Predictive Neural Speech Coding

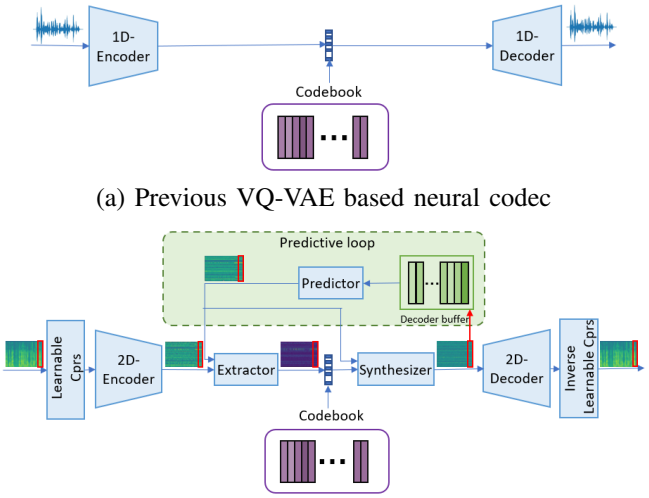
Xue Jiang, Xiulian Peng, Huaying Xue, Yuan Zhang, Yan Lu

***Abstract*—Neural audio/speech coding has shown its capability to deliver a high quality at much lower bitrates than traditional methods recently. However, existing neural audio/speech codecs employ either acoustic features or learned blind features with a convolutional neural network for encoding, by which there are still temporal redundancies inside encoded features. This paper introduces latent-domain predictive coding into the VQ-VAE framework to fully remove such redundancies and proposes the TF-Codec for low-latency neural speech coding in an end-to-end way. Specifically, the extracted features are encoded conditioned on a prediction from past quantized latent frames so that temporal correlations are further removed. What's more, we introduce a learnable compression on the time-frequency input to adaptively adjust the attention paid on main frequencies and details at different bitrates. A differentiable vector quantization scheme based on distance-to-soft mapping and Gumbel-Softmax is proposed to better model the latent distributions with rate constraint. Subjective results on multilingual speech datasets show that with a latency of 40ms, the proposed TF-Codec at 1kbps can achieve a much better quality than Opus 9kbps and TF-Codec at 3kbps outperforms both EVS 9.6kbps and Opus 12kbps. Numerous studies are conducted to show the effectiveness of these techniques.**

***Index Terms*—Neural audio/speech coding, auto-encoder, predictive coding.**

I. Introduction

Neural audio/speech coding has emerged and made fast progress recently to deliver a high quality at very low bitrates, especially for speech. Existing neural codecs could mainly be categorized into two types, one based on generative decoder models [1]–[5] and one based on end-to-end neural audio/speech coding [6]–[11]. The former extracts acoustic features from the audio for encoding and uses a strong decoder to recover the waveform based on generative models. The latter mainly leverages the VQ-VAE [12] framework to learn an encoder, a vector quantizer and a decoder in an end-to-end way. The latent features to quantize are mostly blindly learned using a convolutional network (CNN) without any prior knowledge. These methods have largely increased the coding efficiency by achieving a high quality at a low bitrate. However, temporal correlations are not fully exploited in these algorithms. There are still much redundancy among neighboring frames in encoded features. In light of this, we propose to incorporate predictive coding into the VQ-VAE-based neural coding framework to remove such redundancies.



(a) Previous VQ-VAE based neural codec

(b) Our predictive VQ-VAE based neural codec

Fig. 1. Proposed latent-domain predictive neural speech coding.

Predictive coding is widely used in traditional image [13], video [14]–[16] and audio coding [17], [18] for spatial and temporal redundancy removal, where reconstructed neighboring blocks/frames/samples are used to predict the current block/frame/sample and the predicted residuals are quantized and encoded to bitstream. The residuals after prediction are much sparser and the entropy is largely reduced. In neural video codec [19], [20], such temporal correlation is also exploited by utilizing motion-aligned reference frame as a prediction or context for encoding current frame. However, in neural audio codec, such techniques have not been investigated yet, to the best of our knowledge.

Although temporal correlations are exploited in encoder and decoder of neural audio/speech coding by convolutional or recurrent neural networks, these operations can be seen as a kind of open-loop prediction or nonlinear transform (See Fig. 1 (a)). After quantization, the temporal correlation is broken to some extent. For better recovery quality, the neural network tends to leave some redundancy in the learned latent representation. Nevertheless, by closed-loop prediction as in our predictive coding (see Fig. 1 (b)), such a redundancy is removed in encoded features but the recovery capability is not affected. The learned latent features are sparse and the decoding could recover with a good quality by using the same prediction as that in encoding.

This paper is the first to introduce predictive coding into the VQ-VAE framework for neural speech coding. To reduce the delay, this contextual coding is performed in latent domain as shown in Fig. 1 (b). Unlike traditional predictive video/audio

Xue Jiang is with the School of Information and Communication Engineering, Communication University of China, Beijing 100024, China (e-mail: jiangxhoho@cuc.edu.cn).

X. Peng, H. Xue and Y. Lu are with the Microsoft Research Asia, Beijing 100080, China (e-mail: xipe@microsoft.com; huxue@microsoft.com; yanlu@microsoft.com).

Y. Zhang is with the State Key Laboratory of Media Convergence and Communication, Communication University of China, Beijing 100024, China (e-mail: yzhang@cuc.edu.cn).

This work was done when Xue Jiang was an intern at Microsoft Research Asia.

coding by subtracting samples from predictions, we introduce a learnable extractor to fuse the prediction with encoder features to get the sparse "new" information for coding of each frame. All modules are end-to-end learned with adversarial training. Moreover, unlike most previous neural codecs that take time-domain input, we introduce the time-frequency input with a learnable compression on the amplitude so that the network can automatically balance the attention paid on main and detailed components at different bitrates (see Fig. 1 (b)), which largely boost the quality at a bitrate as low as 1kbps for low-latency speech coding.

The main contributions of this paper are summarized as follows:

- We propose TF-Codec, a low-latency neural speech codec that is the first to report high quality at 1kbps with a latency of 40ms as far as we know.
- We introduce predictive coding into the VQ-VAE based neural speech codec, which largely removes the temporal redundancy and therefore boosts the coding efficiency.
- We introduce a learnable compression on time-frequency input to adaptively adjust the attention paid on main and detailed components at different bitrates.
- We introduce a differentiable vector quantization mechanism based on distance-to-soft mapping and Gumbel-Softmax to facilitate rate control and achieve better rate-distortion optimization.

II. Related Work

A. Neural Audio/Speech coding

Generative model based audio coding With the advancement of generative models in providing high-quality speech synthesis, recently some researchers proposed to leverage them for speech coding as well [1]–[5], such as WaveNet [21] and LPCNet [22]. WaveNet is the first to be used as a learned generative decoder to produce high quality audio from a conventional encoder at 2.4kbps [1]. Some researchers [5] improve Opus speech quality at low bitrates by using LPCNet for speech synthesis. Lyra [2] is a generative model which synthesizes speech from quantized mel-spectrum with an auto-regressive WaveGRU model to produce high quality speech at 3kbps. These methods have achieved good quality at low bitrates but the potential of neural audio coding is not fully exploited.

End-to-end audio coding This category learns the encoding, vector quantization and decoding in an end-to-end way based on the VQ-VAE [12] framework [6]–[11]. In [6], a VQ-VAE encoder and a WaveNet-based decoder are jointly learned end to end, yielding a high reconstruction quality while passing speech through a compact latent representation corresponding to very low bitrates. The recently proposed SoundStream [8] achieves a superior audio quality at a wide range of bitrates from 3kbps up to 18kbps with end-to-end learning and a mix of adversarial and reconstruction losses. More recently, an end-to-end audio codec with a cross-module residual coding pipeline was proposed for scalable coding [10]. Unlike previous methods based on the waveform input with 1D convolutions, the recent TFNet [11] takes a time-frequency input with a causal 2D encoder-temporal filtering-decoder paradigm for end-to-end speech coding. Among all these methods, the latent features from the encoder are mostly blindly learned without any prior and there are typically temporal correlations remaining in them. We propose the predictive coding in this paper to further remove the redundancies.

B. Predictive Coding

Classical audio compression DPCM [17]/ADPCM [18] as a way of predictive coding, is widely used in classical audio coding systems. The DPCM leverages differential coding to remove the temporal redundancy among samples or acoustic parameters, where the difference between a sample and its estimate by a predictor based on the past reconstructed samples, is encoded. Another widely used technique in speech coding and processing, the linear predictive coding (LPC), leverages a linear predictor to estimate future samples based on the source-filter model. LPC analyzes a speech signal by estimating the predictor coefficients to minimize the energy of residual signals. The residual along with the linear coefficients are quantized and encoded.

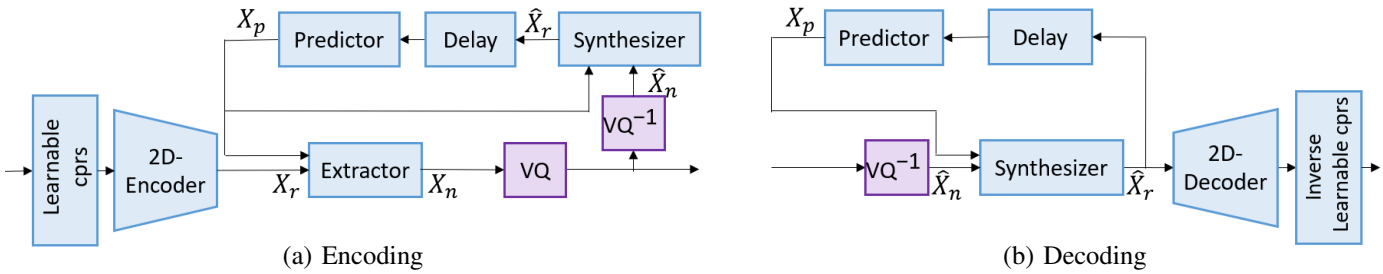
Classical video/image compression Traditional video coding standards [14]–[16] always take a predictive coding paradigm for removing temporal redundancies, where a prediction is generated by block-based motion estimation and compensation and the residual between the original frame and the prediction is transformed, quantized and entropy coded. In image coding [13] and intra-frame coding of video [14]–[16], reconstructed neighboring blocks are used to predict the current block, either in frequency or pixel domain, and the predicted residuals are encoded.

Deep video compression In neural video coding, a typical approach is to replace handcrafted modules such as motion estimation with neural networks , still sticking to a predictive coding paradigm. DVC [19] provides more accurate temporal predictions by jointly trained motion estimation and compensation networks. The residual information after prediction is then encoded by a residual encoder network. The most relevant work, DCVC [20], instead proposes a paradigm shift from predictive coding to conditional coding. It introduces rich temporal context information as a condition for both the encoder and the decoder and largely improves the coding efficiency.

Motivated by these methods, we introduce a predictive coding into the VQ-VAE framework for neural audio coding, to better remove temporal redundancies and achieve better coding efficiency.

C. Autoregressive Model

Autoregressive generative models have shown strong capabilities in speech synthesis [21], [23]. They typically generate audio samples one at a time in an autoregressive way where previous generated samples are used in generating current sample. Our predictive coding also uses an autoregressive way but it operates in latent domain to reduce the delay by the autoregressive loop and crosses the quantization layer.



(a) Encoding (b) Decoding

Fig. 2. Encoding and decoding modules for proposed method.

D. Vector Quantization

Vector quantization (VQ) is a fundamental technique that is widely used in traditional audio codecs such as Opus [24] and CELP [25]. Recently, it is also applied to discrete representation learning [12] and serves as the basis of end-to-end neural audio coding [6]–[11]. As quantization is inherently not differentiable, to enable end-to-end learning in neural audio coding, several ways have been proposed in the literature, including the one with commitment loss in VQ-VAE [12], EMA [12], Gumbel-Softmax based method [26] [27] and the soft-to-hard technique [28]. VQ-VAE [12] approximates the derivative by the identity function that directly copies gradients from the decoder input to the encoder output. The codebook is learned by approaching the codeword selected through some distance metric towards the encoder features. Differently, the Gumbel-Softmax and soft-to-hard methods introduce probability of selecting a codeword into the VQ, making the selection of discrete codewords in a differentiable way. However, the former uses a linear projection with Gumbel-Softmax to get the probability where there is no explicit correlation with quantization error. The latter maps distance to probability and uses soft assignment with annealing during training, which may lead to a gap between training and inference. Motivated by these works, we propose a Distance-Gumbel-Softmax scheme with rate control that explicitly maps quantization error to probability while uses hard assignment during training and inference.

III. The Proposed Scheme

A. Overview

Let x denote the signal to be encoded and $\hat{x}$ the recovered signal after decoding. The optimization of neural audio coding targets at minimizing the recovered signal distortion $D(x, \hat{x}|\Theta)$ at a given rate constraint, i.e. $R(x|\Theta) \leq R_{target}$. Θ denotes the neural network parameters. We consider low-latency speech coding in this paper.

As shown in Fig. 2, we employ an encoder to extract latent representations X_r from x. For each frame X_r^t in X_r, a prediction X_p^t is learned from past reconstructed latent codes $\hat{X}_r$ through a predictor f_{pred} with a receptive field of N past frames, given by $X_p^t = f_{pred}(\hat{X}_r^{t-i}|i = 1, 2, ..., N)$. This prediction serves as a temporal context for both encoding and decoding. For encoding, an extractor f_{extr} learns residual-like information X_n^t from both X_r^t and X_p^t by $X_n^t = f_{extr}(X_r^t, X_p^t)$, which is "new" to past frames. With this auto-regressive operation, the temporal redundancy could be effectively reduced. The extracted residual-like feature is then quantized through a codebook learned by Distance-Gumbel-Softmax and entropy coded using Huffman coding. For decoding, the quantized residual-like feature $\hat{X}_n^t$ is merged with the predicition X_p^t through a synthesizer f_{synr} to get the current reconstructed latent code $\hat{X}_r^t$, given by $\hat{X}_r^t = f_{synr}(\hat{X}_n^t, X_p^t)$. Then a decoder is employed to reconstruct the waveform $\hat{x}$. We apply adversarial training to achieve good perceptual quality. In the following subsections, we will describe these techniques in detail.

B. Learnable Input Compression

The input waveform x is first transformed into frequency domain with short-time fourier transform (STFT), yielding time-frequency spectrum $X \in \mathbb{R}^{T\times F\times 2}$, where T is the number of frames and F is the number of frequency bins. We take frequency domain input instead of the time domain widely adopted in previous works [8], [12] because frequency domain matches human perception well. As frequency domain input typically exhibits a high dynamic range and highly unbalanced distribution due to harmonics, we employ a power-law compression on the amplitude part given by A^p, where A is the magnitude spectrum of X. The compression performs as a kind of input normalization so that the importance of different frequencies is balanced and the training is more stable. Further, we make the power parameter p learnable during training so that at different bitrates the model can make adaptable balances. To be specific, at low bitrates a higher p may be preferred because it leads to more attention on main components while at high bitrates, more attention might be paid on details with a lower p. This technique is verified to particularly effective for very low bitrate coding such as 1kbps in our experiments.

C. Encoder and Decoder

The encoder takes the compressed time-frequency spectrum $X_{cprs} \in \mathbb{R}^{T\times F\times 2}$ as input. As shown in Fig. 3, four 2D causal convolutional layers are first employed to decorrelate it in two dimensions (T, F) with a kernel size of $(2, 5)$, output channels of $16, 32, 64$ and 64 and a stride of $1, 4, 4$ and 2 along the frequency F dimension. The temporal dimension T is kept without any resampling. This will yield a feature $X_1 \in \mathbb{R}^{T\times F'\times C'}$ with a reshape into a 1D signal $X_1' \in \mathbb{R}^{T\times C}, C = F' \times C'$. To capture long-range temporal

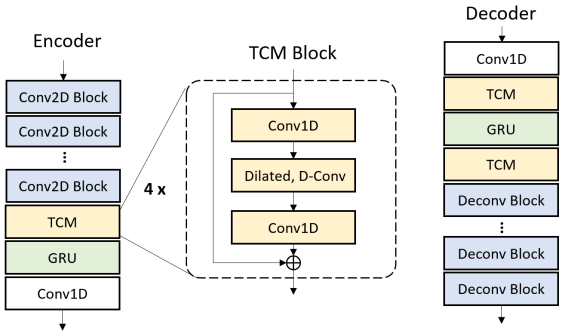


Fig. 3. Architecture of the encoder and the decoder. D-Conv denotes depthwise convolution.

dependencies, we further employ a TCM module with causal dilated depthwise convolutions [29] followed by a GRU block on X_1', like that in [11], to capture both short-term and long-term temporal dependencies. A final 1D convolutional layer with a kernel size of 1 is used to change the channel dimension to D for quantization with predictive coding. The encoder finally yields an output $X_r \in \mathbb{R}^{T\times D}$.

The decoder is the opposite of the encoder to reconstruct $\hat{x}$ from reconstructed features $\hat{X}_r$. For better restoration, more TCM modules are used at the decoder than the encoder. There are one TCM module, one GRU block and another TCM module used in an interleaved way to capture local and global temporal dependencies at different depths. Causal deconvolutions are used to recover the frequency resolution to F and the decoder outputs a feature $\hat{X}_{cprs} \in \mathbb{R}^{T\times F\times 2}$. After an inverse amplitude compression and an inverse STFT, the waveform $\hat{x}$ is finally reconstructed. The whole process is causal so that it can achieve low latency.

D. Latent-Domain Predictive Coding

As the predictive coding is auto-regressive, to reduce the delay we only investigate it in latent domain. As shown in Fig. 1(b) and the encoding/decoding split in Fig. 2, there is a predictor f_{pred} to get the prediction for each t-th frame from past reconstructed latent features $\{\hat{X}_r^{t-i}|i=1,2,...,N\}$. As the prediction X_p^t may contain some undesired information for encoding frame t, instead of residual coding by $X_r^t - X_p^t$, we concat X_r^t and X_p^t and then feed them to an learnable extrator f_{extr} to extract "new" information X_n^t which could not be estimated from the past. X_n^t is then quantized using vector quantization. Symmetrically, for reconstructing latent representation of current frame, a learnable synthesizer f_{synr} is employed to merge X_p^t and dequantized output $\hat{X}_n^t$ to get $\hat{X}_r^t$ for both encoding and decoding.

Predictor The predictor provides a prediction of current frame from the past, given by $X_p^t = f_{pred}(\hat{X}_r^{t-i}|i=1,2,...,N)$, with a window of N frames. We investigate two ways for this prediction, i.e. convolution-based and adaptive as shown in Fig. 4. The former uses two 1D convolutional layers by parametric ReLU (PReLU) [30] to get a receptive field of 280ms. The latter learns the prediction kernel from the past to adapt to the time-varying speech signal. The kernel is deduced from the

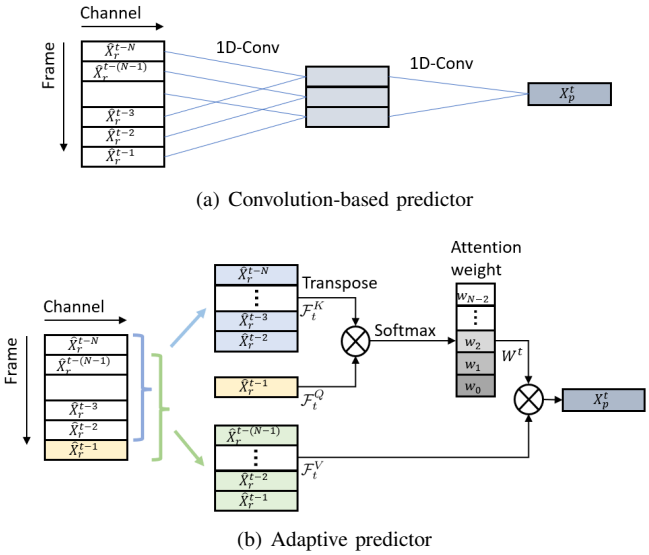


Fig. 4. The network structure of the predictor.

past based on the assumption that the linear prediction coefficients are locally constant. Specifically, it takes $\hat{X}_r^{t-1}$ as a query matrix $\mathcal{F}_t^Q = [\hat{X}_r^{t-1}]$, $\{\hat{X}_r^{t-i}|i=2,3,...,N\}$ as the key matrix $\mathcal{F}_t^K = [\hat{X}_r^{t-N}, \hat{X}_r^{t-N+1}, ..., \hat{X}_r^{t-2}]$ and $\{\hat{X}_r^{t-i}|i=1,2,...,N-1\}$ as the value matrix $\mathcal{F}_t^V = [\hat{X}_r^{t-(N-1)}, \hat{X}_r^{t-(N-2)}, ..., \hat{X}_r^{t-1}]$. It learns an attentive weight vector $W^t \in R^{N-1}$ which is served as the prediction kernel by

$$W^t = Softmax(\mathcal{F}_t^Q \cdot (\mathcal{F}_t^K)^{Transpose}/\sqrt{D}), \quad (1)$$

where $Softmax(\cdot)$ is the softmax function. The attentive weight vector is then multiplied with $\mathcal{F}_t^V$ to get the prediction by $p_t = W^t \cdot \mathcal{F}_t^V$. This method is similar to self-attention [31] in the way to adaptively capture the attention weights from input features but here we extend it as a kind of prediction. We will show the comparison between these two types of predictors in the experimental part.

To guide the predictor to yield a good temporal prediction for redundancy removal, a prediction loss is introduced in the training as

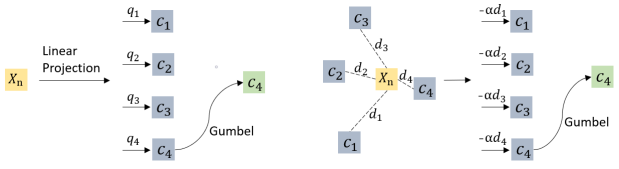
$$\mathcal{L}_{pred} = \mathbb{E}(D(X_p, sg(X_r))), \quad (2)$$

where $D(\cdot)$ is a distance metric given by ℓ_1 in our implementation. $sg(\cdot)$ is the stop-gradient operator, used for more stable training.

Extractor and synthesizer Both the extractor f_{extr} and the synthesizer f_{synr} consist of a 1D convolutional layer with a kernel size of 1 and a stride of 1, followed by parametric ReLU as the nonlinear activation function.

E. Vector Quantization with Rate Control

As discussed in Section II.D, Gumbel-Softmax [26] [27] and soft-to-hard [28] methods introduce the probability of selecting a codeword and thus make rate control feasible. However, Gumbel-Softmax uses a linear projection to select the codeword without explicitly correlating it with the quantization error, as shown in Fig. 5(a). The soft-to-hard gives



(a) Gumbel-Softmax (b) Distance-Gumbel-Softmax

Fig. 5. Vector Quantization mechanism. (a) Gumbel-Softmax in [26]. Latent n is projected to logits q_i through a linear projection and turned into probabilities with Gumbel-Softmax. (b) Our Distance-Gumbel-Softmax. Distance between latent n and codewords c_i is first calculated and then mapped to probabilities with Gumbel-Softmax.

soft assignments based on distances with different codewords but a weighted average of codewords is used for quantization in training, which can easily lead to a gap between training and inference. In light of this, we employ a Distance-Gumbel-Softmax method as shown in Fig. 5(b) for quantization which leverages the advantages of the two to provide a quantization-error-aware assignment with rate control.

Distance-Gumbel-Softmax-based VQ As shown in Fig. 5(b), given a codebook with K codewords $C = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_K\} \in \mathbb{R}^{K\times D}$, we first compute the distance between the current latent vector $X_n^t \in \mathbb{R}^D$ and all K codewords as

$$\boldsymbol{d}_t = [D(X_n^t, \boldsymbol{c}_1), ..., D(X_n^t, \boldsymbol{c}_K)] \in \mathbb{R}^K, \tag{3}$$

where $D(\cdot)$ is a distance metric and we use ℓ_2 in our implementation. Then the distance is mapped to logits and we employ the Gumbel-Softmax to get probability for assignment, that is

$$\boldsymbol{q}_t = GumbelSoftmax(-\alpha \cdot \boldsymbol{d}_t) \in \mathbb{R}^K. \tag{4}$$

The probability for selecting the k-th codeword $\boldsymbol{c}_k$ to quantize X_n^t is given by

$$q_{t,k} = \frac{exp((-\alpha \cdot d_{t,k} + v_{t,k})/\tau)}{\sum_{i=1}^K exp((-\alpha \cdot d_{t,i} + v_{t,i})/\tau)}, \tag{5}$$

where τ is the temperature of Gumbel-Softmax and $v_{t,k} \sim Gumbel(0, 1)$ are samples drawn from Gumbel distribution. α is a positive scalar to control the mapping from distance $D(X_n^t, \boldsymbol{c}_k)$ to logits such that the codeword closer to the current feature X_n^t will have a higher probability of being selected. During the forward pass, the hard index $argmax_{k\in\{1,2,..,K\}} q_{t,k}$ is selected; thus there is no gap between training and inference. During the backward pass, the gradient with respect to logits is used.

Entropy estimation and rate control As entropy serves as the lower bound of actual bitrates, we leverage entropy estimation to control the bitrate $R(x|\Theta)$ towards a given target R_{target}, motivated by the work in [10], [28]. Using the Distance-Gumbel-Softmax based VQ, we can calculate the sample soft assignment distribution, by summing up the probabilistic assignment logits to each codeword within a minibatch $q_{t,k,b}$ as

$$Q_k = \frac{1}{BT}\sum_{b=1}^{B}\sum_{t=1}^{T} q_{t,k,b}, k \in \{1, 2, ..., K\}, \tag{6}$$

where B and T are batch size and number of frames in each audio clip, respectively. Then we can estimate the "soft entropy" on the soft assignment distribution $\boldsymbol{Q} \in \mathbb{R}^{\boldsymbol{K}}$ as

$$\mathcal{H}(\boldsymbol{Q}) \approx -\sum_{k=1}^{K} Q_k \log Q_k. \tag{7}$$

The rate control is conducted over each minibatch with the following loss function $\mathcal{L}_{rate}$:

$$\mathcal{L}_{rate} = ||R_{target} - \mathcal{H}(\boldsymbol{Q})||_1. \tag{8}$$

This loss $\mathcal{L}_{rate}$ not only constraints the bitrate but also performs rate-distortion optimization by $\mathcal{L}_{RD} = D(x, \hat{x}) + \lambda \cdot \mathcal{L}_{rate}$. When the current entropy is higher than R_{target}, it will push similar features quantized to the same codeword through a tradeoff between rate and distortion; whereas when it is lower than R_{target}, similar features may be quantized to different codewords to retain a higher quality but higher rate. It should be noted that although there are some estimations here, we found that the actual bitrate is controlled well during testing.

To reduce the codebook size for easy training, the group vector quantization is employed. Specifically, for each frame $X_n^t \in \mathbb{R}^D$ is split into G groups along the channel dimension, yielding $X_n'^t \in \mathbb{R}^{\frac{D}{G}\times G}$, and each group is quantized with a separate codebook with K codewords $C = \{c_1, c_2, ..., c_K\} \in \mathbb{R}^{K\times\frac{D}{G}}$. We set up a large codebook size so that it could capture the real distribution of the latent features through the rate-distortion optimization. For example, at 3kbps each 40ms data (four overlapped STFT frames) is expected to consume 120 bits. The codebook parameters G and K are set to 16 and 1024, respectively, where $G \cdot log_2(K) = 16 \cdot log_2(1024) = 160 > 120$. The real bitrate is then controlled by Eq. 8 to achieve 3kbps. This is quite different from the diversity loss in Gumbel-Softmax based method [26], where a uniform distribution is enforced on the codeword usage.

F. Adversarial Training

Adversarial training has been shown to be very effective in high quality speech generation [32] [33]. For high reconstructed perceptual quality, we also employ adversarial training in our scheme with a frequency-domain discriminator. It takes the complex time-frequency spectrum of the input waveform as input. The magnitude spectrum is power-law compressed with a power of 0.3 to balance the importances of different components. Four 2D convolutional layers with a kernel size of 3×2 and a stride of (2,2) are used to extract features with progressively reduced resolutions in both time and frequency dimensions. The channel numbers are 8,8,16 and 16, respectively. Each convolutional layer is followed by an instance normalization (IN) and a Leaky ReLu [34]. Finally, a linear transformation is used to fold all frequency information into channels followed by a temporal pooling layer to aggregate information across the time dimension and produce the final logit.

We use the least-square loss as the adversarial objective like that in LSGAN [35]. The adversarial loss for the generator G is

$$\mathcal{L}_{adv} = \mathbb{E}_x[||D(G(x)) - 1||_2^2]. \tag{9}$$

And the loss for the discriminator D is

$$\mathcal{L}_D = \mathbb{E}_x[||D(x)-1||_2^2] + \mathbb{E}_x[||D(G(x))||_2^2]. \quad (10)$$

We also employ a feature loss $\mathcal{L}_{feat}$ to guide the training of the generator for high perceptual quality. It is computed as the ℓ_1 difference of the deep features from the discriminator between the generated and the original audios, given by

$$\mathcal{L}_{feat} = \mathbb{E}_x[\frac{1}{L}\sum_{i=1}^{L}||D^i(x) - D^i(G(x))||_1], \quad (11)$$

where D^i is the feature map of the i-th layer of the discriminator. We compute the feature loss on the first four 2D convolutional layers of the discriminator.

G. Objective Function

We employ the following loss function to guide the training for maximized output audio quality at the target bitrate. The total loss for the generator consists of a reconstruction term $\mathcal{L}_{recon}$, a rate-constraint term $\mathcal{L}_{rate}$, a prediction term $\mathcal{L}_{pred}$, an adversarial term $\mathcal{L}_{adv}$, and a feature-matching term $\mathcal{L}_{feat}$, i.e.

$$\begin{aligned}\mathcal{L}_G = \mathcal{L}_{recon} + \lambda_{rate}\mathcal{L}_{rate} + \lambda_{pred}\mathcal{L}_{pred} \\ +\lambda_{adv}\mathcal{L}_{adv} + \lambda_{feat}\mathcal{L}_{feat},\end{aligned} \quad (12)$$

where $\mathcal{L}_{rate}$, $\mathcal{L}_{pred}$, $\mathcal{L}_{adv}$ and $\mathcal{L}_{feat}$ have been explained in Eq. 8, 2, 9 and 11, respectively. The reconstruction term is selected to achieve both high signal fidelity and high perceptual quality. We use two frequency-domain terms for $\mathcal{L}_{recon}$, as shown below

$$\mathcal{L}_{recon} = \mathcal{L}_{bin} + \lambda_{mel}\mathcal{L}_{mel}. \quad (13)$$

The first term $\mathcal{L}_{bin}$ is the mean-square-error (MSE) loss on the power-law compressed STFT spectrum [36]. To keep STFT consistency [37], the reconstructed spectrum is first transformed to time domain and then to the frequency domain to calculate the loss. The second term $\mathcal{L}_{mel}$ is the multi-scale mel-spectrum loss given by

$$\mathcal{L}_{mel} = \mathbb{E}_x[\frac{1}{W}\sum_{n=1}^{W}||\phi^n(x) - \phi^n(\hat{x})||_1], \quad (14)$$

where $\phi^n(\cdot)$ is the function that transforms a waveform into the mel-spectrogram using the n-th window size. Following [38], we calculate the mel spectra over a sequence of window-lengths between 64 and 2048. All scalars λ_{rate}, λ_{pred}, λ_{adv}, λ_{feat} and λ_{mel} to balance different terms are decided during experiments.

IV. Experimental Results

In this section, we evaluate the proposed TF-Codec against the state of the art and provide a detailed analysis on each part to show what it learns and why it works.

A. Datasets and Settings

We take 890 hours of 16kHz clean speech from the Deep Noise Suppression Challenge at ICASSP 2021 [39], including multilingual speech, emotional and singing clips. Each audio is cut into 3-second clips with a random speech level from $[-50, -10]$db for training. For evaluation, we use 1458 clips of 10 seconds without any overlapping with the training data, which covers more than 1000 speakers with multiple languages. Hanning window is used in STFT with a window length of 40 ms and a hop length of 10 ms.

All the modules of the TF-Codec including the encoding, decoding and quantization could be trained end to end in a single stage. For adversarial training, we first train a good generator from end to end and then finetune the generator with a discriminator in an adversarial way.

During training, we use Adam optimizer [40] with a learning rate of 3×10^{-4} for the generator in the first stage. Then the generator and discriminator are trained with a learning rate of 3×10^{-5} and 3×10^{-4}, respectively. We train both stages for 100 epochs with a batch size of 100.

B. Comparison with State-of-the-Art Codecs

We first compare the proposed TF-Codec with several standard codecs to show the strong representation capability of our backbone. We conduct a subjective listening test by a MUSHRA-inspired crowd-sourced method [41], where 10 participants evaluate 15 samples from the test set. In MUSHRA evaluation, the listener is presented with a hidden reference and a set of test samples generated by different methods. The lowpass-filtered anchor is not used in our experiment.

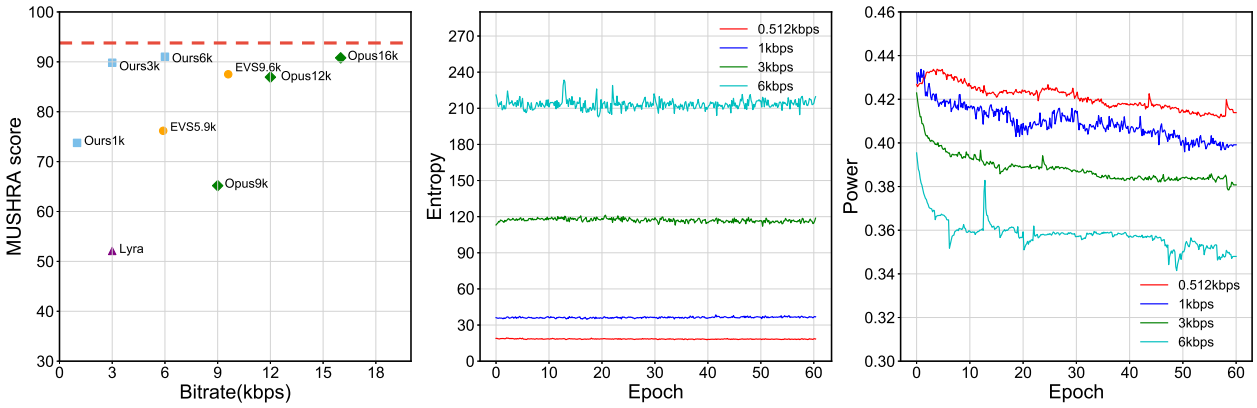
We compare the TF-Codec with three real-time audio codecs, i.e., Opus, EVS and Lyra. Opus[1] [24] is a versatile codec widely used for real-time communications, which supports narrowband to fullband speech and audio with a bitrate from 6kbps to 510kbps. EVS codec [42] is developed and standardized by the 3GPP primarily for Voice over LTE (VoLTE). Lyra[2] [2] is an autoregressive neural speech codec proposed recently, which reconstructs high quality speech at 3 kbps.

Fig. 6(a) shows the evaluation results, where we compare our TF-Codec from 1kbps to 6kbps to Lyra, Opus and EVS at various bitrates. It is observed that our TF-Codec at 1kbps significantly outperform Lyra at 3kbps and Opus at 9kbps, showing the strong representation capability of the TF-Codec. When operating at 3kbps, our TF-Code achieves better performance than EVS at 9.6kbps and Opus at 12kbps. In the higher bitrate range, our TF-Codec at 6kbps performs on par with Opus at 16kbps. Besides, Fig. 6(b) shows that the total entropy of our TF-Codec are under the control with the rate loss $\mathcal{L}_{rate}$ during training. During testing, we also found that the actual bitrates are controlled well after Huffman coding.

C. Ablation Study

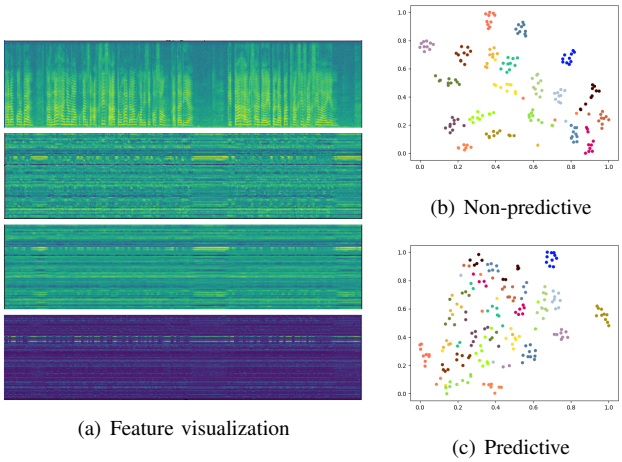
To evaluate different parts of the proposed method, we employ several objective metrics including the wideband PESQ

[1]https://opus-codec.org
[2]https://github.com/google/lyra



(a) TF-Codec vs. standard codecs (b) Entropy (c) The learnable power

Fig. 6. (a) Subjective evaluation results. The red dotted line represents the score of the reference. (b) Entropy for 40ms data during training. (c) The learned power coefficient during training. In (b) and (c), the curves correspond to the adversarial training stage only.



(a) Feature visualization (b) Non-predictive (c) Predictive

Fig. 7. Feature visualization of contextual coding. (a) The four rows from upper to bottom show the STFT spectrum (log-scale) of the uncompressed audio, the output of the encoder before predictive coding, the output of the predictor and that of the extractor, respectively. Values are linearly normalized between 0 and 1. (b)(c) T-SNE visualization of speaker information by non-predictive and predictive coding, respectively.

TABLE I
EVALUATION ON PREDICTIVE CODING AT 3KBPS(W/O. ADV).

Methods	PESQ	STOI	ViSQOL
TF-Codec w/o. Prediction	2.763	**0.917**	3.219
TF-Codec w. Adapt	2.774	0.914	3.332
TF-Codec w. Conv	**2.895**	**0.917**	**3.345**

[43], STOI [44] and the VISQOL [45]. Although they are not designed and optimized for exactly the same task, we found that for the same kind of distortions in all compared schemes, they match well with perceptual quality.

1) Predictive coding: We first evaluate the effectiveness of predictive coding. We compare the two variations of the TF-Codec by convolution-based and adaptive predictors, respectively, with the one without predictive coding by disabling the predictive loop. Table I shows the evaluation results where the adversarial training is disabled for all compared methods. It can be seen that when all operating at 3kbps, the predictive coding improves the reconstructed audio quality by both PESQ and ViSQOL with similar speech intelligibility by STOI. The convolution-based method outperforms the adaptive mechanism because after quantization the assumption for local constant linear prediction may not hold any more in adaptive scheme.

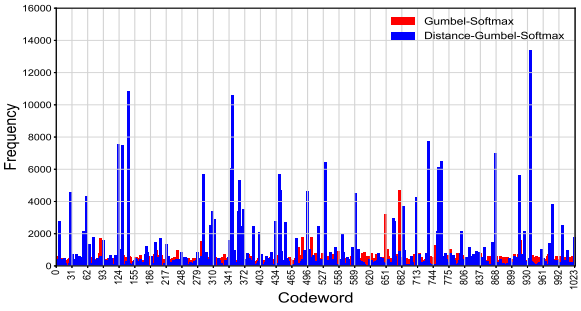
To further look into the representations it learns, we visualize the features of different modules by predictive coding in Fig. 7(a). The four rows from upper to bottom show the STFT spectrum of the uncompressed audio, the output of the encoder X_r before predictive coding, the output of the predictor X_p and that of the extractor X_n. It can be found that the prediction X_p is quite similar to X_r, indicating that the predictor provides a good prediction of the current frame from the past. We can also observe that the feature X_n after the extractor becomes much sparser than X_r, indicating that most redundant information has been removed. We also calculate the temporal correlation coefficient of the learned representation, i.e., the last layer output of the encoding. The non-predictive coding without the predictive loop achieves an average correlation coefficient of 0.37 while predictive coding reduces it to 0.09, showing that the temporal correlation is removed more thoroughly in predictive coding. This is also consistent with the visualization results in Fig. 7(a).

To further explore what redundant information has been removed by predictive coding, we show the t-SNE [46] visualization of the speaker information contained in the learned representations in Fig. 7(b)(c) for non-predictive and predictive coding, respectively. To achieve this, we perform a temporal pooling on the learned representation, yielding one embedding vector for one audio. The utterances of 20 randomly selected speakers from the Librispeech dataset [47] are used for visualization. We can observe that representations from non-predictive coding are clustered well for each speaker, showing that they contain most speaker information; while in predictive coding, the embeddings scatter for most speakers indicating that the speaker information is effectively removed. This is reasonable as speaker-related information is relatively constant in time and easy to predict.

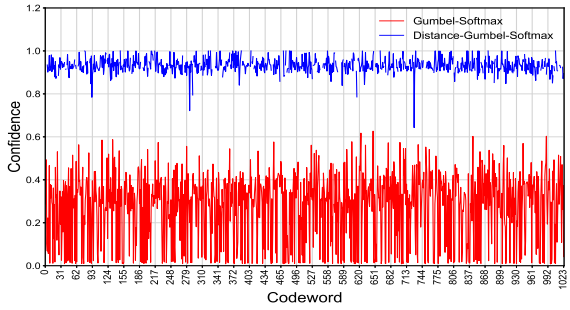
2) Learnable input compression: To show the effectiveness of the learnable input compression, we compare it with a

TABLE II
EVALUATION ON LEARNABLE INPUT COMPRESSION AT 1KBPS.

Methods	PESQ	STOI	ViSQOL
fixed compression	2.289	0.877	2.781
learnable compression	**2.351**	**0.887**	**2.851**



(a) Codeword usage histogram



(b) Codeword confidence

Fig. 8. Characteristics of one randomly selected learned codebook at 3kbps. (a) The frequency of 1024 codewords being selected. (b) The confidence sore of 1024 codewords.

fixed power-law compression where the power parameter is set to 0.3 as that in [36]. Table II shows that at 1kbps, the learnable compresssion clearly outperforms the fixed one in all three metrics, showing both better perceptual quality and better speech intelligibility. In our subjective evaluation, we also found obvious perceptual quality boost for very low bitrate such as 1kbps. To see what power parameters it learns, we also report the learned power p at various bitrates during the training as shown in Fig. 6(c). We can see that the learned power gradually decreases during the training process, indicating that the model first mainly looks at high-energy bins, usually the low-frequency bands, and as the epoch increases, the model turns to pay more attention to the low-energy details of the spectrum. It is also observed that the higher the bitrate, the smaller the p, which means that the model tries to look at the detailed components with more bits available, yielding better perceptual quality.

3) Distance-Gumbel-Softmax-based VQ: We compare the Distance-Gumbel-Softmax-based VQ mechanism with the previous Gumbel-Softmax-based method in [26]. Table III shows that at 3kbps, our method outperforms the previous Gumbel-Softmax-based method in all metrics, indicating that the explict injection of distance information helps improve reconstruction quality.

TABLE III
EVALUATION ON VECTOR QUANTIZATION AT 3KBPS.

Methods	PESQ	STOI	ViSQOL
Gumbel-Softmax	2.738	0.910	3.204
Distance-Gumbel-Softmax	**2.763**	**0.917**	**3.219**

TABLE IV
EVALUATION ON ADVERSARIAL TRAINING.

Methods	Bitrate	PESQ	STOI	ViSQOL
TF-Codec w/o Adv.	1kbps	2.085	0.868	2.742
TF-Codec w Adv.	1kbps	**2.351**	**0.887**	**2.851**
TF-Codec w/o Adv.	3kbps	2.763	0.917	3.219
TF-Codec w Adv.	3kbps	**3.124**	**0.933**	**3.510**
TF-Codec w/o Adv.	6kbps	3.426	0.949	**3.966**
TF-Codec w Adv.	6kbps	**3.547**	**0.953**	3.841

We also show the distribution of the learned codebooks to help understand how the Distance-Gumbel-Softmax-based vector quantization learns. Fig. 8(a) shows the usage of 1024 codewords of one codebook on the test set with 1458 audios for both the Gumbel-Softmax-based method [26] and the proposed Distance-Gumbel-Softmax mechanism. We can observe that the codewords tend to be more uniformly distributed in Gumbel-Softmax-based method; while in the Distance-Gumbel-Softmax, the codewords are distributed more diversely with some codewords being used very frequently. This is reasonable as in Gumbel-Softmax-based method, a diversity loss $\mathcal{L}_{diversity}$ is imposed on the learned codewords which encourages each codeword to be equally used; while in Distance-Gumbel-Softmax, we employ a larger codebook and use the rate loss $\mathcal{L}_{rate}$ to reach the target bitrate in a rate-distortion optimization sense. In this way, real distribution of the latent features could be captured in Distance-Gumbel-Softmax.

We also show the confidence score of selecting the best codeword in Fig. 8(b), based on the learned soft probability by softmax. The Distance-Gumbel-Softmax shows obviously much higher confidence than Gumbel-Softmax, which indicates that the learned codebook in Distance-Gumbel-Softmax has more distinct class centers. This is due to the explicit introduction of distance map, i.e., the quantization error, into the soft probability, by which the codeword closer to the current feature is encouraged to be selected and more distinct codewords are learned through back propagation.

4) Adversarial training: We also conduct ablation study to evaluate the performance of adversarial training as presented in Table IV. We show the comparison with and without adversarial training at various bitrates. It is observed that by adversarial training, the PESQ, STOI and ViSQOL are largely improved especially at 1kbps and 3kbps. We also observe that the feature-matching loss $\mathcal{L}_{feat}$ plays a quite important role in recovering high-frequency details in the generated audio in our experiment.

D. Analysis

To better understand what information is learned and encoded at various bitrates, we make some analysis on the learned representations and codebooks in this section. In this

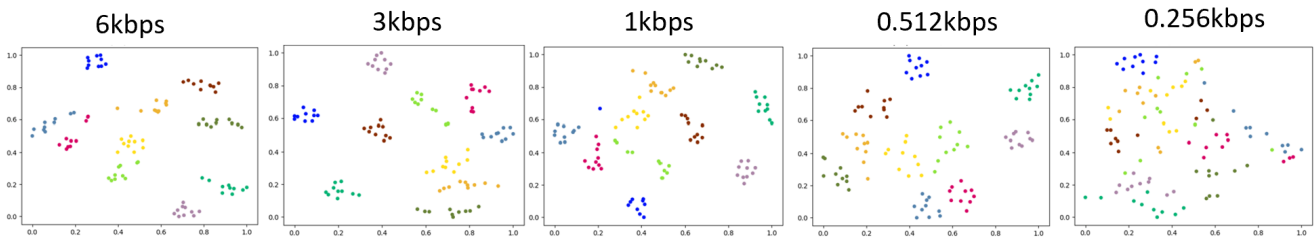
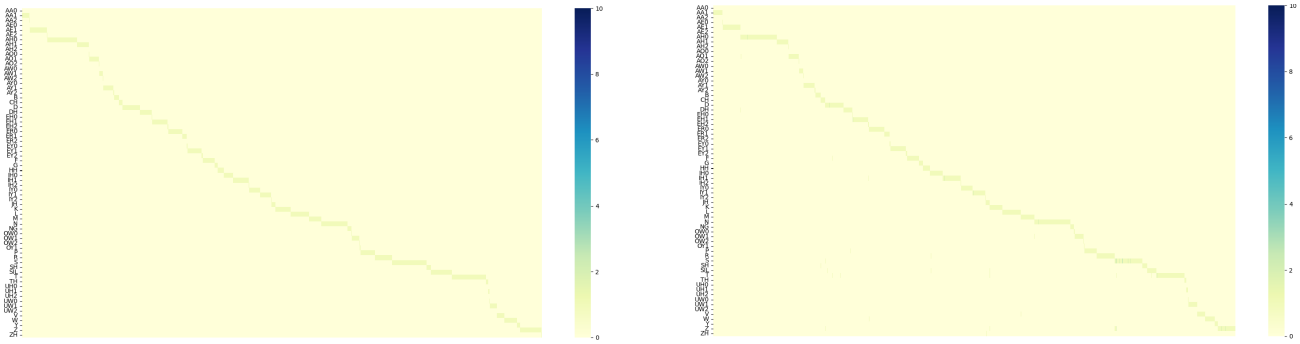


Fig. 9. T-SNE of speaker information in discrete latent codes on Librispeech dataset.



(a) 3kbps (b) 0.512kbps

Fig. 10. Co-occurrence of latent codes and phonemes on LJ single speaker dataset. The horizontal axis is latent code index and the vertical axis denotes phonemes. We obtain the phoneme-level alignments with the Montreal Forced Aligner (MFA) [48], using their pre-trained Librispeech acoustic model. The frame-level phoneme labels are determined by the phoneme with most occurrences in the duration of each frame.

analysis, we disable the predictive loop and choose the discrete latent codes for visualization.

1) What discrete features it learns: We first visualize the speaker and content information contained in the learned discrete latent codes. We employ two datasets: (i) Librispeech multi-speaker dataset [47] for speaker-related information analysis; (ii) LJ single speaker dataset [49] for linguistic information analysis.

Speaker information We randomly select 10 speakers from Librispeech, each with 10 utterances. For each utterance, we perform a temporal average pooling on the multi-frame features, yielding a global embedding per utterance. Fig. 9 shows the t-SNE visualization of those speaker embeddings from 0.256kbps to 6kbps. We can observe that the model at high bitrates generates more compact speaker clusters while at very low bitrates, the cluster begins to diffuse and speakers could not be identified at 0.256kbps. This indicates that at very low bitrates, the model turns to drop speaker-related information so as to leave bandwidth for some key information of speech. It is worth noting that the bitrate 0.256kbps is even close to the estimated information rate of speech communication in [50]. At such low bitrates, linguistic information is more important than speaker variations for real-time communications.

Linguistic information We evaluate the linguistic information in the discrete codes by the co-occurence map between phonemes and discrete latent codes by group vector quantization. We use all 13100 audio clips of the LJSpeech dataset spoken by the same speaker to remove the impact by speaker variations. Fig. 10 shows the co-occurence map at difference bitrates. It can be seen that these discrete latent codes learned in a self-supervised way are closely related to phonemes and many latents are dedicated to specific phonemes. For example, a large amount of discrete codewords are automatically allocated to specific phonemes, e.g., AH, N, S and T. It is also observed that the distributions of the co-occurrence map for 0.512kbps and 3kbps are quite close to each other, indicating that the latent codes at different bitrates preserve phoneme information well. This is consistent with our hypothesis that at extremely low bitrates the model tries to allocate the limited bandwidth to key content information (linguistic-related) in speech and drop less important information (speaker-related).

2) Generalization to music and highly revereberated data: To further verify what information the codebooks capture, we perform the testing on unseen music and highly reverberated speech data using model trained on speech data only without high reverberation. We use the MagnaTagATune dataset [51] for music and use the RIRs in OpenSLR [52] to synthesize highly reverberated speech audios. Interestingly, we found that although there is data type mismatch between training and testing, the model generalizes well at 6kbps. This is attributed to the strong representation capability by multiple codebooks in group vector quantization. Different groups capture different feature spaces while in some spaces these different types of data may overlap with each other. When the bitrate decreases, the generalization capability drops as at low bitrates more key speech information is encoded.

V. Conclusion

We propose the TF-Codec, a low-latency neural speech codec that outperforms state-of-the-art audio codecs with very low bitrates. We introduce the latent-domain predictive coding into the VQ-VAE framework to fully remove the temporal

redundancy. A learnable input compression is proposed to balance the attention paid on main components and details in the STFT spectrum at different bitrates. We also introduce the Distance-Gumbel-Softmax mechnism for vector quantization which could capture the real distribution of latent features with rate-distortion optimization. It should be noted that although speech coding is taken as an example in this paper, the proposed techniques could be extended to audio coding as well. In the future, we will investigate more detailed representations in terms of not only speaker and content information but also the prosody and emotions. We will also investigate how to promote the error resilience when the predictive coding loop is present.

REFERENCES

[1] W. Kleijin, F. Lim, A. Luebs, and J. Skoglund, "WaveNet based low rate speech coding," in *ICASSP*. IEEE, 2018, pp. 676–680.

[2] W. B. Kleijn, A. Storus, M. Chinen, T. Denton, F. S. Lim, A. Luebs, J. Skoglund, and H. Yeh, "Generative speech coding with predictive variance regularization," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6478–6482.

[3] J. Klejsa, P. Hedelin, C. Zhou, R. Fejgin, and L. Villemoes, "High-quality speech coding with sample RNN," in *ICASSP*. IEEE, 2019, pp. 7155–7159.

[4] R. Fejgin, J. Klejsa, L. Villemoes, and C. Zhou, "Source coding of audio signals with a generative model," in *ICASSP*. IEEE, 2020, pp. 341–345.

[5] J. Skoglund and J. Valin, "Improving Opus low bit rate quality with neural speech synthesis," in *Interspeech*, 2020.

[6] C. Gârbacea, A. van den Oord, Y. Li, F. Lim, A. Luebs, O. Vinyals, and T. C. Walters, "Low bit-rate speech coding with VQ-VAE and a WaveNet decoder," in *2019 IEEE Int. Conf. Acoust Speech Signal Processing (ICASSP)*. IEEE, 2019, pp. 735–739.

[7] J. Williams, Y. Zhao, E. Cooper, and J. Yamagishi, "Learning disentangled phone and speaker representations in a semi-supervised VQ-VAE paradigm," in *2021 IEEE Int. Conf. Acoust Speech Signal Processing (ICASSP)*. IEEE, 2021.

[8] 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*, 2021.

[9] K. Zhen, J. Sung, M. 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 and Communication Association (Interspeech)*, 2019.

[10] K. Zhen, J. Sung, M. S. Lee, and M. Kim, "Scalable and efficient neural speech coding: a hybrid design," *IEE/ACM Transactions on audio, speech, and language processing*, vol. 30, pp. 12–25, 2022.

[11] X. Jiang, X. Peng, C. Zheng, H. Xue, Y. Zhang, and Y. Lu, "End-to-end neural speech coding for real-time communications," in *IEEE Int. Conf. Acoust Speech Signal Processing (ICASSP)*, 2022.

[12] A. v. d. Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," *NIPS*, 2017.

[13] G. K. Wallace, "The jpeg still picture compression standard," *IEEE transactions on consumer electronics*, vol. 38, no. 1, pp. xviii–xxxiv, 1992.

[14] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h. 264/avc video coding standard," *IEEE Transactions on circuits and systems for video technology*, vol. 13, no. 7, pp. 560–576, 2003.

[15] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, pp. 1649–1668, 2012.

[16] B. Bross, J. Chen, J.-R. Ohm, G. J. Sullivan, and Y.-K. Wang, "Developments in international video coding standardization after avc, with an overview of versatile video coding (vvc)," *Proceedings of the IEEE*, vol. 109, no. 9, pp. 1463–1493, 2021.

[17] B. S. Atal, "Predictive coding of speech at low bit rates," *IEEE Transactions on Communications*, vol. 30, pp. 600–614, 1982.

[18] P. Cummiskey, N. S. Jayant, and J. L. Flanagan, "Adaptive quantization in differential pcm coding of speech," *The Bell System Technical Journal*, vol. 52, pp. 1105–1118, 1973.

[19] G. Lu, W. Ouyang, D. Xu, X. Zhang, C. Cai, and Z. Gao, "DVC: An end-to-end deep video compression framework," in *CVPR*, 2019.

[20] J. Li, B. Li, and Y. Lu, "Deep contextual video compression," in *NIPS*, 2021.

[21] A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[22] V. J.M. and S. J., "LPCNet: improving neural speech synthesis through linear prediction," in *ICASSP*. IEEE, 2019.

[23] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. Courville, and Y. Bengio, "SampleRNN: an unconditional end-to-end neural audio generation model," in *ICLR*, 2017.

[24] J.-M. Valin, K. Vos, and T. Terriberry, "Definition of the Opus audio codec," 2012.

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

[26] H. Zhou, A. Baevski, and M. Auli, "A comparison of discrete latent variable models for speech representation learning," in *2021 IEEE Int. Conf. Acoust Speech Signal Processing (ICASSP)*. IEEE, 2021.

[27] A. Baevski, S. Schneider, and M. Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," *arXiv preprint arXiv:1910.05453*, 2019.

[28] 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," *NIPS*, 2017.

[29] A. Pandey and D. Wang, "TCNN: Temporal convolutional neural network for real-time speech enhancement in the time domain," in *2019 IEEE Int. Conf. Acoust Speech Signal Processing (ICASSP)*. IEEE, 2019.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[32] J. Kong, J. Kim, and J. Bae, "Hifi-GAN: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 17 022–17 033, 2020.

[33] K. Kumar, R. Kumar, T. de Boissiere, L. Gestin, W. Z. Teoh, J. Sotelo, A. de Brébisson, Y. Bengio, and A. C. Courville, "MelGAN: Generative adversarial networks for conditional waveform synthesis," *Advances in neural information processing systems*, vol. 32, 2019.

[34] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.

[35] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2794–2802.

[36] A. Ephrat, I. Mosseri, O. Lang, T. Dekel, K. Wilson, A. Hassidim, W. T. Freeman, and M. Rubinstein, "Looking to listen at the cocktail party: A speaker-independent audio-visual model for speech separation," *arXiv preprint arXiv:1804.03619*, 2018.

[37] S. Wisdom, J. R. Hershey, K. Wilson, J. Thorpe, M. Chinen, B. Patton, and R. A. Saurous, "Differentiable consistency constraints for improved deep speech enhancement," in *ICASSP*. IEEE, 2019, pp. 900–904.

[38] A. Gritsenko, T. Salimans, R. van den Berg, J. Snoek, and N. Kalchbrenner, "A spectral energy distance for parallel speech synthesis," *Advances in Neural Information Processing Systems*, vol. 33, pp. 13 062–13 072, 2020.

[39] C. K. A. Reddy, H. Dubey, V. Gopal, R. Cutler, S. Braun, H. Gamper, R. Aichner, and S. Srinivasan, "ICASSP 2021 deep noise suppression challenge," *arXiv preprint arXiv:2009.06122*, 2020.

[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[41] ITU-R, "Recommendation BS.1534-1: Method for the subjective assessment of intermediate quality level of audio systems," *International Telecommunication Union Radiocommunication Assembly*, 2001.

[42] M. Dietz, M. Multrus, V. Eksler, V. Malenovsky, E. Norvell, H. Pobloth, L. Miao, Z. Wang, L. Laaksonen, A. Vasilache *et al.*, "Overview of the evs codec architecture," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5698–5702.

[43] I. Rec, "P.862.2: Wideband extension to recommendation p.862 for the assessment of wideband telephone networks and speech codecs," *International Telecommunication Union, CH–Geneva*, 2005.
[44] C.H.Taal, R.C.Hendriks, R.Heusdens, and J.Jensen, "A short-time objective intelligibility measure for time-frequency weighted noisy speech," in *ICASSP*, 2010.
[45] A. Hines, J. Skoglund, A. Kokaram, and N. Harte, "Visqol: The virtual speech quality objective listener," in *IWAENC 2012; International Workshop on Acoustic Signal Enhancement*. VDE, 2012, pp. 1–4.
[46] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.
[47] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.
[48] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal forced aligner: Trainable text-speech alignment using kaldi." in *Interspeech*, vol. 2017, 2017, pp. 498–502.
[49] K. Ito and L. Johnson, "The lj speech dataset," https://keithito.com/LJ-Speech-Dataset/, 2017.
[50] S. V. Kuyk, W. B. Kleijn, and R. C. Hendriks, "On the information rate of speech communication," in *ICASSP*, 2017, pp. 5625–5629.
[51] E. Law, K. West, M. I. Mandel, M. Bay, and J. S. Downie, "Evaluation of algorithms using games: the case of music tagging," in *ISMIR*, 2009, pp. 387–392.
[52] "Open speech and language resources," http://www.openslr.org/index.html, 2021.