LRAC SYSTEM DESCRIPTION FOR TRACK1 AND TRACK2

Ziqian Wu *JiaWei Jiang* *Kunpeng Lin* *He Wang* *Qingbo Huang*

ByteDance

ABSTRACT

This paper describes our team's submission to the 2025 Low-Resource Audio Codec (LRAC) Challenge, covering the models for both track1 and track2—with the same model architecture used for both tracks. Key details presented include the model structure, loss function design, hyperparameter settings, computational complexity, and latency. These details reflect our approach to meeting the low-resource requirements of the challenge, providing transparency for our codec design.

Index Terms— Neural audio codec, residual vector quantization, audio enhancement

1. INTRODUCTION

Low-resource audio codecs are critical for applications such as edge devices or low-bandwidth networks, where limited computing power and storage require efficient compression without sacrificing audio quality. The 2025 Low-Resource Audio Codec (LRAC) Challenge was launched to advance such technologies, setting clear goals to balance perceptual quality, compression ratio, and resource efficiency across two tracks.

Our team participated in this challenge, aiming to design a codec that meets the low-resource criteria while performing well on both tracks. A key choice in our design is that we used the same model architecture for track1 and track2—this simplifies development while ensuring consistent performance principles.

In the following sections, we will detail our model's structure, loss function, hyperparameter settings, computational complexity, and latency. These details explain how our codec addresses the LRAC Challenge's requirements and provide a basis for understanding its performance.

2. DATA PROCESSING

For track1 and track2 of the challenge, we adopted an identical data selection strategy. Specifically, we utilized the official dataset selection script provided by the challenge organizers to filter and process the data. Through this standardized script, a total of 340k audio sequences were selected, corresponding to more than 700 hours of speech data. Additionally, to ensure compliance with the challenge's requirements, the noise and reverberation data used for data augmentation were strictly sourced from the datasets specified in the challenge guidelines.

Before training, the data undergo preprocessing as follows:

1. **Pitch modification**: 10% speech signals are applied randomly with pitch shift in the range of -2 to 12 semitones.
2. **Duration normalization**: All speech segments are standardized to 8 seconds. Segments longer than 8 seconds are truncated, while those shorter than 8 seconds are repeated to reach the target length.
3. **Speech type configuration**: The preprocessed data consists of four types with specific proportions: clean speech, noisy speech, reverberant speech, and multi-speaker speech.
    - For noisy speech, the signal-to-noise ratio (SNR) is randomly set between **-5 and 10.0 in track1, -20 and 20.0 in track2**. After adding noise, there is a 40% probability of further applying reverberation.
    - Reverberant speech is generated directly using the challenge-specified reverberation dataset. After adding reverberation, there is a 40% probability of further adding noise.
    - For simultaneous talkers, the amplitude of one speaker's voice is randomly scaled to 0.3 to 1.0 times its original value, then directly summed with the voice of the other speaker.

The proportions of signal types are as follows in Table 1:

	Clean	**Noisy**	**Reverb**	**Simultaneous Talkers**
Track1	8	5	5	2
Track2	4	4	1	0

Table 1. Proportions of signal type weights in different tracks

For track 1, the goal is transparent audio transmission, so its training target is input audio to input audio.For track 2, the goal is noise reduction and dereverberation, so its training target is input audio to the denoised and dereverberated audio.

3. MODEL STRUCTURE

The model is composed of three core components: an encoder, a quantizer and a decoder, which processes an input audio sequence in the time domain with shape $[1, T]$ and produces an output sequence with the same shape.

The encoder begins with a Conv1D layer with a kernel size $k = 7$. Next, it incorporates 4 repeated modules, with stride = 3, 4, 5, 8. In each module, 3 residual units with dilation = 1, 3, 9 and SnakeBeta activation are applied. Finally, a GRU layer is used to leverage inter-frame correlations between features. The SnakeBeta is defined as follows in Equation 1:

$$\text{SnakeBeta}(x) = x + \frac{1}{\beta}\sin^2(\alpha x) \tag{1}$$

The quantizer adopts Residual Vector Quantization: 12 codebooks are used at a bitrate of 6 kbps, while 2 codebooks are employed at 1 kbps. Additionally, each layer of the codebooks has a size of 1024, and each codebook has a dimension of 8.

The Decoder starts with a Conv1D layer to project the quantized features into a suitable dimension for subsequent processing. 8 Conv2FormerBlocks[1] are stacked to transform and reconstruct the features, leveraging the strengths of Conv2Former in modeling both local and global feature dependencies. A final Conv1D layer further refines the feature map, preparing it for time-frequency conversion. Ultimately, an ISTFT (Inverse Short-Time Fourier Transform) layer converts the processed features back into the time domain. Model struct is showed in Figure 1.

The model takes 20ms audio data as input. The latency will be introduced in section 7.

Both tracks used the same model struture with different model size, the main different params of both model are listed in Table 2.

Parameter	Track 1	Track 2
encoder_dim	12	32
encoder_group	4	8
encoder_output_latent_dim	256	512
conv2formerblock_input_dim	372	512
conv2formerblock_hidden_dim	380	620

Table 2. Comparison of model parameters between Track 1 and Track 2

4. DISCROMINATORS AND LOSS FUNCTIONS

4.1. Discriminators

We used a variety of discriminators, including the Multi-period Discriminator[2], Multi-res STFT Discriminator[2], Multi-res Subband STFT Discriminator, and Multi-seq length Mel-spectrogram Discriminator[3]. All these discriminators are updated at every training step. Parameters of these discriminators are showed in Table 3.

Discriminator Type	Params	Values
Multi-period Discriminator	periods	2, 3
Multi-res STFT Discriminator	fft_sizes	64, 128, 256, 512, 1024, 2048
	window_lengths	64, 128, 256, 512, 1024, 2048
	hop_factor	0.25
Multi-res Subband STFT Discriminator	fft_sizes	2048, 1536, 1024, 768, 512
	window_lengths	2048, 1536, 1024, 768, 512
	hop_factor	0.25
Multi-seq Length Mel-spec Discriminator	n_mel	80
	fft_size	1024
	fft_window_length	1024
	hop_length	512
	seq_length	64, 128, 256

Table 3. Parameters of Different Discriminators

4.2. Loss Functions

We employed a range of loss functions in our framework, including multiscale mel loss, multiscale STFT loss, discriminator feature loss, generator loss, RVQ commitment loss, RVQ codebook loss, PESQ[4] loss, and modified multiscale STFT loss[5]. These losses collectively contribute to optimizing the model's performance by addressing different aspects of audio generation quality, feature alignment, and perceptual consistency. The total loss functions are defined as follows in Equation 2:

$$\begin{aligned}\text{Loss} = {} & \lambda_1 \text{Loss}_{\text{mel}} + \lambda_2 \text{Loss}_{\text{stft}} \\ & + \lambda_3 \text{Loss}_{\text{disc}} + \lambda_4 \text{Loss}_{\text{gen}} \\ & + \lambda_5 \text{Loss}_{\text{vqcommit}} + \lambda_6 \text{Loss}_{\text{vqcodebook}} \\ & + \lambda_7 \text{Loss}_{\text{pesq}} + \lambda_8 \text{Loss}_{\text{modified\_stft}}\end{aligned} \tag{2}$$

5. TRAINING PROCESS

During the model training, we adopted a two-stage training process. In the second stage, we significantly reduced the

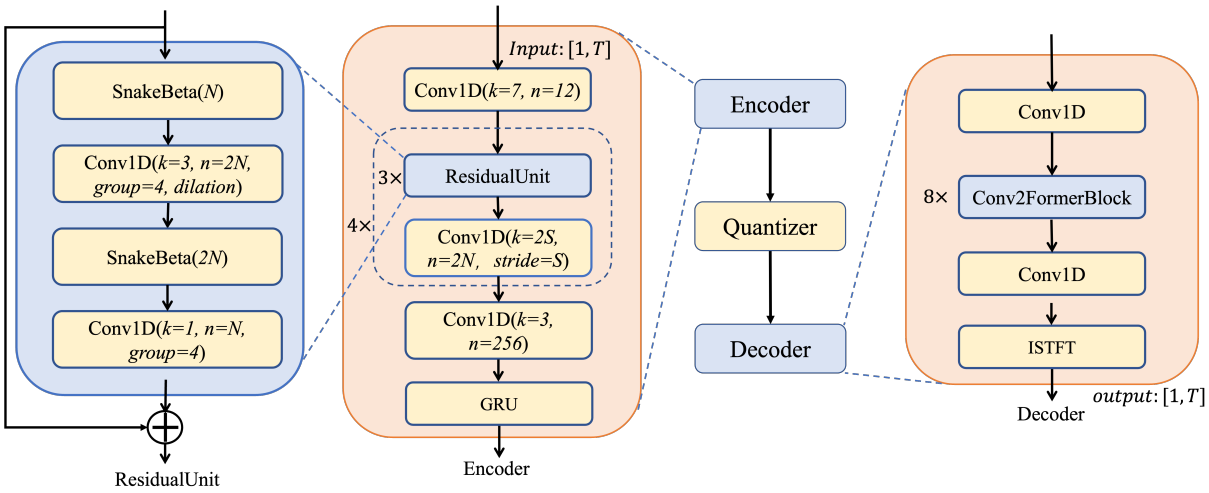

Fig. 1. Schematic diagram of the model architecture

weight of the mel loss, which facilitates the generation of clear harmonics in the audio. **We only applied the two-stage training to the model for Track 1, while the model for Track 2 only underwent one-stage training.**

Training parameters are defined in Table 4.

Param	Stage 1	Stage 2
Batch size	16	16
Training steps	800000	200000
LR	0.0001	0.0001
LR decay (Exp)	0.999996	0.999996

Table 4. Training parameters (two stages)

During the training of the model, half of the training iterations bypass quantization entirely. For the remaining half quantization-enabled training, the codebook dropout method is adopted to support training for multiple bitrates.

Loss functions weights for different training steps are listed in Table 5.

Loss lambda	Stage 1	Stage 2
λ_1	15.0	1.0
λ_2	10.0	10.0
λ_3	2.0	2.0
λ_4	1.0	1.0
λ_5	0.25	0.25
λ_6	1.0	1.0
λ_7	5.0	5.0
λ_8	10.0	10.0

Table 5. Loss function weights (λ) for different training stages

6. PARAMETER COUNTS AND COMPUTATIONAL COMPLEXITY

We statistically analyzed the computational complexity and parameter counts of the two models. The computational complexity includes Short-Time Fourier Transform (STFT) operations and codebook distance computation. The parameter counts and computational complexity are listed in Table 6.

Metric & Module	Unit	Track 1	Track 2
Model Complexity			
Encoder	mmacs	192.75	937.69
Quantizer	mmacs	7.73	9.83
Decoder	mmacs	147.01	297.13
Parameter Count			
Encoder	M	0.973	5.145
Quantizer	M	0.154	0.209
Decoder	M	2.954	5.967

Table 6. Comparison of Model Complexity (mmacs) and Parameter Count (M) between Track 1 and Track 2

7. SYSTEM LATENCY

The encoder accepts 20-ms audio frames as input. The decoder outputs 40-ms audio, consisting of 10 ms of prior audio, 20 ms of current audio, and 10 ms of subsequent audio. For seamless output, the 20 ms of current audio needs to be overlapped and added with the 10 ms of subsequent audio, resulting in a decoder latency of 10 ms. The total latency is the sum of the encoder frame size (20 ms) and decoder latency (10 ms), totaling 30 ms. The schematic diagram of system latency is shown in Figure 2.

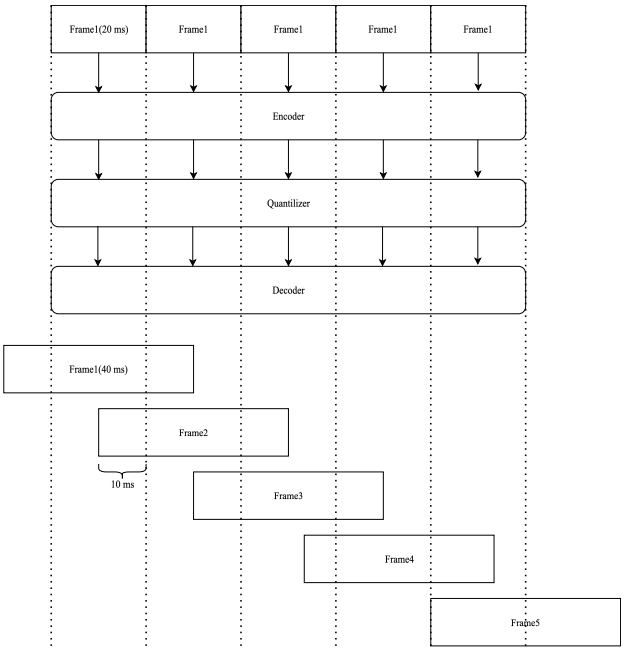

Fig. 2. System latency description

8. REFERENCES

[1] Qibin Hou, Cheng-Ze Lu, Ming-Ming Cheng, and Jiashi Feng, "Conv2former: A simple transformer-style convnet for visual recognition," 2022.

[2] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," 2020.

[3] Jiawei Chen, Xu Tan, Jian Luan, Tao Qin, and Tie-Yan Liu, "Hifisinger: Towards high-fidelity neural singing voice synthesis," 2020.

[4] 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 *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, 2001, vol. 2, pp. 749–752 vol.2.

[5] Tianze Luo, Xingchen Miao, and Wenbo Duan, "Wavefm: A high-fidelity and efficient vocoder based on flow matching," 2025.