GENERATIVE DE-QUANTIZATION FOR NEURAL SPEECH CODEC VIA LATENT DIFFUSION

Haici Yang[1], *Inseon Jang*[2], *and Minje Kim*[3] †

[1]Indiana University, Department of Intelligent Systems Engineering, Bloomington, IN, USA 47408
[2]Electronics and Telecommunications Research Institute, Daejeon, South Korea 34129
[3]University of Illinois at Urbana-Champaign, Department of Computer Science, IL, USA 61801

ABSTRACT

End-to-end speech coding models achieve high coding gains by learning compact yet expressive features and a powerful decoder in a single network. A challenging problem as such results in unwelcome complexity increase and inferior speech quality. In this paper, we propose to separate the representation learning and information reconstruction tasks. We leverage an end-to-end codec for learning low-dimensional discrete tokens. Instead of using its decoder, we employ a latent diffusion model to de-quantize coded features into a high-dimensional continuous space, relieving the decoder's burden of de-quantizing and upsampling. To mitigate the issue of over-smooth generation, we introduce midway-infilling with less noise reduction and stronger conditioning. We investigate the hyperparameters for midway-infilling and latent diffusion space with different dimensions in ablation studies. Subjective listening tests show that our model outperforms the state-of-the-art at two low bitrates, 1.5 and 3 kbps. We open-source the project for reproducibility [1].

Index Terms— Speech Codec, Latent Diffusion Model, Speech Synthesis

1. INTRODUCTION

Neural speech codecs are designed to capture intricate patterns from human speech more effectively than traditional methods. Recently, with the successful attempts of codec-based generation [1, 2, 3], high-bitrate neural codecs [4, 5, 6] gain much attention, for its capability of recovering high-fidelity audio.

Current high-fidelity codecs mostly perform waveform coding [7, 8]. They typically learn encoder and decoder models end-to-end, aiming at extracting expressive latent features and learning powerful decoders in a single network. Reconstruction objectives favor preserving essential information. Thus, with abundant data, end-to-end models are generally good at learning representations and achieving high-fidelity reconstruction at higher bitrates. For example, SoundStream achieves reasonable speech quality at 3kbps with fully convolutional architecture and residual vector quantization (RVQ) [4]. Pre-trained transformer and language models have been used to assist low-bitrate coding in open-sourced EnCodec [6] and an ultra-low-bitrate codec [9]. More recently, DAC [5] implements RVQGAN and snake activation into the architecture and improves coding quality, especially on high bitrates. Additionally, studies on disentangling inherited components of the input speech [10, 11, 12] further reduce redundancy and improve robustness. However, the above-mentioned works achieve quality reconstruction only at medium or high bitrate ($\geq$ 3 kbps). In the low-bitrate case, excessively complex networks are necessary to learn low-dimensional representation, impairing end-to-end training. Learning expressive features and powerful decodes at the same time remains a challenge.

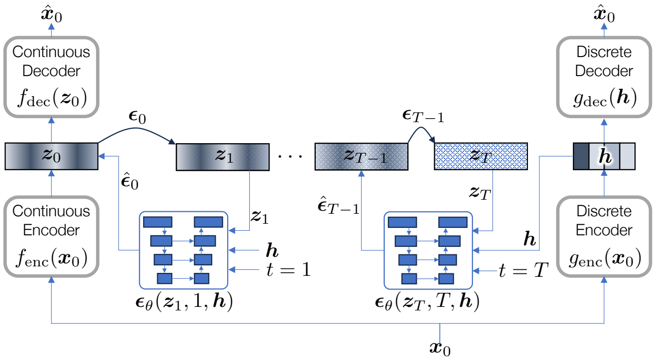

$\hat{\boldsymbol{x}}_0$
Continuous Decoder
$f_{\text{dec}}(\boldsymbol{z}_0)$
$\boldsymbol{\epsilon}_0$
$\boldsymbol{z}_0$
$\boldsymbol{z}_1$
· · ·
$\boldsymbol{z}_{T-1}$
$\boldsymbol{\epsilon}_{T-1}$
$\boldsymbol{z}_T$
$\hat{\boldsymbol{x}}_0$
Discrete Decoder
$g_{\text{dec}}(\boldsymbol{h})$
$\boldsymbol{h}$
$\hat{\boldsymbol{\epsilon}}_0$
Continuous Encoder
$f_{\text{enc}}(\boldsymbol{x}_0)$
$\boldsymbol{z}_1$
$\boldsymbol{h}$
$t = 1$
$\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_1, 1, \boldsymbol{h})$
$\hat{\boldsymbol{\epsilon}}_{T-1}$
$\boldsymbol{z}_T$
$\boldsymbol{h}$
$t = T$
$\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_T, T, \boldsymbol{h})$
Discrete Encoder
$g_{\text{enc}}(\boldsymbol{x}_0)$
$\boldsymbol{x}_0$

Fig. 1: A diffusion model de-quantizes the low-dimensional, low-bitrate discrete speech tokens (right-hand side) into high-dimensional continuous variables (left-hand side).

Therefore, many low-bitrate codecs utilize a vocoder to leverage generative models, which resynthesize speech from existing features, e.g., WaveNet-based codecs [13, 14], LPCNet for real-time coding [15], GAN-empowered resynthesis [16]. Recent employment of AudioLM [1] in SoundStream as in LMCodec [17] reduces the bitrate down to ∼1kbps. Neural feature predictor [18] applies a generative model in the feature domain to assist LPCNet with more efficient feature input.

We observed from pilot experiments that low-bitrate codes from waveform codecs can preserve distinguishable speech features and better capture essential information than the traditional speech features at the same bitrate. Thus, it is worth utilizing the representation learned from the end-to-end models. Meanwhile, the decoder can be replaced with a more powerful generative model. To this end, the proposed system consists of three modules. 1) An end-to-end codec whose encoder performs dimension reduction and quantization at its bottleneck; 2) A separate auto-encoder that defines a continuous and high-dimensional latent space, and is responsible for high-fidelity reconstruction; 3) Finally, a latent diffusion model bridges the gap between the two feature representations. By conditioning the dif-

*This work was supported by Electronics and Telecommunications Research Institute (ETRI) grant funded by the Korean government (23ZH1200; "The research of the basic media contents technologies").

†Work done at Indiana University.

[1] https://minjekim.com/research-projects/LaDiffCodec

H. Yang, I. Jang and M. Kim, "Generative De-Quantization for Neural Speech Codec Via Latent Diffusion," IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Seoul, Korea, Republic of, 2024, pp. 1251-1255, doi: 10.1109/ICASSP48485.2024.10446556.

fusion model with the lower-dimensional quantized code from the end-to-end codec, we expect this model to perform generative dequantization and upsampling.

We opt for the diffusion model among other generative models because 1) diffusion models have shown potential in generating natural-sounding speech and audio [19] from an extensive range of conditions; 2) Diffusion models provide a way to generate the entire feature map altogether and refine it iteratively. In contrast to the autoregressive models, diffusion models look at a larger condition space, which significantly increases the quality upper bound. Specifically, we found that the latent diffusion model [20] surpasses a regular time-domain diffusion model under this setup, possibly because running on the latent space offloads the model's burden to reconstruct raw waveforms. We explore different choices over the diffusion space's dimension in an ablation study. As diffusion models are prone to generate over-smooth speech and hallucinate content, we introduce a new sampling technique that adds stronger prior to the conditional generation. We notice a concurrent work that explores high-fidelity audio generation from the speech codec by multi-band diffusion [21]. In comparison, our model focuses on speech generation and achieves better quality at both low- and high-bitrate cases with only one *latent* diffusion model, thus more efficient with respect to the model design. We call our model the latent diffusion codec (LaDiffCodec).

2. GENERATIVE DE-QUANTIZATION WITH LATENT DIFFUSION

Fig. 1 provides an overview of the proposed LaDiffCodec. The characteristic latent diffusion process, depicted in the middle, converts quantized codes into continuous representation.

2.1. Latent Diffusion

Diffusion models are generative models characterized by two Markov processes: diffusion and reverse processes. The diffusion process, $q(\boldsymbol{x}_{1:T}|\boldsymbol{x}_0) = \prod_{t=1}^{T} q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$, corrupts clean data point $\boldsymbol{x}_0$ by gradually adding Gaussian noise until it reaches a random variable $\boldsymbol{x}_T$ close to the standard normal distribution. Hence, $q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) \sim \mathcal{N}(\sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t I)$, where β_t is the pre-defined noise schedule ($0 < \beta_0 < ... < \beta_T < 1$). With reparameterization, sampling the variable of step t from the diffusion process can be acquired by, $\mathcal{F}(\boldsymbol{x}_0, t) = \sqrt{\bar{\alpha}_t}\boldsymbol{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon} \sim \mathcal{N}(0,1)$ and $\bar{\alpha}_t := \prod_{i=0}^{t}(1-\beta_i)$.

The reverse process $p_\theta(\boldsymbol{x}_{0:T}) = p(\boldsymbol{x}_T)\prod_{t=1}^{T} p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t)$ is commonly represented by a parametric function, e.g., a neural network. This process can be conditioned by various types of auxiliary information.

While diffusion models succeed in many data generation tasks, the high computational complexity limits their accessibility. Meanwhile, the models are prone to spend excessive amounts of resources on modeling imperceptible details, especially on the multi-modality tasks [20]. As a solution, latent diffusion models propose to operate in the latent space $\boldsymbol{z}$ that can be learned by a pre-trained autoencoder,

$$\hat{\boldsymbol{x}}_0 \leftarrow f_{\text{dec}}(\boldsymbol{z}_0), \quad \boldsymbol{z}_0 \leftarrow f_{\text{enc}}(\boldsymbol{x}_0), \tag{1}$$

assuming this space to be computationally preferable and perceptually equivalent to the data domain. Accordingly, the diffusion process defines the distribution of latent variables, $q(\boldsymbol{z}_{1:T}|\boldsymbol{z}_0)$, rather than the raw data $\boldsymbol{x}$.

2.2. Diffusion-Based De-Quantization

LaDiffCodec maps the two latent spaces, the low-dimensional discrete code $\mathbb{H}$ and the high-dimensional continuous feature $\mathbb{Z}$. The restorative nature of this mapping requires conditional generation. This section describes the three components of LaDiffCodec: discrete coding, continuous coding, and conditional diffusion sampling.

Discrete coding: Our discrete coding module $g(\cdot)$ is an autoencoder-type codec that learns the discrete code space $\mathbb{H}$ using its encoder component, $g_{\text{enc}} : \mathbb{X}^N \to \mathbb{H}^D$. It provides the discretized features $\boldsymbol{h} \in \mathbb{H}^D$, which serve as the transmission bitstream in the ordinary codec usage. Ideally, the quantized speech tokens $\boldsymbol{h}$ should contain sufficient information for faithful speech reconstruction. We opt for autoencoder codecs as we believe that end-to-end training learns more effective codes. Once trained, we discard the decoder module, $g_{\text{dec}} : \mathbb{H}^D \to \mathbb{X}^N$, which is the performance bottleneck when $\mathbb{H}^D$ is low dimensional and discrete. Instead, LaDiffCodec repurposes this discrete code $\boldsymbol{h}$ to condition the reverse diffusion sampling process, which redefines the decoding process in a generative fashion. We employ EnCodec [6] as the backbone of this discrete coding module.

Continuous coding: To de-quantize the discrete token $\boldsymbol{h}$ into a continuous feature vector $\boldsymbol{z}$, LaDiffCodec utilizes a latent diffusion model defined in the continuous space $\mathbb{Z}$. To this end, we pretrain another EnCodec-like continuous autoencoder, whose encoder maps the raw signal space $\mathbb{X}$ into the continuous feature space $\mathbb{Z}$, i.e., $f_{\text{enc}} : \mathbb{X}^N \to \mathbb{Z}^L$, followed by a decoder that maps it back to the signal domain: $f_{\text{dec}} : \mathbb{Z}^L \to \mathbb{X}^N$. A trade-off exists between different sizes of the continuous latent space. While a large latent dimension L enables high expressiveness, it also causes longer sampling time and lower efficiency. Moreover, the gap between the high-dimensional continuous space and a low-dimensional discrete space has to be filled with upsampling layers, which brings additional artifacts. In our experiments, we investigate a few options of L to minimize the trade-off.

Conditional Latent Diffusion: A diffusion model built on $\mathbb{Z}$ gradually adds noise $\boldsymbol{\epsilon}_t$ to $\boldsymbol{z}_t$ in the diffusion process. During training, a conditional neural network model estimates the posterior distribution along the de-noising (reverse) path $p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$. We use the $\boldsymbol{\epsilon}$-prediction parameterization [22],

$$\mathcal{L} = \mathbb{E}_{\boldsymbol{z}_0,t,\boldsymbol{h}}(||\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t, t, \boldsymbol{h})||) \tag{2}$$

where $\boldsymbol{\epsilon}_\theta(\boldsymbol{z}_t, t, \boldsymbol{h})$ is the output of the neural network with weights θ. It predicts the noise to be removed in the sampling process, resembling the gradient of data density. The quantized tokens $\boldsymbol{h}$ condition both training and sampling stages to steer the generation.

2.3. Midway-Infilling

The original sampling algorithm proposed in denoising diffusion probabilistic models (DDPM) [22] iteratively removes predicted noise $\boldsymbol{\epsilon}_\theta$ from the noisy data samples, $\mathcal{G}\colon \boldsymbol{x}_t \mapsto \boldsymbol{x}_{t-1}$,

$$\mathcal{G}(\boldsymbol{x}_t, t, \boldsymbol{h}) = \frac{1}{\sqrt{1-\beta_t}}\left(\boldsymbol{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\theta\big(\boldsymbol{x}_t, \sqrt{\bar{\alpha}_t}, \boldsymbol{h}\big)\right) + \sqrt{\beta_t}\boldsymbol{n},$$

from $\boldsymbol{x}_T$ to $\boldsymbol{x}_0$, where $\boldsymbol{n}$ is a Gaussian noise. T usually equals the number of time steps used for training, e.g., 1,000. DDPM sampling can be tedious, given the large number of sampling steps. Denoising diffusion implicit models' (DDIM) sampling method [23] significantly reduces the sampling steps. However, in our task, both DDPM and DDIM are prone to generate overly smooth samples at very low bitrates, i.e., 1 and 1.5 kbps, with some hallucination effects, such as

Algorithm 1: Midway-Infilling

```
Input: Conditioning vector $\boldsymbol{h}$, midway step $\tau$, interpolation
       ratio $\gamma$, sampling function $\mathcal{G}(\cdot)$
$\boldsymbol{s}_\tau \leftarrow \boldsymbol{h}, \quad \boldsymbol{x}_\tau \sim \mathcal{N}(0,1), \quad \boldsymbol{x}_\tau = (1-\gamma)\boldsymbol{x}_\tau + \gamma\boldsymbol{s}_\tau$
for $t = \tau \dots 1$ do
    $\boldsymbol{s}_{t-1} = \mathcal{G}(\boldsymbol{s}_t, t, \boldsymbol{h})$ – Infilling branch
    $\boldsymbol{x}_{t-1} = \mathcal{G}(\boldsymbol{x}_t, t, \boldsymbol{h})$ – Sampling branch
    $\boldsymbol{x}_{t-1} = (1-\gamma)\boldsymbol{x}_{t-1} + \gamma\boldsymbol{s}_{t-1}$
end
return $\boldsymbol{x}_0$
```

missing or replaced phonemes.

We believe the smoothing effect is caused by excessive noise reduction and insufficient assistance from the condition. Therefore, we propose to use another sampling technique, *midway-infilling*. It improves the sampling quality and efficiency in two folds. 1) It starts sampling from a mid-point step $\tau < T$ rather than from $\boldsymbol{x}_T$, a random noise space, thus reducing sampling steps by 10 to 20 times without sacrificing the sampling quality; 2) it implements a separate conditioning branch to impose stronger conditioning during sampling.

Midway-infilling is inspired by the infilling algorithm proposed in [24]. Infilling aims to condition and steer the sampling steps on unconditional diffusion models. In the original infilling process, an occluded sample $\boldsymbol{s}_0$ is provided. The diffusion process runs on $\boldsymbol{s}_0$ (infilling branch) to meet the time step of reversed sampling branch $\boldsymbol{x}_t$ (sampling branch), i.e., $\boldsymbol{s}_t = \mathcal{F}(\boldsymbol{s}_0, t)$. $\boldsymbol{x}_t$ and $\boldsymbol{s}_t$ are then interpolated with a certain ratio as the final sampling output for each step. Akin to infilling, the proposed midway-infilling involves two branches. The difference is, instead of providing $\boldsymbol{s}_0$, we treat the condition feature $\boldsymbol{h}$ or its upsampled version as $\boldsymbol{s}_\tau$, a midway variable of the infilling branch's Markov chain path. Accordingly, the infilling branch runs a parallel reverse process from step τ to 0 with the sampling branch. The interpolation happens after each reverse step, as described in Algorithm 1.

3. EXPERIMENT SETUP

3.1. Model Design and Hyperparameter Setup

In the forward process, we use $T = 1000$ steps and set noise schedule linearly from $\beta_1 = 0.0001$ to $\beta_T = 0.02$. A U-Net-based model parameterizes the reverse diffusion process, similar to [20, 25]. The U-Net is built with ResNet blocks, each containing three convolution layers and one four-head self-attention layer. The model comprises five encoder blocks, one middle block, and five decoder blocks. The channel dimensions of encoder blocks are $[128, 256, 256, 512, 512]$. Decoder blocks have the reversed order of dimensions as encoder blocks, and the channel dimension of the middle block is 512. While AudioLDM [25] uses FiLM conditioning, we condition the diffusion model by the stacked discrete tokens $\boldsymbol{h}$ and model input $\boldsymbol{z}_t$, as we found it more effective than FiLM in this task. When the dimensionality of the continuous and discrete code spaces do not match, i.e., $D < L$, $\boldsymbol{h}$ is firstly upsampled with transposed convolutional layers. We scale each frame of the upsampled tokens to $[-1, 1]$ before conditioning them on the diffusion.

We use the proposed midway-infilling method for sampling, where the hyperparameters are shared in all bitrate cases to keep the usage simple. We set the midway step $\tau = 100$ and $\gamma = 0.3$. Once latent diffusion sampling is finished, the continuous decoder takes in the obtained sample $\boldsymbol{z}_0$ and maps it to the time-domain signal. With a reduced sampling step (100), it takes ~5.65 seconds to generate a 3.2-second sample.

We retrain the non-streamable version of EnCodec with the 16 kHz Librispeech dataset [26] as the discrete autoencoder. Its encoder and decoder use SEANet [27] as the backbone. The encoder downsamples input through four convolution layers with strides of size 2, 4, 5, and 8, respectively. Using transposed convolution, the decoder upsamples the latent space in the reverse order. In addition, we build a continuous autoencoder akin to EnCodec. We keep only one downsampling layer with a stride size of 8 to achieve higher dimension and, consequently, expressiveness.

3.2. Data and training

All experiments are conducted on the Libirspeech dataset, `train-clean-100` fold for training, and `dev-clean` for testing. Model training runs on the 3.2s sequences. The three components, i.e., discrete autoencoder (16khz EnCodec), continuous autoencoder, and the latent diffusion model, are trained separately. When training the latent diffusion model, both autoencoders are frozen. Our diffusion models are bitrate-specific. For example, LaDiffCodec at 1.5 kbps uses EnCodec's 1.5 kbps tokens as its condition. We use Adam optimizer for all the training tasks, with a batch size of 20 and a learning rate 5×10^{-5}. It takes six hours to train the autoencoders and three days for the latent diffusion model on one NVIDIA A100 GPU.

3.3. Evaluation and Ablation study

We run a MUSHRA-like subjective test [28] to compare LaDiffCodec with 16 kHz EnCodec at two bitrates, 1.5kbps and 3kbps. Sequences from a 16 kHz DAC at 3kbps are also included for comparison. 13 audio experts participated in and rated ten gender-balanced samples of 3 seconds.

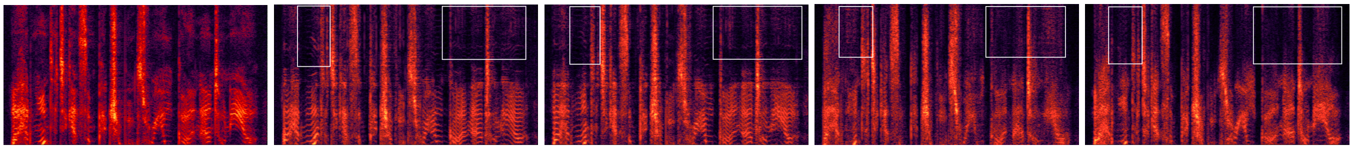
All the ablation studies are evaluated with PESQ [29] on randomly picked 50 samples from the test set. We notice that PESQ does not reflect the real perceptual preference among different coding systems. However, when comparing sequences generated from systems of the same kind, they tend to exhibit a steady trend.

4. EXPERIMENTAL RESULTS

4.1. Comparison with other codec

Figure 3 shows MUSHRA scores of different codec systems. We see LaDiffCodec surpasses EnCodec and DAC at both bitrates. Particularly, LaDiffCodec's 1.5 kbps samples are preferred by the subjects to the 3 kbps samples of the other codecs.

We believe that LaDiffCodec has two main traits that contribute to its superior performance. Firstly, it recovers coding artifacts. Lossy compression can cause various speech alterations and degradation. We observe that at 1.5kbps, EnCodec starts to lose intelligibility because some phonemes are not recovered precisely. DAC exhibits a similar artifact at 1.5 kbps (using three quantizer cookbooks). At 3kbps, while the intelligibility is better preserved, the baseline codecs still produce artifacts such as subtle background noise (DAC) or metallic and hissing sound (EnCodec). LaDiffCodec can fix severe quantization artifacts by generating variables in the continuous latent space. The fact that the latent diffusion process works in the pre-trained latent space narrows its synthesis process to a more straightforward problem. In addition, the well-trained continuous latent space leads to higher speech reconstruction quality, eliminating the non-speech artifacts and distortion.



(a) Reference (b) EnCodec@1.5kbps (c) EnCodec@3kbps (d) LaDiffCodec@1.5kbps (e) LaDiffCodec@3kbps

Fig. 2: Spectrograms of the reference speech and its coded versions by EnCodec and LaDiffCodec at 1.5kbps and 3kbps. The audio samples of these spectra are available on the sample page. White blocks point out the example areas where LaDiffCodec eludes aliasing artifacts.

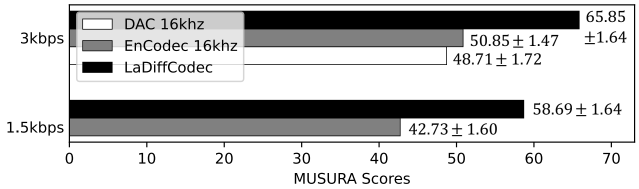
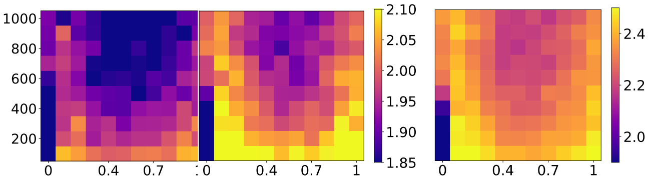


Fig. 3: Average MUSHRA scores and their confidence intervals of LaDiffCodec and baseline codec systems at 1.5kbps and 3kbps.



(a) 1kbps (b) 1.5kbps (c) 3kbps

Fig. 4: PESQ scores of LaDiffCodec using different midway-infilling hyperparameters. The X-axis denotes the mask ratios γ from 0 to 1. The Y-axis denotes the midway timestep τ. Values lower than the color bar's lower bound are clipped.

Secondly, it generates more natural-sounding speech. Fig. 2 presents spectrograms of the reference and various decoded versions. These low-bitrate EnCodec results suffer from losing high-frequency information due to the reduced expression space. Waveform coding, in particular, often experiences aliasing artifacts, as shown in 2b and 2c. In these spectrograms, the high-frequency area over $\sim$4 kHz shows a mirrored reflection of the lower frequency harmonics. These high-frequency aliasing effects can add unnatural artifacts to the reconstruction. In contrast, LaDiffCodec makes up some high-frequency energy and eludes the aliasing effect. As a result, it produces a more natural and pleasant sound. As we use DAC's public 16khz checkpoint, which is not re-trained on the Librispeech dataset, its performance appears to be suboptimal compared to the re-trained EnCodec in our experiments.

4.2. Ablation Studies

Hyperparameters of Midway-Infilling: This ablation explores different settings of midway-infilling hyperparameters. Smaller γ values correlate to less contribution from the condition branch $[\boldsymbol{s}_\tau, ..., \boldsymbol{s}_0]$ during sampling, while a large τ means the sampling process conducts more noise reduction. When $\gamma = 0$ and $\tau = 1000$, it is equivalent to DDPM's original sampling method. With the correct set of hyperparameters, midway-infilling gains higher PESQ than the original DDPM sampling. The leftmost columns of each graph present the degrading performance by reducing DDPM sampling steps, with no extra infilling branch involved (i.e., $\gamma = 0$). The rightmost columns, on the other hand, present the infilling branch's sole contribution to sampling. According to the PESQ score, the best quality is obtained when sampling step τ is small and γ is close to 0 or 1. Our perceptual rating aligns with PESQ concerning τ. However, a large interpolation ratio γ leads to a better phoneme-level reconstruction at the cost of less naturalness. A sequence of speech samples regarding different interpolation ratios can be found on our webpage.

Strides	@1kbps	@1.5kbps	@3kbps
[1]	1.18 ± 0.04	1.20 ± 0.04	1.77 ± 0.19
[8]	**1.81** ± **0.15**	1.95 ± 0.15	**2.23** ± **0.17**
[4, 8]	1.71 ± 0.71	**2.19** ± **0.75**	2.16 ± 0.69
[4, 5, 8]	1.66 ± 0.11	1.71 ± 0.12	1.84 ± 0.10
[2, 4, 5, 8]	1.49 ± 0.09	1.65 ± 0.13	1.71 ± 0.12

Table 1: Performance of diffusion models with different latent space dimensions. The arrays in the first column present the stride sizes of each down-sampling layer in the continuous encoder.

Latent Dimensionality: Table 1 presents the PESQ scores of LaDiffCodec with different latent space dimensions. The total downsampling rate is the product of stride sizes. The first row shows results from the ordinary (non-latent) diffusion model, which samples in the time domain with no continuous autoencoder involved. All the experiments are made to run for the same amount of time. In comparison, the LD models outperform the time-domain diffusion method (stride= 1), indicating that a reduced dimension and auxiliary feature learning can facilitate diffusion modeling, especially in this speech coding task. However, reduced dimensions do not always lead to superior performance. When more downsampling layers are added to the continuous autoencoder, the latent space starts losing its expression power or becomes hard to model with the diffusion process.

5. CONCLUSION

This work proposed LaDiffCodec and demonstrated the effectiveness of integrating waveform coding-based feature learning and latent diffusion model for high-quality, low-bitrate speech coding. By mapping the low-dimensional discrete speech token into high-dimensional continuous space using latent diffusion, the codec relieved the burden of upsampling and de-quantization from the decoder. It improved speech quality with reduced artifact and increased naturalness. While mainly focusing on the low-bitrate scenarios, our work potentially sheds light on the high-fidelity codec-based generation. Our models provides a solution that enables using fewer codebooks for categorical generation, reducing the task's difficulties without sacrificing output sound quality.

6. REFERENCES

[1] Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, D. Roblek, O. Teboul, Da. Grangier, M. Tagliasacchi, et al., “AudioLM: a language modeling approach to audio generation,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 31, pp. 2523–2533, 2023.

[2] C. Wang, S. Chen, Y. Wu, Z. Zhang, L. Zhou, S. Liu, Z. Chen, Y. Liu, H. Wang, J. Li, et al., “Neural codec language models are zero-shot text to speech synthesizers,” *arXiv preprint arXiv:2301.02111*, 2023.

[3] F. Kreuk, G. Synnaeve, A. Polyak, U. Singer, A. Défossez, J. Copet, D. Parikh, Y. Taigman, and Y. Adi, “Audiogen: Textually guided audio generation,” *arXiv preprint arXiv:2209.15352*, 2022.

[4] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, “Soundstream: An end-to-end neural audio codec,” *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 30, pp. 495–507, jan 2022.

[5] R. Kumar, P. Seetharaman, A. Luebs, I. Kumar, and K. Kumar, “High-fidelity audio compression with improved RVQGAN,” *arXiv preprint arXiv:2306.06546*, 2023.

[6] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, “High fidelity neural audio compression,” *arXiv preprint arXiv:2210.13438*, 2022.

[7] 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.

[8] D. Petermann, S. Beack, and M. Kim, “HARP-Net: Hyper-autoencoded reconstruction propagation for scalable neural audio coding,” in *Proc. of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2021.

[9] A. Siahkoohi, M. Chinen, T. Denton, W. Kleijn, and J. Skoglund, “Ultra-low-bitrate speech coding with pretrained transformers,” in *Proc. Interspeech*, 2022.

[10] H. Yang, K. Zhen, S. Beack, and M. Kim, “Source-aware neural speech coding for noisy speech compression,” in *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2021.

[11] X. Jiang, X. Peng, Y. Zhang, and Y. Lu, “Disentangled feature learning for real-time neural speech coding,” in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[12] A. Omran, N. Zeghidour, Z. Borsos, F. de Chaumont Quitry, M. Slaney, and M. Tagliasacchi, “Disentangling speech from surroundings with neural embeddings,” in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[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 *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2018, pp. 676–680.

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

[15] J.-M. Valin and J. Skoglund, “A real-time wideband neural vocoder at 1.6 kb/s using LPCNet,” in *Proc. Interspeech*, 2019.

[16] A. Polyak, Y. Adi, J. Copet, E. Kharitonov, K. Lakhotia, W.-N. Hsu, A. Mohamed, and E. Dupoux, “Speech resynthesis from discrete disentangled self-supervised representations,” *arXiv preprint arXiv:2104.00355*, 2021.

[17] T. Jenrungrot, M. Chinen, W. B. Kleijn, J. Skoglund, Z. Borsos, N. Zeghidour, and M. Tagliasacchi, “LMCodec: A low bitrate speech codec with causal transformer models,” in *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2023.

[18] H. Yang, W. Lim, and M. Kim, “Neural feature predictor and discriminative residual coding for low-bitrate speech coding,” in *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2023.

[19] S. Pascual, C. Bhattacharya, G.and Yeh, J. Pons, and J. Serrà, “Full-band general audio synthesis with score-based diffusion,” in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[20] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10684–10695.

[21] R. S. Roman, Y. Adi, A. Deleforge, R. Serizel, G. Synnaeve, and A. Défossez, “From discrete tokens to high-fidelity audio using multi-band diffusion,” 2023.

[22] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[23] J. Song, C. Meng, and S. Ermon, “Denoising diffusion implicit models,” in *International Conference on Learning Representations*, 2021.

[24] G. Mittal, J. Engel, C. Hawthorne, and I. Simon, “Symbolic music generation with diffusion models,” *arXiv preprint arXiv:2103.16091*, 2021.

[25] H. Liu, Z. Chen, Y. Yuan, X. Mei, X. Liu, D. Mandic, W. Wang, and M. D. Plumbley, “Audioldm: Text-to-audio generation with latent diffusion models,” *arXiv preprint arXiv:2301.12503*, 2023.

[26] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “Librispeech: An ASR corpus based on public domain audio books,” in *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015, pp. 5206–5210.

[27] M. Tagliasacchi, Y. Li, K. Misiunas, and D. Roblek, “Speech enhancement by online non-negative spectrogram decomposition in non-stationary noise environments,” in *Proc. Interspeech*, 2012.

[28] ITU-R Recommendation BS 1534-3, “Method for the subjective assessment of intermediate quality levels of coding systems (MUSHRA),” 2015.

[29] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, “Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs,” in *Proc. of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2001, vol. 2, pp. 749–752.