})}_{n:} \leftarrow \text{Softmax}(-\alpha \boldsymbol{D}_{n:})$
if Training **then**
    Soft quantization: $\hat{\boldsymbol{h}} \leftarrow \boldsymbol{A}^{(\text{soft})}\boldsymbol{\beta}$
else if Testing **then**
    Hard quantization: $\tilde{\boldsymbol{h}} \leftarrow \boldsymbol{A}^{(\text{hard})}\boldsymbol{\beta}$
end if

proposed in [30]. Consequently, we expect that the codec is aware of the quantization error, which the training procedure tries to reduce it. It is also convenient to control the bitrate by controlling the entropy of the code value distribution, which can be also done as a part of network training.

In NWC, the quantization process is represented as classification on each scalar value of the encoder output. Given a vector with K centroids, $\boldsymbol{\beta} = [\beta_1, \beta_2, \cdots, \beta_K]^\top$, the quantizer's goal is to assign each feature h_n to the closest centroid in terms of ℓ_2 distance, which is defined as follows:

$$\boldsymbol{D} = \begin{bmatrix} ||h_1 - \beta_1||_2 & \cdots & ||h_1 - \beta_K||_2 \\ \vdots & \ddots & \vdots \\ ||h_N - \beta_1||_2 & \cdots & ||h_N - \beta_K||_2 \end{bmatrix}, \tag{4}$$

where n-th row in $\boldsymbol{D}$ is a vector of ℓ_2 distance between n-th code value h_n to all K quantization bins. Then, we employ the softmax function to turn each row of $\boldsymbol{D}$ into a K-dimensional probabilistic assignment vector:

$$\boldsymbol{A}^{(\text{soft})} = \begin{bmatrix} \text{softmax}(-\alpha \boldsymbol{D}_{1:}) \\ \text{softmax}(-\alpha \boldsymbol{D}_{2:}) \\ \vdots \\ \text{softmax}(-\alpha \boldsymbol{D}_{N:}) \end{bmatrix}, \tag{5}$$

where we turn the distance into a similarity metric by multiplying a negative number $-\alpha$, such that the shortest distance is converted to the largest probability.

Note that Eq. (5) yields a soft assignment matrix $\boldsymbol{A}^{(\text{soft})} \in \mathbb{R}^{N \times K}$. In practice, though, the quantization process must perform a hard assignment, so each code value h_n is replaced by an integer index to the closest centroids: $z_n \in \{1, 2, \cdots, K\}$, which is represented by $\lceil \log_2 K \rceil$ bits as the quantization result. The hard kernel assignment matrix $\boldsymbol{A}^{(\text{hard})}$, where each row is a one-hot vector, can be induced by turning on the maximum element of $\boldsymbol{A}^{(\text{soft})}$ while suppressing the non-maximum:

$$\boldsymbol{A}^{(\text{hard})}_{nk} = \begin{cases} 1 & \text{if } \arg\max_{j \in \{1,2,\cdots,K\}} \boldsymbol{A}^{(\text{soft})}_{nj} = k \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

On the decoder side, $\tilde{\boldsymbol{h}} = \boldsymbol{A}^{(\text{hard})}\boldsymbol{\beta}$ recovers $\boldsymbol{h}$.

Since $\arg\max$ operation in Eq. (6) is not differentiable, a soft-to-hard scheme is proposed in [30], where $\boldsymbol{A}^{(\text{hard})}$ is used only at test time. During backpropagation for training, the soft classification mode is enabled with $\boldsymbol{A}^{(\text{soft})}$ so as not to block the gradient flow. In other words, $\hat{\boldsymbol{h}} = \boldsymbol{A}^{(\text{soft})}\boldsymbol{\beta}$ represents each encoder output with a linear combination of all quantization bins. The process is summarized in Algorithm 1. Although this soft quantization process is differentiable and desirable during training, the discrepancy between $\boldsymbol{A}^{(\text{soft})}$ and $\boldsymbol{A}^{(\text{hard})}$ creates higher error during the test time, requiring a mechanism to reduce the discrepancy as in the following section.

1) Soft-to-hard quantization penalty: Although the limit of $\boldsymbol{A}^{(\text{soft})}$ is $\boldsymbol{A}^{(\text{hard})}$ as α approaches ∞, the change of α should be gradual to allow gradient flows in the initial phase of training. We control the *hardness* of $\boldsymbol{A}^{(\text{soft})}$ using the soft-to-hard quantization loss derived from [31]:

$$\mathcal{L}_{\text{Q}} = \frac{1}{N} \sum_{n,k} \sqrt{\boldsymbol{A}^{(\text{soft})}_{nk}}, \tag{7}$$

whose minimum, 1, is achieved when $\boldsymbol{A}^{(\text{soft})}_{n:}$ is a one-hot vector for all n. Conversely, when $\boldsymbol{A}^{(\text{soft})}_{nk} = 1/K$, the loss is maximum. Hence, by minimizing this soft-to-hard quantization penalty term, we can regularize the model to have *harder* $\boldsymbol{A}^{(\text{soft})}$ values by updating α and the other model parameters accordingly. As a result, the test time quantization loss will be reasonably small when $\boldsymbol{A}^{(\text{soft})}$ is replaced by $\boldsymbol{A}^{(\text{hard})}$.

2) Bitrate calculation and entropy control: The bitrate is calculated as a product of the number of code values per second and the average bit depth for each code. The former is defined by the dimension of the code vector N multiplied by the number of frames per second, $\frac{F}{T-o}$, where T, o, and F are the input frame size, overlap size, and the original sampling rate, respectively. If we denote the average bit depths per sample by a function $g(\tilde{h}_n)$, the bitrate can be computed as in Eq. (8),

$$\text{bitrate} = g(\tilde{h}_n) N F / (T - o). \tag{8}$$

When $F = 16,000$, $T = 512$, $o = 32$, and $N = 256$ after downsampling, for example, there are about $8,533$ samples per second. If $g(\tilde{h}_n) = 3$ bits, the bitrate is estimated as 25.6 kbps. On the contrary, the uncompressed bitrate is 256 kbps because $N = T = 512$, $o = 0$, and $g(x_t) = 16$ bits.

We adjust the entropy of $\boldsymbol{\beta}$ to indirectly control the codec's bitrate, because the entropy serves as the lower bound of $g(\tilde{h}_n)$ based on Shannon's entropy theory. We first estimate the entropy using the sample distribution,

$$\mathcal{H}(\boldsymbol{\beta}) \approx -\sum_{k=1}^{K} p(\beta_k) \log_2 p(\beta_k), \tag{9}$$

where $p(\beta_k) = \frac{1}{N} \sum_n \boldsymbol{A}^{(\text{hard})}_{nk}$ is the relative frequency of the k-th centroid being chosen during quantization. It is only an estimate of the true entropy, because it depends on the quality of the sample distribution $p(\beta_k)$.

To navigate the model training towards the target bitrate, $\mathcal{H}(\boldsymbol{\beta})$ defined in Eq. (9) is included to the loss function as a regularizer: its smaller value leads to a lower bitrate, and vice versa. However, during training, we approximate the relative frequency $p(\beta_k)$ using the soft assignment matrix $\boldsymbol{A}^{(\text{soft})}$ rather

than the hard one $\boldsymbol{A}^{(\text{hard})}$, i.e., $p(\beta_k) \approx \frac{1}{N}\sum_n \boldsymbol{A}^{(\text{soft})}_{nk}$, due to the discrete nature of $\boldsymbol{A}^{(\text{hard})}$ that prevents gradient-based updates. Since $\mathcal{H}(\boldsymbol{\beta})$ is parameterized by $\boldsymbol{A}^{(\text{soft})}$ and $\boldsymbol{\beta}$, their optimal values are learned during training, making the entire quantization process trainable. For example, if the current codec's bitrate is higher than desired, the optimization process will make the regularization effect stronger to lower the entropy, i.e., to increase the "spikiness" of the distribution. More details on the training process is discussed in Sec. IV-B

III. The Proposed Scalable NWC Models

By having NWC introduced in Sec. II as the basic module, we propose two different extension mechanisms to improve the codec's performance in a wider range of bitrates, but without increasing the model complexity significantly. The NWC module's capability is limited to perform the whole speech reconstruction process due to its compact topology as well as the discrepancy between the optimization objectives and the hard-to-quantify perceptual speech quality. We resolve this issue by introducing a scalable model architecture that can concatenate multiple speech coding modules, so a module can improve the mistake its predecessors made. First, in Sec. III-A, we propose to harmonize LPC as our first coding module. Followed by an NWC module, and by making LPC's quantization module trainable, we achieve a win-win strategy that fuses the traditional DSP technique and the modern deep learning model. Starting from this idea, which works well in the low-bitrate cases, we also extend it to cascading more autoencoders (Sec. III-B). The proposed CMRL system relays residual signals among the series of NWCs to scale up the coding performance at high bitrates.

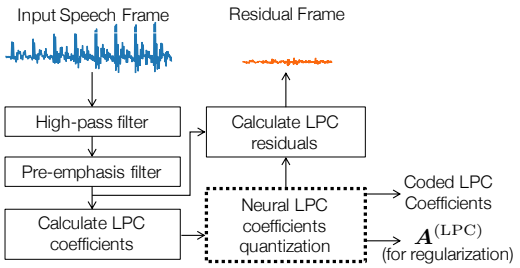
A. Trainable LPC Analyzer

LPC has been widely used to facilitate speech compression and sysnthesis, where *source-filter* model "explains out" the envelope of a speech spectrum, leaving a low-entropy residual signal [32]. Similarly, LPC serves as a pre-processor in our system before its residual signal being compressed by NWC as we will see in Sec. III-B. In this subsection, we redesign the LPC coefficient quantization process as a trainable module. We introduce collaborative quantization (CQ) to jointly optimize the LPC analyzer and NWCs as a residual coder.

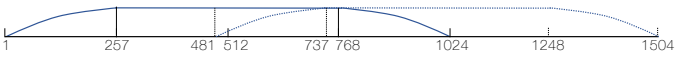
1) Speech resonance modeling: In the speech production process, the source as wide-band excitation signals go through the vocal tract tube. The shape-dependent resonances of the vocal tract filter the excitations before it being transformed to speech signals [33]. In speech coding, the "vocal tract response" is often modeled as an all-pole filter [34]. Having that, the t-th sample x_t can be approximated by a autoregressive model using M previous samples,

$$x_t = \sum_{k=1}^{M} l_k x_{t-k} + e_t, \tag{10}$$

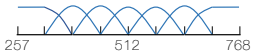
where the estimation error e_t represents the LPC residual, and l_k denotes the filter coefficients. Typically, l_k can be



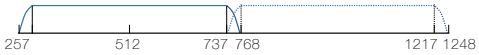
(a) The trainable LPC analyzer where the linear spectral pairs are quantized by our trainable soft-to-hard quantizer (in the dotted box).



(b) Cross-frame windowing



(c) Sub-frame windowing



(d) Synthesis windowing

Fig. 2: The signal flow chart for LPC analyzer (a) and windowing schemes in LPC (b)-(d).

efficiently estimated via Levinson-Durbin algorithm [35], and are to be quantized before LPC residual is calculated, i.e., e_t encompasses the quantization error. The LPC residual e_t serves as input to the NWC module, which works as explained in Sec. II, but on $\boldsymbol{e}$ rather than $\boldsymbol{x}$. Hence, how LPC coefficients are quantized determines NWC's input, the LPC residual.

2) Collaborative quantization: The conventional LPC coefficient quantization process is standardized in ITU-T G.722.2 (AMR-WB) [36]: 2.4k bits are assigned to represent the LPC coefficient per second though multistage vector quantization (MSVQ) [37] in a classic LPC analyzer. Once again, we employ the soft-to-hard quantizer as illustrated in Sec. II to make the quantization and bit allocation steps in the LPC analyzer trainable and communicatable with the neural codec.

We compute the LPC coefficients as in [38], first by applying high-pass filtering followed by pre-emphasizing (Fig. 2 (a)). When calculating LPC coefficients, the window in Fig. 2 (b) is used. The window is symmetric with the left and right 25% parts being tapered by a 512-point Hann window. After representing the 16 LPC coefficients in linear spectral pairs (LSP) [39], we quantize it using the soft-to-hard quantization scheme. Then, the sub-frame window in Fig. 2 (c) is applied to calculate LPC residual, which assures a more accurate residual calculation. The frame that covers samples [256:768], for instance, is decomposed into 7 sub-frames to calculate LPC residuals separately. Each 128-point Hann window in Fig. 2 (c) is with 50% overlap, except for the first and last window. They altogether form a constant overlap-add operation. Finally, after the synthesis using the reconstructed residual signal and corresponding LPC coefficients, the window in Fig. 2 (d) ta-

pers both ends of the synthesized signal, covering 512 samples with 32 overlapping samples between adjacent windows.

As an intuitive example, given the samples [1:1024] as the input, after the LPC analysis, neural residual coding, and LPC synthesis, samples [257:768] are decoded; the next input frame is [481:1504] (the dotted window in Fig. 2 (b)), whose decoded samples are within [737:1248]. The overlap-add operation is applied to the final decoded samples [737:768] (Fig. 2 (d)).

During this process, the calculated LPC coefficients are quantized using Algorithm 1, where the code vector is with 16 dimensions, i.e., $\boldsymbol{h} \in \mathbb{R}^{16}$. The number of kernels is set to be $K = 2^8 = 256$. Note that the soft assignment matrix for the LPC quantization, $\boldsymbol{A}^{(\text{LPC})}$, is also involved in the loss function to regularize the bitrate.

We investigate the impact of the trainable LPC quantization in collaboration with the rest of the NWC modules in Sec. IV.

B. Cross-Module Residual Learning (CMRL)

To achieve scalable coding performance towards transparency at high bitrates, we propose cross-module residual learning (CMRL) to conduct bit allocation among multiple neural codecs in a cascaded manner. CMRL can be regarded as a natural extension of what is described in Sec. III-A, where the LPC as a codec conducts the first round of coding by only modeling the spectral envelope. It leaves the residual signal for a subsequent NWC to be further compressed. With CMRL, we employ the concept of residual coding to cascade more NWCs. We also present a dual-phase training scheme to effectively train the CMRL model.

CMRL's scalability comes from its residual coding concept that enables a concatenation of multiple autoencoding modules. We define the residual signal recursively: i-th codec takes the residual of its predecessor as input, and the i-th reconstruction creates another residual for the next round, and so on. Hence, we have

$$\hat{\boldsymbol{x}}^{(i)} \leftarrow \mathcal{F}^{(i)}(\boldsymbol{x}^{(i)}),\ \boldsymbol{x}^{(i)} \leftarrow \boldsymbol{x}^{(i-1)} - \hat{\boldsymbol{x}}^{(i-1)},\ \boldsymbol{x}^{(1)} \leftarrow \boldsymbol{x}, \quad (11)$$

where $\hat{\boldsymbol{x}}^{(i)}$ stands for the reconstruction of the i-th input using the i-th coding module $\mathcal{F}^{(i)}(\cdot)$, while the input to the first codec is defined by the raw input frame $\boldsymbol{x}$. If we expand the recursion, we arrive at the non-recursive definition of $\boldsymbol{x}^{(i)}$,

$$\boldsymbol{x}^{(i)} = \boldsymbol{x} - \sum_{j=1}^{i-1} \hat{\boldsymbol{x}}^{(j)}, \quad (12)$$

which means the input to i-th model is the residual of the sum of all preceding $i-1$ codecs' decoded signals. It ensures the additivity of the entire system: adding more modules keeps improving the reconstruction quality. Hence, CMRL can scale up to high bitrates at the cost of increased model complexity.

CMRL is optimized in two phases. During Phase-I training, we sequentially train each codec from the first to the last one using a module-specific residual reconstruction goal,

$$\mathcal{E}(\boldsymbol{x}^{(i)} || \hat{\boldsymbol{x}}^{(i)}). \quad (13)$$

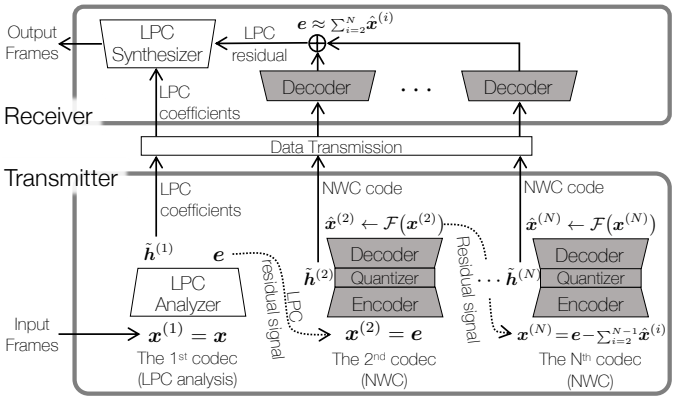


Fig. 3: The flow diagram of the test-time inference.

The purpose for Phase-I training is to get parameters for each codec properly initialized. Then, Phase-II finetunes all trainable parameters of the concatenated modules to minimize the global reconstruction loss,

$$\mathcal{E}\left(\boldsymbol{x} \middle|\middle| \sum_{i=1}^{N} \hat{\boldsymbol{x}}^{(i)}\right). \quad (14)$$

Phase-II also re-adjusts the quantization components of all modules, seeking the optimal bit allocation for all modules.

C. Signal Flow during Inference

Fig. 3 shows the full CMRL signal flow with N sub-codecs, having an LPC module as the first one. On the transmitter side the LPC analyzer first processes the input frame $\boldsymbol{x}$ of 512 samples and computes 16 coefficients, $\boldsymbol{h}^{(1)}$, as well as the residual samples $\boldsymbol{x}^{(2)}$. Then, the residual signal goes through the $N-1$ NWCs in sequentially. Note that the transmission process's primary job is to produce a quantized bitstring $\tilde{\boldsymbol{h}}^{(i)}$ from LPC and each NWC. To this end, NWC's decoder part must also run to compute the residual signal and relay it to the next NWC module. The bitstring is generated as a concatenation of all encoder outputs: $\tilde{\boldsymbol{h}} = \left[\tilde{\boldsymbol{h}}^{(1)}; \tilde{\boldsymbol{h}}^{(2)}; \cdots; \tilde{\boldsymbol{h}}^{(N)}\right]$. Once the bitstring is available on the receiver side, all NWC decoders run to reconstruct the LPC residual signal, i.e., $\hat{\boldsymbol{x}}^{(2)} \approx \sum_{i=2}^{N} \mathcal{F}_{\text{dec}}^{(i)}(\tilde{\boldsymbol{h}}^{(i)})$. Then it is used as input of the LPC synthesizer, along with the LPC coefficients.

IV. Evaluation

In this section, we examine the proposed neural speech coding model presented in Sec.II and III. The evaluation critera include both objective measures such as PESQ [40] and signal-to-noise ratio (SNR) and subjective scores from MUSHRA listening tests [41]. In addition, we conduct ablation analysis to provide a detailed comparison between various loss terms and bit allocation schemes. Finally, we report the system delay and execution time under four hardware specifications.

A. Data Processing

The training dataset is created from 300 speakers randomly selected from the TIMIT corpus [42] with no gender pref-

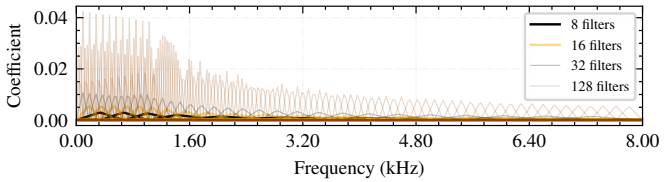


Fig. 4: The coarse-to-fine filter bank analysis in the mel scale.

erence. Each speaker contributes 10 utterances totaling 2.6 hour-long training set, which is a reasonable size due to our compact design. The same scheme is adopted when creating the validation dataset and test dataset with 50 speakers, respectively. All three datasets are mutually exclusive. All neural codecs in this work are trained and tested via the same set of data for a fair comparison. We normalize each utterance to have a unit variance, then divided by the global maximum amplitude, before being framed into segments with the size of 512 samples. On the receiver side, we conduct overlap-and-add after the synthesis of the frames, where a 32-sample Hann window is applied to the overlapping region of the same size.

With the LPC codec, we apply high-pass filtering defined in the z-space, $\mathcal{G}_{\text{hp}}(z)=\frac{0.989502-1.979004z^{-1}0.989502z^{-2}}{1-1.978882^{-1}0.979126^{-2}}$, to the normalized waveform. A pre-emphasis filter, $\mathcal{G}_{\text{preemp}}(z)=1-0.68z^{-1}$, follows to boost the high frequencies.

B. Training Targets and Hyperparameters

The loss function is defined as

$$\mathcal{L} = \lambda_{\text{MSE}} \sum_{t=1}^{T} (x_t - \hat{x}_t)^2 + \lambda_{\text{mel}} \sum_{b=1}^{4} \sum_{f=1}^{F_b} \left(y_f^{(b)} - \hat{y}_f^{(b)} \right)^2 + \lambda_{\text{Q}} \mathcal{L}_{\text{Q}} + \lambda_{\text{ent}} \mathcal{H}(\boldsymbol{\beta}) \quad (15)$$

where the first term measures the mean squared error (MSE) between the raw waveform samples and their reconstruction. Ideally, if the model complexity and the bitrate is sufficiently large, an accurate reconstruction is feasible by using MSE as the only loss function. Otherwise, the result is usually sub-optimal due to the lack of bits: coupled with the MSE loss, the decoded signals tend to contain broadband artifact. The second term supplements the MSE loss and helps suppress this kind of artifact. To this end, we follow the common steps to conduct mel-scaled filter bank analysis, which results in a mel spectrum $\boldsymbol{y}$ that has a higher resolution in the low frequencies than in the high frequencies. The filter bank size defines the granularity level of the comparison. Following [31], we conduct a coarse-to-fine filter bank analysis by setting four filter bank sizes, $F_1 = 8, F_2 = 16, F_3 = 32, F_4 = 128$ as shown in Fig. 4, which result in four kinds of resolutions for mel spectra $\boldsymbol{y}^{(b)}$ indexed by $b \in \{1, 2, 3, 4\}$.

All models are trained on Adam optimizer with default learning rate adaptation rates [43]. The batch size is fixed with 128 frames . The initial learning rate is 2×10^{-3} for the first neural codec. With CMRL, the learning rate for the successive neural codecs is 2×10^{-4}. Finetuning of all those models is with a smaller learning rate 2×10^{-5}. All models are sufficiently trained until the validation loss converges after being exposed to about 5×10^5 batches. These hyperparameters were chosen based on validation.

The blending weights in the loss function in Eq. (15) are also selected based on the validation performance. Empirically, the ratio between the time-domain loss and mel-scaled frequency loss affects the trade-off between the SNR and perceptual quality of decoded signals. If the time-domain loss dominates the optimization process, the model compresses each sub-band with an equal effort. In that case, the artifact will be audible unless the SNR reaches a rather high level (over 30 dB) which entails a high bitrate and model complexity. On the other hand, if only the mel-scaled frequency loss is in place, the reconstruction quality in the high frequency will degrade. The impact of these blending weights for these two loss terms is detailed in Sec. IV-F via an ablation analysis.

The weights for the quantization regularizer λ_{Q} and entropy regularizer λ_{ent} are set to be 0.5 and 0.0, respectively. As for λ_{ent}, we alter it after every epoch by 0.015: if the current model's bitrate is higher than the target bitrate, λ_{ent} increases to penalize the model's entropy more, and vice versa. Note that we omit the module index i in Eq. (15), so the meaning of $\hat{x}_t$ depends on the context: either the module-specific reconstruction as in Eq. (13) or the sum of all recovered residual signals for Phase-II finetuning as in Eq. (14). Similarly, $\mathcal{L}_{\text{Q}}$ and $\mathcal{H}_{\boldsymbol{\beta}}$ can encompass all modules' quantization and entropy losses including LPC's for Phase-II. We delay the introduction of the quantization and entropy loss until the fifth epoch.

C. Bitrate Modes and Competing Models

We consider three bitrates, 12, 20, and 32 kbps, to validate models' performance in a range of use cases. We evaluate following different versions of neural speech codec:

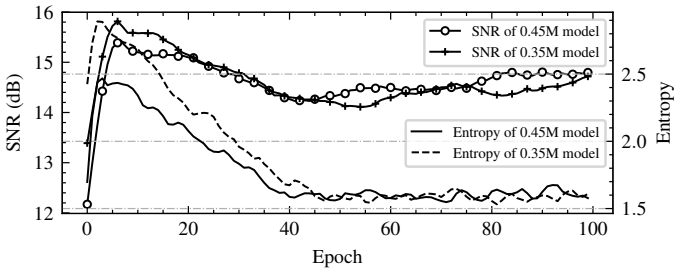
- Model-I: The NWC baseline (Sec. II).
- Model-II: Another baseline that combines the legacy LPC and an NWC module for residual coding.
- Model-III: A trainable LPC quantization module followed by an NWC and finetuning (Sec. III-A);
- Model-IV: Similar to Model-III but with two NWC modules: the full-capacity CMRL implementation (Sec.III-B). It is tested cover the high bitrate case, 32 kbps.

Regarding the standard codecs, AMR-WB [44] and Opus [45] are considered for comparison. AMR-WB, as an ITU standard speech codec, operates in nine different modes covering a bitrate range from 6.6 kbps to 23.85 kbps, providing excellent speech quality with a bitrate as low as 12.65 kbps in wideband mode. As a more recent codec, Opus shows the state-of-the-art performance in most bitrates up to 510 kbps for stereo audio coding, except for the very low bitrate range.

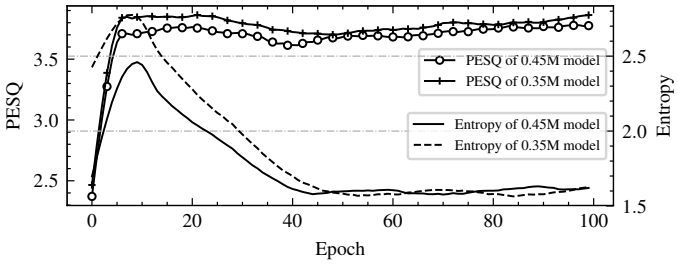
We first compare all models with respect to the objective measures, while being aware that they are not consistent with the subjective quality. Hence, we also evaluate these codecs in two rounds of MUSHRA subjective listening tests: the neural codecs are compared in the first round, whose winner is compared with other standard codecs in the second round.

TABLE III: Objective measurements for neural codec comparison under three bitrate cases.

Bitrate (kbps)	SNR (dB)						PESQ-WB					
	Model-I	Model-II	Model-III	Model-IV	AMR-WB	Opus	Model-I	Model-II	Model-III	Model-IV	AMR-WB	Opus
∼12	**12.37**	10.69	10.85	–	11.60	9.63	3.67	3.45	3.60	–	3.92	**3.93**
∼20	**16.87**	10.73	13.65	–	13.14	9.46	**4.37**	3.95	4.01	–	4.18	**4.37**
∼32	**20.24**	11.84	14.46	17.11	–	17.66	**4.42**	4.15	4.18	4.35	–	4.38

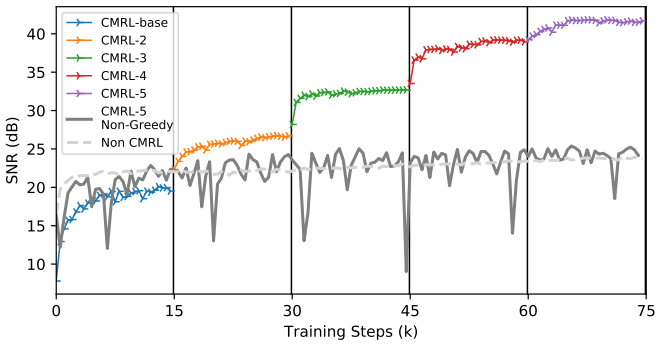


(a) The validation SNR curve during training

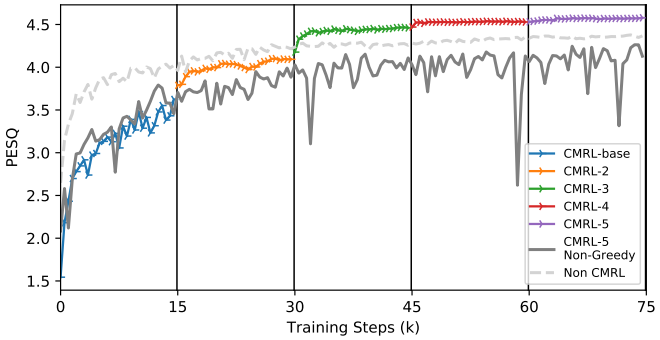


(b) The validation PESQ curve during training

Fig. 5: Speech reconstruction performance stays almost the same when the model size decreases from 0.45 to 0.35 million parameters with the help from the structural modification.



(a) Scalability with respect to SNR



(b) Scalability with respect to PESQ

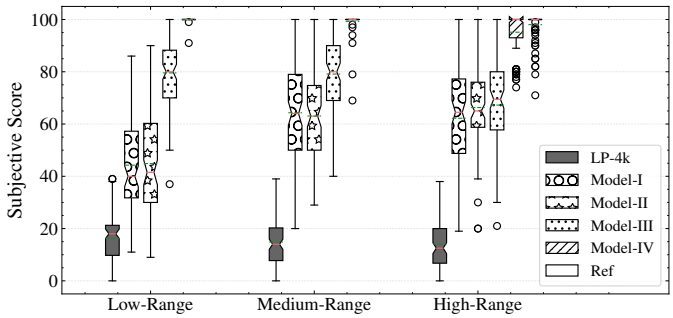
Fig. 6: In CMRL, performance leaps when the new neural codec is added for residual cascading.

D. Objective Measurements

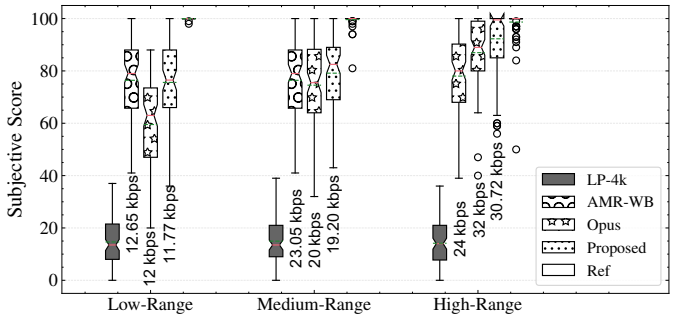
1) The compact NWC module and its performance: Compared to our previous models in [20][46][47] that use 0.45 million parameters, the newly proposed NWC in this work only has 0.35 million parameters. It is also a significant reduction from the other compact neural speech codec [31] with 1.6 million parameters. As introduced in Sec. II the model size reduction is achieved via the GLU [26] and depthwise separable convolution for upsampling [27]. In our first experiment, we show that the objective measures stay the same. Fig. 5 compares the NWC modules before and after the structural modification proposed in Sec. II in terms of (a) signal-to-noise ratio (SNR) and (b) PESQ-WB [40]. We can see that the newly proposed model with 0.35M parameters is comparable to the larger model. Therefore, it justifies its use as the basic module in a range of models from Model-I to IV.

2) The impact of CMRL's residual coding: To validate the merit of CMRL's residual coding concept, we scale up the CMRL model by incrementally adding more NWC modules up to five. In Fig. 6, both SNR and PESQ values keep increasing when CMRL keeps adding a new NWC module. There are two noticeable points in these graphs. First, the greedy module-wise pretraining is important for the performance: whenever a new model is added, it is pretrained to minimize the module specific loss Eq. (13) first (Phase-I), then the global loss Eq. (14), subsequently (Phase-II). A model that does not perform Phase-II (thick gray line) stagnates no matter how many NWCs are added. Second, we also train a very large NWC model with the same amount of parameters as CMRL with five NWCs combined (grey dash). It turns out the equally large model fails to scale up due to its single integrated architecture. While we eventually decide to use only up to two NWCs for speech coding for our highest bitrate case, 32 kbps, CMRL's scalability is clearly beneficial if one has to extend to higher bitrates for non-speech audio coding.

3) Overall objective comparison of all competing models: TABLE III reports SNR and PESQ-WB from all competing systems. AMR-WB in the low-range bitrate setting operates at 12.65 kbps and 23.05 kbps for the mid-range. It is noticeable that, among neural codecs, the simplest Model-I outperforms others in all three bitrate setups both in terms of SNR and



(a) Neural waveform codecs comparison.



(b) Comparison between proposed methods and standard codecs.

Fig. 7: MUSHRA subjective listening test results.

PESQ-WB. Even compared with AMR-WB and Opus, Model-I is the winner except for the low bitrate case where Opus achieves the highest PESQ score. It is because the single autoencoder model is highly optimized for the objective loss during training, although it does not necessarily mean that the higher objective score leads to a better subjective quality as presented in Sec. IV-E. It is also observed that with CQ, Model-III gains slightly higher SNR and PESQ scores compared to Model-II, which uses the legacy LPC. Finally, the performance scales up significantly when Model-IV starts to employ two NWCs on top of LPC, which is our proposed full neural speech coding setup. Aside from objective measure comparison, to further evaluate the quality of proposed codec, we discuss the subjective test in the next section.

E. Subjective Test

We conduct two rounds of MUSHRA tests: (a) to select the best one out of the proposed models (from Model-I to IV) (b) to compare it with the standard codecs, i.e., AMR-WB and Opus. Each round covers three different bitrate ranges, totaling six MUSHRA sessions. A session consists of ten trials, for which ten gender-balanced test signals are randomly selected. Each trial has one low-pass filtered signal serving as the anchor (with a cutoff frequency at 4kHz), the hidden reference, as well as signals decoded from competing systems. We recruit ten participants who are audio experts with prior experiences in speech/audio quality evaluation. The subjective scores are rendered in Fig. 7 as boxplots. Each box ranges from the 25 to 75 percentile with a 95% confidence interval. The mean and median are presented as the green dotted line and pink hard line, respectively. Outliers are represented in circles.

TABLE IV: Ablation analysis on blending weights.

(a) Neural codec only

Blending Ratio (MSE : mel)	Decoded SNR (dB)	Decoded PESQ
1 : 0	**18.12**	3.67
0 : 1	0.16	4.23
1 : 1	6.23	4.31
10 : 1	16.88	**4.37**

(b) Collaboratively trained LPC codec and neural codec

Blending Ratio (MES : mel)	Residual SNR (dB)	Decoded SNR (dB)	Decoded PESQ
1 : 0	**9.73**	14.25	3.84
0 : 1	1.79	17.23	4.02
1 : 1	7.11	**17.82**	**4.08**
10 : 1	8.26	17.55	4.01

1) Comparison among the proposed neural codecs: In Fig. 7 (a) we see that Model-III's produces decoding results that are much more preferred than both Model-I and Model-II, which are a pure end-to-end model and with the non-trainable legacy LPC module, respectively. The advantage is more significant in lower bitrates. It is contradictory to the objective scores reported in TABLE III where Model-I often achieved the highest scores. It is because Model-III's joint training of the LPC quantization module and NWCs. Compared to the deterministic quantization module in the legacy LPC, CQ can assign a different amount of bits to different frames in collaboration with the following NWC module, maximizing the coding efficiency. We also note that Model-III's performance stagnates in the high bitrate experiments, suggesting its poor scalability. To this end, for the high bitrate experiment, we additionally test Model-IV with two NWC residual coding modules instead of just one. Model-IV outperformed Model-II by a large margin, showcasing a transparent quality.

2) Comparison with standardized codecs: Fig. 7 (b) shows that, our Model-II is on par with AMR-WB for the low-range bitrate case, while outperforming Opus which tends to lose high frequency components. In the medium-range, Model-II at 19.2 kbps is comparable to Opus at 20.0 kbps and AMR-WB at 23.05 kbps. In the high bitrate range, our Model-IV outperforms Opus that operates 32 and 24 kbps, while AMR-WB is omitted as it does not support those high bitrates.

F. Ablation Analysis

In this section, we perform some ablation analyses to justify our choices that led to CQ and CMRL's superior subjective test results. We investigate how different blending ratios between loss terms can alter the performance. We will also explore the optimal bit allocation strategy among coding modules.

1) Blending weights for the loss terms: Out of the two major reconstruction loss terms, MSE serves as the main loss for the end-to-end NWC system, while the mel-scaled loss prioritizes certain frequency bands over the others. In TABLE IV (a), the SNR reaches the highest when there is only the MSE term, which however, leads to the lowest PESQ score.

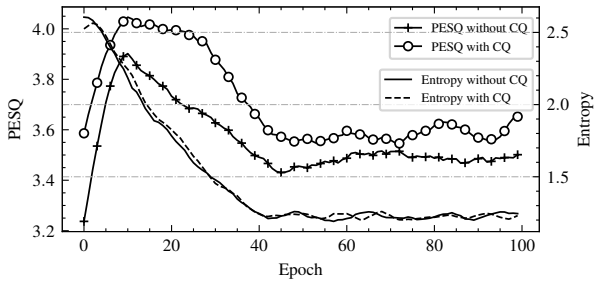


Fig. 8: The ablation analysis on CQ.

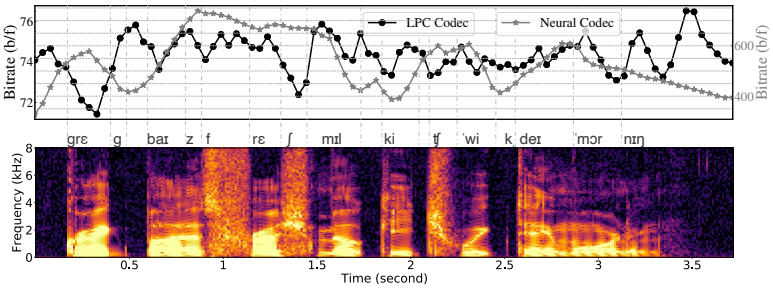


Fig. 9: The frame-wise bit allocation analysis.

By only keeping the mel-scaled loss term, the PESQ score is decent (4.23) while the sample-by-sample reconstruction is poor as suggested by the SNR value (0.16 dB). Similarly, in TABLE IV (b) where the input of the neural codec is the LPC residuals, MSE alone yields the highest SNR for the reconstruction of the LPC residual, which however, does not benefit the final synthesized signal even in terms of SNR. We choose the blending ratio of $10:1$, which consistently shows a good performance in all proposed models.

2) CQ's impact on the speech quality: We compare the PESQ values of the decoded signals from Model II and III. Since Model-III shares the same architecture with Model-II except for the CQ training strategy, the comparison is to verify that CQ can effectively allocate bits to the LPC and NWC modules. Fig. 8 shows that the total entropy of the two models are under the control regardless of the use of CQ mechanism. However, we can see that Model-III with CQ achieves higher PESQ during and after the control of the entropy, showcasing that the CQ approach benefits the codec's performance.

3) Bit allocation between the LPC and NWC modules: Since the proposed CQ method is capable of assigning different bits to the LPC and NWC modules dynamically, i.e., in a frame-by-frame manner, we analyze its impact in more detail. In the mid-range bitrate setting, Fig. 9 shows the amount of bits assigned to both modules per frame (b/f). First of all, we observe that the dynamic bit allocation scheme indeed adjusts the LPC and NWC bitrates over time. It is also noticeable that the LPC module consumes more than average bits for near-silent frames, while the corresponding NWC bitrate is reduced. Given that the LPC module consumes a significant smaller order of magnitudes, i.e., less than 80 b/f in general, it seems to be an efficient bit allocation behavior that the NWC module saves ~ 100 b/f at the cost of a small increase of 2 b/f in the LPC module. However, it still requires a significant amount of bits to even represent those near-silent frames, which can be further optimized. Finally, it appears that NWC is less efficient for fricatives (e.g., f and $\int$) and affricates (e.g., $t\int$). TABLE V shows the overall bit allocation among different modules. In the low bitrate case, it is worth noting that CQ uses 58 b/f or 1.93 kbps, differently from AMR-WB's standard, 2.4 kbps.

4) Bit allocation between the two NWC modules in Model-IV: To find the optimal bit allocation between two NWC modules, we first conduct an ablation analysis on 3 different bit allocation choices. In Fig. 10, both the SNR and PESQ

TABLE V: Bit allocation among coding components.

Bitrate Modes (kbps)	LPC Coefficients (bits / frame)	LPC Residual (bits / frame)	Total (bits / frame)
~ 11.77	58	295	353
~ 19.20	74	502	576
~ 30.72	74	486+384	944

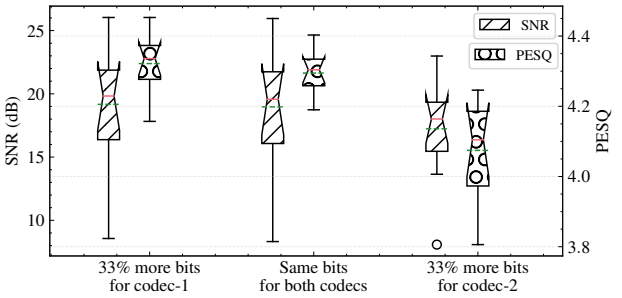


Fig. 10: Ablation analysis on bit allocation schemes between codec-1 and codec-2 in Model-IV at 32 kbps.

scores degrade when the second NWC uses 33.3% more bits than the first one. Among these 3 choices, the highest PESQ score is obtained when the first NWC module uses 33.3% more bits. In practice, the bit allocation is automatically determined during the optimization process. In TABLE V, for example, the bit ratio between two NWC modules of Model-IV in the high bitrate case is about $486:384 \approx 55.9:44.1$, in accord with the observation from the ablation analysis that the first module should use more bits.

G. Complexity and Delay

The proposed NWC model is with 0.35 million parameters, a half of which is for the decoder. Hence, in 32 kbps with two NWC modules for residual coding, the model size totals 0.7M parameters, with the decoder size of 0.35M. Even though our decoder is still not as compact as those in traditional codecs, it is $100\times$ smaller than a WaveNet decoder.

Aside from the model size, we investigate the codec's delay and the processing time. The codec will have algorithmic delay if it relies on future samples to predict the current sample. The processing time during the encoding and decoding processes also adds up to the runtime overhead.

TABLE VI: Execution time ratios during model inference (%).

Hardware	0.45M	0.35M	0.45M×2	0.35M×2
1× Tesla V100	12.49	13.38	20.69	21.12
1× Tesla K80	24.45	22.53	39.42	38.82
8× CPU cores	20.76	18.91	35.17	33.80
1× CPU core	46.88	42.44	87.38	80.21

1) Algorithmic delay: The delay of our system is defined by the frame size: the first sample of a frame can be processed only after the entire frame is buffered: $512/16000 = 32$ms. Causal convolution can minimize such delay at the expense of the reduced speech quality, because it only uses past samples.

2) Analysis of the processing time: The execution time is another important factor to be considered for real-time communications. The bottom line is that the execution of the encoding and decoding processes is expected to be within the duration of the hop length so as not to add extra delays. For example, WaveNet codec [13] minimizes the system delay using causal convolution, but its processing time, though not reported, can be rather high as it is an autoregressive model with over 20 million parameters. TABLE VI lists the execution time ratio of our models. The ratio (in percentage) is defined as the execution time to encode and decode the test signals divided by the duration of those signals. Meanwhile, Kankanahalli's model requires 4.78ms to encode and decode a hop length of 30ms on an NVIDIA® GeForce® GTX 1080 Ti GPU, and 21.42ms on an Intel® Core™ i7-4970K CPU (3.8GHz), which amount to 15.93% and 71.40% of the execution time ratio, respectively [31]. Our small-sized models (0.45M and 0.35M) on both CPU (Intel® Xeon® Processor E5-2670 V3 2.3GHz) and GPU run faster than Kankanahali's, while direct comparison is not fair due to the different computing environment. The CMRL models with two NWC modules require more execution time. Note that all our models compared in this test achieved the real-time processing goal as their ratios are under 100%. However, the ratio comparison results between the 0.45M model and 0.35M model are not consistent. We believe this is because that the internal implementation of different residual learning blocks in TensorFlow may lead to various runtime optimization effects.

V. Concluding Remarks

Recent neural waveform codecs have outperformed the conventional codecs in terms of coding efficiency and speech quality, at the expense of model complexity. We proposed a scalable and lightweight neural acoustic processing unit for waveform coding. Our smallest model contains only 0.35 million parameters whose decoder is more than 100X smaller than the WaveNet based codec. By incorporating a trainable LPC analyzer with collaborative quantization and residual cascading, our model clearly demonstrates superior or comparable performance to the standardized codecs. Our model operates frame-wise, leading to a delay as small as 16 msec; even on a single-core CPU without compiler-level matrix multiplication optimization, it achieved real-time processing.

Admittedly, conventional codecs have well performed from narrow to full band scenarios already. But it is not easy to integrate these standalone DSP components to a neural network-powered conversational AI engine, as the acoustic signal representation is predetermined by the bit allocation logic, not learned in the latent space via a global training process. To this end, our trainable codec provides an affordable way for unsupervised speech waveform representation learning that also provides competent level of compression.

References

[1] M. R. Schroeder, "Vocoders: Analysis and synthesis of speech," *Proceedings of the IEEE*, vol. 54, no. 5, pp. 720–734, 1966.

[2] A. V. McCree and T. P. Barnwell, "A mixed excitation LPC vocoder model for low bit rate speech coding," *IEEE Transactions on Speech and audio Processing*, vol. 3, no. 4, pp. 242–250, 1995.

[3] J. Stachurski and A. McCree, "Combining parametric and waveform-matching coders for low bit-rate speech coding," in *2000 10th European Signal Processing Conference*. IEEE, 2000, pp. 1–4.

[4] M. Schroeder and B. Atal, "Code-excited linear prediction (CELP): High-quality speech at very low bit rates," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'85.*, vol. 10. IEEE, 1985, pp. 937–940.

[5] J. Stachurski and A. McCree, "A 4 kb/s hybrid MELP/CELP coder with alignment phase encoding and zero-phase equalization," in *ICASSP'20, IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 2000, pp. 1379–1382.

[6] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "A regression approach to speech enhancement based on deep neural networks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 1, pp. 7–19, 2014.

[7] Y. Luo and N. Mesgarani, "TasNet: Surpassing ideal time-frequency masking for speech separation," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 27, no. 8, pp. 1256–1266, 2019.

[8] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[9] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. Ieee, 2013, pp. 6645–6649.

[10] O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu, "Convolutional neural networks for speech recognition," *IEEE/ACM Transactions on audio, speech, and language processing*, vol. 22, no. 10, pp. 1533–1545, 2014.

[11] J. Chorowski, R. J. Weiss, S. Bengio, and A. van den Oord, "Unsupervised speech representation learning using WaveNet autoencoders," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 27, no. 12, pp. 2041–2053, 2019.

[12] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio." *Speech Synthesis Workshop*, vol. 125, 2016.

[13] W. B. Kleijn, F. S. C. Lim, A. Luebs, J. Skoglund, F. Stimberg, Q. Wang, and T. C. Walters, "WaveNet based low rate speech coding," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2018, pp. 676–680.

[14] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 6306–6315.

[15] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[16] Y. L. C. Garbacea, A. van den Oord, "Low bit-rate speech coding with VQ-VAE and a WaveNet decoder," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2019.

[17] J.-M. Valin and J. Skoglund, "LPCNet: Improving neural speech synthesis through linear prediction," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5891–5895.

[18] N. Kalchbrenner, E. Elsen, K. Simonyan, S. Noury, N. Casagrande, E. Lockhart, F. Stimberg, A. Oord, S. Dieleman, and K. Kavukcuoglu, "Efficient neural audio synthesis," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2410–2419.

[19] J.-M. Valin and J. Skoglund, "A real-time wideband neural vocoder at 1.6 kb/s using LPCNet," in *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, 2019.

[20] K. Zhen, J. Sung, M. S. Lee, S. Beack, and M. Kim, "Cascaded cross-module residual learning towards lightweight end-to-end speech coding," in *Proceedings of the Annual Conference of the International Speech Communication Association (Interspeech)*, 2019.

[21] C. Un and D. Magill, "The residual-excited linear prediction vocoder with transmission rate below 9.6 kbits/s," *IEEE transactions on communications*, vol. 23, no. 12, pp. 1466–1474, 1975.

[22] K. Zhen, M. S. Lee, J. Sung, S. Beack, and M. Kim, "Psychoacoustic calibration of loss functions for efficient end-to-end neural audio coding," *IEEE Signal Processing Letters*, vol. 27, pp. 2159–2163, 2020.

[23] K. Tan, J. T. Chen, and D. L. Wang, "Gated residual networks with dilated convolutions for supervised speech separation," in *Proc. ICASSP*, 2018.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[25] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[26] K. Tan, J. T. Chen, and D. L. Wang, "Gated residual networks with dilated convolutions for monaural speech enhancement," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 1, pp. 189–198, 2019.

[27] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[28] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1874–1883.

[29] M. Garey, D. Johnson, and H. Witsenhausen, "The complexity of the generalized Lloyd-max problem (corresp.)," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 255–256, 1982.

[30] E. Agustsson, F. Mentzer, M. Tschannen, L. Cavigelli, R. Timofte, L. Benini, and L. V. Gool, "Soft-to-hard vector quantization for end-to-end learning compressible representations," in *Advances in Neural Information Processing Systems (NIPS)*, 2017, pp. 1141–1151.

[31] S. Kankanahalli, "End-to-end optimized speech coding with deep neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018.

[32] F. Itakura, "Early developments of LPC speech coding techniques," in *icslp*, 1990, pp. 1409–1410.

[33] I. R. Titze and D. W. Martin, "Principles of voice production," 1998.

[34] D. O'Shaughnessy, "Linear predictive coding," *IEEE potentials*, vol. 7, no. 1, pp. 29–32, 1988.

[35] J. Franke, "A levinson-durbin recursion for autoregressive-moving average processes," *Biometrika*, vol. 72, no. 3, pp. 573–581, 1985.

[36] ITU-T G.722.2:, "Wideband coding of speech at around 16 kbit/s using adaptive multi-rate wideband (AMR-WB)," 2003.

[37] A. Gersho and V. Cuperman, "Vector quantization: A pattern-matching technique for speech coding," *IEEE Communications magazine*, vol. 21, no. 9, pp. 15–21, 1983.

[38] B. Bessette, R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola, and K. Jarvinen, "The adaptive multirate wideband speech codec (AMR-WB)," *IEEE Transactions on Speech and Audio Processing*, vol. 10, no. 8, pp. 620–636, 2002.

[39] F. Soong and B. Juang, "Line spectrum pair (LSP) and speech data compression," in *ICASSP'84. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 9. IEEE, 1984, pp. 37–40.

[40] A. Rix, J. Beerends, M. Hollier, and A. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in *Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on*, vol. 2. IEEE, 2001, pp. 749–752.

[41] ITU-R Recommendation BS 1534-1, "Method for the subjective assessment of intermediate quality levels of coding systems (MUSHRA)," 2003.

[42] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, N. L. Dahlgren, and V. Zue, "TIMIT acoustic-phonetic continuous speech corpus," *Linguistic Data Consortium, Philadelphia*, 1993.

[43] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2015.

[44] B. Bessette, R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola, and K. Jarvinen, "The adaptive multirate wideband speech codec (AMR-WB)," *IEEE transactions on speech and audio processing*, vol. 10, no. 8, pp. 620–636, 2002.

[45] J.-M. Valin, G. Maxwell, T. B. Terriberry, and K. Vos, "High-quality, low-delay music coding in the Opus codec," *arXiv preprint arXiv:1602.04845*, 2016.

[46] K. Zhen, M. S. Lee, J. Sung, S. Beack, and M. Kim, "Efficient and scalable neural residual waveform coding with collaborative quantization," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 361–365.

[47] K. Zhen, M. S. Lee, J. Sung, S. Beack, and M. Kim, "Psychoacoustic calibration of loss functions for efficient end-to-end neural audio coding," *IEEE Signal Processing Letters*, pp. 1–1, 2020.