LOW RESOURCE AUDIO CODEC CHALLENGE TRACK1: TRANSPARENCY CODEC

Zixiang Wan, Guochang Zhang, Haoran Zhao, Runqiang Han, Jianqiang Wei

Anker Innovations, Beijing, China

ABSTRACT

We propose a frequency–time domain fusion audio codec for the 2025 Low-Resource Audio Coding (LRAC) Challenge, designed to meet strict constraints on complexity, latency, and bitrate while ensuring high quality and robustness. The system achieves 698 M FLOPs, 1.48 M parameters, and sub-30 ms latency, combining a frequency-domain encoder, Residual Vector Quantization (RVQ), and a time-domain decoder. Multi-Period and Multi-Resolution GANs jointly refine temporal and spectral fidelity. A multi-stage training process combines spectral reconstruction with adversarial objectives and noise-reduction strategies to ensure stable optimization and high-quality output. Evaluations at 1 kbps and 6 kbps in clean, noisy, and reverberant settings show consistent and significant gains over the baseline.

***Index Terms*—** speech codec, frequency–time domain fusion, low resource

1. INTRODUCTION

Speech interfaces have become essential in embedded systems, mobile devices, and other platforms with limited computational power or energy budgets. In such low-resource environments, speech codecs must deliver real-time processing while balancing complexity, bitrate, and latency, and still preserve high audio quality under noise and reverberation. While end-to-end neural audio coding has improved quality and compression efficiency, simultaneously achieving low complexity, low latency, low bitrate, and robustness in real acoustic conditions remains a major challenge.

The 2025 Low-Resource Audio Coding (LRAC) Challenge provides a stringent benchmark for this problem, with strict limits on complexity, latency, and bitrate, and a requirement for real-world operation. It serves both as a test of engineering capability and a driver for advances in integrated low-resource speech coding.

To address these demands, we propose a frequency–time domain fusion end-to-end audio codec for high-fidelity speech reconstruction under extreme resource constraints. The system combines frequency-domain encoding and time-domain decoding, augmented by a multi-stage training process, and noise-reduction techniques. These components jointly enhance transmission quality and fine-detail reproduction within tight computational and storage budgets, meeting LRAC's requirements for low latency, low bitrate, and high intelligibility, and delivering superior performance across diverse evaluation scenarios.

2. METHOD

2.1. Architecture

We propose a frequency–time domain fusion end-to-end audio codec that achieves high-quality speech transmission under strict resource constraints. The overall architecture, illustrated in Fig. 1, consists of a frequency-domain encoder, a residual vector quantizer (RVQ) [1, 2], and a time-domain decoder. The input audio is first transformed into an amplitude spectrogram via short-time Fourier transform (STFT). The frequency-domain encoder, built upon SpecTokenizer [3], employs a complex convolution layer followed by four cascaded FdownBlocks and RNNBlocks to extract and compress spectral features. Each FdownBlock combines a 2D convolution with Snake2D activation to enhance harmonic structure modeling, while each RNNBlock integrates FLNorm, Tanh, GRU, 2D convolution, and Snake2D activation, with residual connections to maintain stable gradient flow and preserve feature fidelity.

The latent representation is subsequently quantized by the RVQ module and passed to a BigCodec-based time-domain decoder [4]. This decoder comprises a 1D convolution, a unidirectional LSTM with residual connections, four sequential DecoderBlocks, Snake1D activation, an output 1D convolution, and Tanh activation. Each DecoderBlock contains Snake1D activation [5], a 1D transposed convolution for upsampling, and several ResidualBlocks. Each ResidualBlock consists of two 1D convolutions with different kernel sizes and Snake1D activations, coupled with a residual connection at the end, thereby improving high-frequency detail restoration and spatial perceptual quality in waveform reconstruction.

Model training adopts a multi-objective loss function, including multi-scale mel-spectrogram loss, VQ quantization loss, and GAN-based adversarial loss. During adversarial training, a Multi-Period Discriminator (MPD) and Multi-

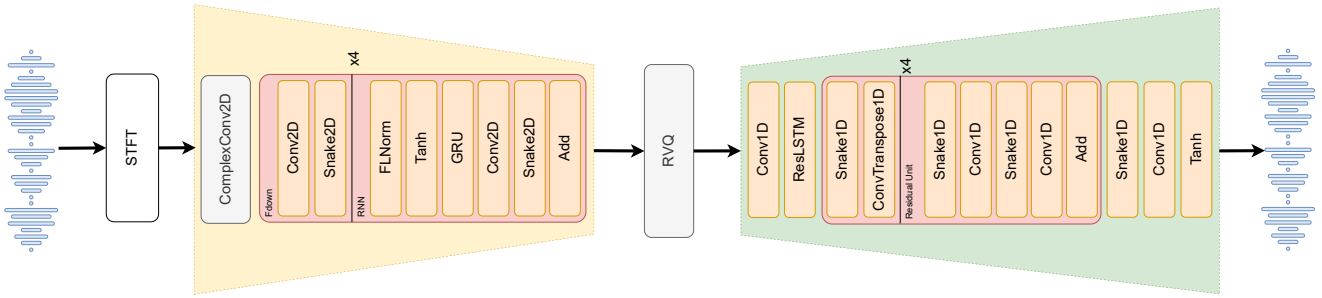


Fig. 1. The proposed model architecture.

Resolution Discriminator (MRD) [6] are employed jointly to constrain both time-domain details and spectral textures, significantly enhancing mid-to-high frequency energy reproduction and naturalness. As a result, the proposed system delivers high-fidelity speech reconstruction that combines audio quality and intelligibility under low-latency and low-bitrate conditions.

2.2. Training Stages

We first trained the codec without noise-reduction to obtain a performance-stable baseline model, and then introduced a noise-reduction stage after convergence. Although the competition rules explicitly state that noise-reduction features neither yield additional credit nor incur penalties in evaluation, our experiments show that incorporating this stage significantly improves speech quality in real acoustic environments. Consequently, we consider noise-reduction training an essential component of system optimization.

The codec training process consists of two parts: a Mel stage and a GAN stage. In the Mel stage, only the multi-scale mel-spectrogram loss is used for optimization. The model converges rapidly in this stage and achieves excellent reconstruction in the low-frequency range (0–1.5 kHz), with correspondingly high objective scores. However, because the mel loss provides insufficient constraint in the mid-to-high frequency range, the generated audio above 1.5 kHz often exhibits blurred spectral detail, energy attenuation, and slight mechanical artifacts, affecting subjective naturalness. To address this issue, we switch to the GAN stage after Mel-stage convergence, leveraging both the Multi-Period Discriminator (MPD) and Multi-Resolution Discriminator (MRD) for adversarial training. This significantly enhances mid-to-high frequency detail restoration, produces spectral energy distributions closer to natural speech, and effectively reduces mechanical noise. While some objective metrics (e.g., PESQ and Scoreq) degrade slightly in this stage, subjective ratings improve markedly, with richer spatial perception and more natural fine detail.

During training, we observed an interesting phenomenon: after several cycles in the GAN stage, returning to the Mel stage for further optimization causes objective scores not only to recover but to exceed the best results of the initial Mel stage. This may be because the GAN stage encourages the generator to explore a broader solution space, providing the mel loss with a better optimization starting point and helping the model escape local minima.

In the noise-reduction training stage, the input data comprise a random mix of clean, noisy, and reverberant speech, with the target output being the corresponding clean speech. The loss functions and hyperparameters are kept identical to those in codec training, and adversarial learning is again applied to further improve the realism and richness of generated audio. The discriminator configuration follows a staged policy: MPD alone in the early phase to strengthen time-domain periodicity discrimination; MPD plus MRD in the mid phase to impose multi-resolution spectral constraints; and MRD alone in the late phase to focus optimization on spectral detail restoration. Subjective listening tests indicate that this configuration yields the best improvements in mid-to-high frequency clarity, spectral extension, and overall intelligibility, producing speech more closely resembling real recordings.

3. EXPERIMENTS

3.1. Datasets

All training data in this study are sourced from the official LRAC2025 dataset and underwent rigorous filtering and preprocessing prior to use. For noise data, labels were predicted using a pre-trained audio understanding model, and any non-pure noise samples containing speech were removed to ensure clean noise content. For reverberation data, room impulse responses (RIRs) were truncated before convolution, retaining only the 1 ms segment following the peak. This reduces long-tail decay that can impair speech clarity while preserving spatial characteristics.

Based on this, we applied a data augmentation strategy by mixing clean, noisy, and reverberant speech in a 1:1:1 ratio. In noise mixing, the signal-to-noise ratio (SNR) was uniformly sampled within the range of 10–30 dB to increase acoustic

Table 1. Evaluation results for different bitrates and acoustic conditions.

Bitrate	Method	Clean					Noisy					Reverb				
		sheet ssqa	scoreq ref	audiobox AE_CE	utmos	pesq	sheet ssqa	scoreq ref	audiobox AE_CE	utmos	pesq	sheet ssqa	scoreq ref	audiobox AE_CE	utmos	pesq
1kbps	Baseline	1.84	1.15	3.90	1.44	1.15	1.72	1.29	3.40	1.33	1.11	1.85	1.36	2.94	1.26	1.07
	Proposed	3.79	0.35	5.31	3.42	2.09	3.65	0.38	5.18	3.32	1.92	2.80	0.59	4.53	2.58	1.46
6kbps	Baseline	3.84	0.35	5.28	3.23	2.67	3.12	0.82	4.37	2.70	1.81	2.22	1.13	3.43	1.32	1.18
	Proposed	4.17	0.18	5.62	3.77	2.98	3.99	0.30	5.45	3.64	2.50	3.14	0.53	4.75	2.74	1.62

Table 2. Latency breakdown of the proposed system.

Source	**Samples**	**Notes**
STFT hopsize	192 @ 16kHz	Frame shift
Decoder Residual Units	272 @ 16kHz	$64 \times 3 + 16 \times 4 + 4 \times 4 + 1 \times 5$
Final decoder convolution	3 @ 16kHz	Kernel size = 7
Resampling delay	8 @ 24kHz	Maximum group delay of the IIR filter
Total (24kHz)	716 @ 24kHz (29.83 ms)	$472 \times \frac{3}{2} + 8$

diversity.

Model evaluation was conducted on an open test set from the same source, with inference performed directly on the original official data without additional processing, and performance tested at both 1 kbps and 6 kbps bitrates.

3.2. Implementation Details

The proposed model has an overall computational complexity of 698 M FLOPs and 1.48 M parameters, with the encoder and RVQ module accounting for 399 M FLOPs and 1.17 M parameters, and the decoder for 299 M FLOPs and 0.32 M parameters. The system operates at a sampling rate of 24 kHz, with a frame length of 720 samples and a frame shift of 288 samples (approximately 83 Hz frame rate). In the STFT computation, only frequency bins 0–240 (0–8kHz) are used, effectively yielding a 24kHz to 16 kHz downsampling without introducing additional latency.

The encoder employs convolution kernels and strides of 1, introducing no additional latency. The decoder primarily uses causal convolutions and causal transposed convolutions, but non-causal convolutions are applied in specific positions to enhance reconstruction quality: the first convolution layer in the decoder (kernel = 1, stride = 1), the first convolution layer within repeated ResidualBlocks (kernel sizes = [7, 9, 9, 11], stride = 1), and the final convolution layer in the decoder (kernel = 7, stride = 1). These designs significantly improve mid-to-high frequency detail within the latency budget. The end-to-end latency is determined by both the STFT window length and the non-causal convolutions, and is kept within 30 ms overall.

To convert the 16 kHz audio output of the decoder to 24 kHz without noticeably increasing latency, we use a fractional-rate resampling strategy. First, the signal is upsampled by a factor of three using zero-insertion. Next, the spectral images introduced by zero-insertion are removed with an 11th-order IIR Butterworth low-pass filter with an 8 kHz cutoff frequency. Finally, the signal is downsampled by a factor of two to reach the target sampling rate. Compared to an FIR-based approach, this IIR design exhibits a maximum passband group delay of only 8 samples near 8 kHz, making it well-suited for real-time applications. The latency breakdown is shown in Table 2.

The RVQ module consists of six codebooks, each containing 4096 entries (indexed with 12-bit codes) and a vector dimension of 8. During inference, either 1 codebook (for 1 kbps) or all 6 codebooks (for 6 kbps) can be selected, enabling operation at two different bitrates. The encoder channel configuration is [32, 32, 32, 128, 335], with time-axis kernel sizes and strides of [1, 1, 1, 1] and frequency-axis kernel sizes and strides of [5, 4, 4, 3]. The decoder channels are [117, 58, 29, 14, 7], with upsampling rates of [3, 4, 4, 4]. For the discriminators, the MPD uses periods [2, 3, 5, 7, 11], and the MRD operates with window sizes [128, 256, 512, 1024, 2048].

For optimization, both the generator and discriminators use an initial learning rate of 8×10^{-4} during the Mel stage and 1×10^{-4} during the GAN stage, gradually reduced to 1×10^{-5}. Adam is used throughout all training stages.

Checkpoint Selection Strategy: For system submission, we

performed subjective listening evaluations on multiple models from different training stages using the open test set, selecting the checkpoint that yielded the best combination of audio quality and fine-detail reproduction as the final competition version.

3.3. Results

Our evaluation uses Versa [7], the official toolkit recommended by the 2025 LRAC Challenge, which provides standardized implementations of multiple metrics, including *sheet_ssqa*, *scoreq_ref*, *audiobox AE_CE*, *UTMOS*, and *PESQ*. Experiments are conducted under three acoustic conditions: clean, noisy, and reverberant. Using the RVQ module's ability to achieve variable bitrate by selectively dropping codebooks during inference, we further evaluate the model at 1 kbps and 6 kbps.

The evaluation results are summarized in Table 1. Under all three acoustic conditions and both bitrates, the proposed method outperforms the baseline system across all metrics.

4. CONCLUSION

We propose a frequency–time domain fusion end-to-end codec for low-resource audio coding, combining iterative optimization with noise-reduction to enhance quality and robustness across diverse acoustic conditions and bitrates. Exploiting the complementarity of frequency-domain encoding and time-domain decoding, the system achieves high-fidelity speech reconstruction within strict complexity and latency limits. Experiments demonstrate consistent gains over the baseline in clean, noisy, and reverberant settings, confirming the effectiveness of the approach and its potential for more complex scenarios.

5. REFERENCES

[1] Neil Zeghidour, Anatoly Luebs, Mohammad Omran, Jan Skoglund, and Marco Tagliasacchi, "SoundStream: An end-to-end neural audio codec," in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2021, vol. 30, pp. 495–507.

[2] Alexandre Défossez, Neil Zeghidour, Nicolas Usunier, and Gabriel Synnaeve, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[3] Zixiang Wan, Guochang Zhang, Yifeng He, and Jianqiang Wei, "SpecTokenizer: A lightweight streaming codec in the compressed spectrum domain," in *Proc. Interspeech 2025*, 2025, pp. 599–603.

[4] Detai Xin, Xu Tan, Shinnosuke Takamichi, and Hiroshi Saruwatari, "BigCodec: Pushing the limits of low-bitrate neural speech codec," *arXiv preprint arXiv:2409.05377*, 2024.

[5] Sang gil Lee, Wei Ping, Boris Ginsburg, Bryan Catanzaro, and Sungroh Yoon, "BigVGAN: A universal neural vocoder with large-scale training," 2023.

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

[7] Jiatong Shi, Hye jin Shim, Jinchuan Tian, Siddhant Arora, Haibin Wu, Darius Petermann, Jia Qi Yip, You Zhang, Yuxun Tang, Wangyou Zhang, Dareen Safar Alharthi, Yichen Huang, Koichi Saito, Jionghao Han, Yiwen Zhao, Chris Donahue, and Shinji Watanabe, "VERSA: A versatile evaluation toolkit for speech, audio, and music," 2025.

LOW RESOURCE AUDIO CODEC CHALLENGE Sublime SYSTEM DESCRIPTION

Piotr Dura
piotrdura7@gmail.com

***Abstract*—This work aims to advance neural audio coding by exploring novel approaches for Speech Vocoding and Vector Quantization (VQ). Both Track 1 and Track 2 systems are proposed, and both are convolutional encoder-decoder models with discrete representation emitted by the encoder. The decoder is a conv1d-conv2d hybrid Fourier-domain vocoder we call Sublime. Both Tracks share the same Vocoder weights. A novel quantization scheme, which we call Simulated Annealing Vector Quantization (SAVQ), is proposed along with a method to prevent codebook collapse.**

***Index Terms*—LRAC 2025, audio coding, VQ, generative adversarial networks**

I. INTRODUCTION

In this work, we present the design of a participant system for the 2025 LRAC challenge Tracks 1 and 2. Track 1 system is comprised of the encoder (3.8M params, 399.7 MFLOPS) and decoder (2.5M params, 294.1 MFLOPS). Track 2 system also contains a frontend (20.6M params, 2284.6 MFLOPS). Quantizer can operate in two modes — 1kbps and 6kbps, both modes can be used interchangeably by the decoder. The model is fully causal, but the buffering latency of analysis-synthesis accounts for the full 30ms end-to-end latency budget. Track 2 reuses the decoder weights, and instead of the encoder-SAVQ combination, a separate convolutional encoder is trained with the objective of predicting the codes via a Cross Entropy Loss. Track 2 encoder has an additional 20ms of algorithmic latency which result in a 50ms end-to-end latency. The latency figures are not estimated, but are the worst-case, measured latencies imposed by the algorithm. Presented MFLOPS numbers are obtained using a pytorch calflops package.

Total amount of training time spent on both Tracks is less than 120 gpu-hours on an NVIDIA RTX 4090, out of which 96 gpu-hours were assigned for Track 1 and 24 gpu-hours for Track 2.

II. ENCODER

The first processing stage converts the input 24kHz mono waveform into two log-mel spectrograms (10ms hop, 20ms window, 64 filters and 10ms hop, 30ms window, 96 filters) and concatenates them in channel dimension. The result is processed by a conv1d block (kernel size 3, 160 input channels, 256 hidden channels, 120 feed-forward channels), then the frames are stacked with stride $N = 2$ to form a 20ms-per-frame sequence. Causal stacking is used to not increase the latency, so that the initial stacked frames are partial during inference. The stacked sequence is further processed by 8 conv1d blocks (each has kernel size 3, 384 hidden channels, 384 feed-forward channels) and projected into query vectors q. Conv1d blocks are inspired by ConvNeXt [1] but use vanilla (non-depth-separable) conv1d and include a transformer-style feed-forward block with expansion and RMSNorm normalization.

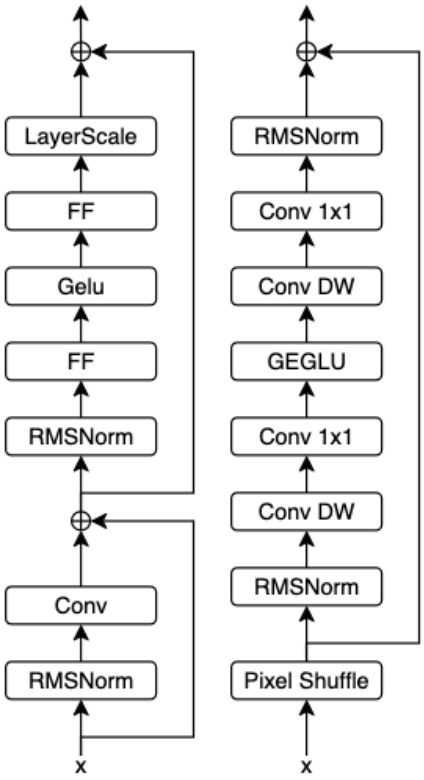


Fig. 1. Conv1d and Conv2d blocks.

III. QUANTIZATION

Standard quantization schemes require finding a nearest-neighbor embedding out of an embedding table for each frame of the input using an L2 or cosine distance. Since the embedding lookup is nondifferentiable, a straight-through estimation (STE) is typically used. An optional commitment loss can be used to penalize the distance between the input frames and the quantized output frames. Typical implementations leverage techniques like K-means initialization, dead code revival or smoothing of the update of an embedding table via an EMA. To improve the efficiency of compression,

SoundStream introduces a residual VQ [4]. RVQ quantizes the input and then iteratively quantizes the resulting quantization error with a number of separate embedding tables. More recent approaches such as FSQ [5] avoid using an embedding table altogether.

The proposed SAVQ utilizes cosine Cross-Attention with a learnable bank of embeddings and is parametrized by the temperature T:

$$\text{SAVQ}(q, k, v; T) = \text{softmax}\left(\frac{\sqrt{D}\ \text{cosine\_sim}(q, k)}{T}\right) V$$

where $k, q \in \mathbb{R}^D$

As the training progresses the temperature is annealed with a fixed annealing schedule. In the early stages when the temperature is high, attention over the embeddings has high entropy. Over time the sharpness of the attention increases and the behavior of the system shifts towards compression. As temperature approaches zero, attention over the embeddings approaches a one-hot vector. Notice, that a standard dot-production attention would not be effective for this purpose, as the network would be able to arbitrarily parametrize norms of query-key pairs. Second, because the cosine metric is used the normalization term of $\sqrt{D}$ is moved to the numerator. To increase the efficiency of compression, G groups of embeddings have been used, which is equivalent to a multi-head attention.

To enable efficient learning of the encoder even with low temperatures a temperature floor parameter T_F is introduced. Activations that are emitted by the quantizer are calculated using the original T, only the gradient that flows back to the encoder is modified as if the attention weights were calculated using $max(T, T_F)$.

This formulation, while empirically effective, suffered from codebook collapse, where roughly 10-20% of all codes ended up never being the top-1 activation. As the training progressed and temperature was annealed these codes were never reused by the model. A simple technique would be to employ entropy maximization loss:

$$L_H(p) = -H(p) = \sum_{t,k} p(t,k)\ log\ p(t,k)$$

where p is a categorical distribution over codes in a given codebook, t is batch-time-step, k is embedding index.

Since we don't want to penalize low codebook entropy as long as all codes have non-zero usage, we applied an ad-hoc loss called reciprocally-weighted smoothed surprisal (RWSS):

$$RWSS(p) = -\sum_{k} \frac{1}{p_K(k) + \epsilon} \sum_{t \in Q_q(p,k)} log\ p(t,k)$$

where $p_K(k)$ denotes empirical probability that the code k is a top-1 activation calculated over batch examples and time steps, ϵ is a smoothing constant, Q_q is a set of batch-time-steps that contains upper q-quantile of all $p(:, k)$

Intuitively, entropy maximization would penalize high log-probabilities of "activated" tokens and would move the attention weights towards a uniform distribution. RWSS loss penalizes low log-probabilities of tokens that are rarely activated (low $p_K(k)$) and routes that penalty only to the frames that already have high contribution of those codes. Version of this loss that penalized all frames instead of the top q-quantile resulted in a codebook in which code utilization oscillated highly over time.

Two quantizers are trained in parallel. Quantizer A uses $G = 4$ groups, each containing $K = 32$ embeddings at 50 frames-per-second. Quantizer B uses $G = 20$ groups. Ultra-low bitrate mode is achieved by calculating both quantizer outputs and adding the resulting embeddings. During training the quantizer B embedding is added with a probability of 50%.

IV. VOCODER

Following recent SOTA systems (Vocos [6], Wavehax [7]) we design a Sublime (SUB-band LInear Magnitude-phase Estimation) vocoder which converts the latent space of the quantizer z into a log-magnitude spectrogram $\hat{M}_{log}$ and raw phases $\hat{P}$ that are inverted using an $ISTFT$ ($20ms$ hop, $40ms$ window):

$$\hat{y} = ISTFT(e^{\hat{M}_{log} + i\hat{P}})$$

Input of the vocoder z is processed by 4 conv1d blocks (kernel size 3, 256 hidden channels, 384 feed-forward channels), then another 4 conv1d blocks (kernel size 3, 256 hidden channels, 256 feed-forward channels), then three separate sub-band conv2d decoders are used to produce three 4d tensors of shape $[batch, features, channels, time]$. All three tensors are concatenated along the channel dimension and projected via conv2d to a $[batch, 2, channels, time]$ tensor containing the log-magnitudes and phases. These sub-band decoders emit the following frequency bands: $[0 - 2kHz]$, $[2 - 6kHz]$ and $[6 - 12kHz]$. Each sub-band decoder is composed of a series of Pixel-Shuffle (PS) upsampling layers, each followed by Universal Inverted Bottleneck (UIB) block introduced in MobileNet V4 [2] and include multiplicative activation GEGLU [3]. PS layers upsample only in the channel dimension, however versions that upsample in time dimension coupled with 10ms or 5ms $ISTFT$ were also tested. The final configuration specifies 2 upsampling layers for the 1st and 2nd sub-band, each with upsample rate $[2, 1]$ and followed by a single UIB block with 8 feature maps, kernel size $[5, 3]$ in both depth-wise convolutions, and expansion factor 1.5. The last sub-band decoder uses a single upsample layer with upsample rate $[4, 1]$ and a single UIB block with kernel size $[3, 3]$.

Training of the vocoder utilized an ensemble of three discriminators: Multi-Period Discriminator (MPD), Multi-scale STFT Discriminator (MSSTFTD) and a Multi-scale Magnitude Discriminator (MSMAGD) which has the same architecture as MSSTFTD, but uses log-magnitude inputs, instead of complex-valued inputs. MSSTFT and MSMAGD use 128 feature maps.

V. TRAINING

Track 1 system has been trained in two phases. In both phases an encoder with a decoder has been both optimized with a waveform reconstruction task.

In the first phase, temperature has been annealed for $20k$ steps from an initial $T_0 = 0.02$ to $T_1 = 0.01$ with a cosine decay, then for additional $130k$ steps using an exponential decay, halving temperature every $8k$ steps. Temperature floor was set to $T_F = 0.01$. Losses used in this phase were multi-scale $L1$ mel loss with weight $w_{mel} = 10.0$, multi-scale $L1$ mfcc loss with weight $w_{mfcc} = 1.0$, as well as RWSS loss with weight $w_{rwss} = 1.0$, smoothing factor $\epsilon = 0.001$ and $q = 0.05$.

In the second phase, encoder was frozen, temperature set to $T = 0$ and discriminators were enabled. Training losses consisted of multi-scale $L1$ mel loss $w_{mel} = 10.0$, feature-matching loss $w_{fm} = 1.0$ and discriminator loss $w_d = 1.0$. In this phase the network was trained for a total of $160k$ steps which is short of the full convergence.

Track 2 system has been trained by freezing the Track 1 system, and training a separate frontend used instead of the encoder-SAVQ, with a cross-entropy objective. Prediction of the codes is assumed to be conditionally independent between the codebooks, and during inference greedy decoding is performed. Track 2 system has been trained for a total of $120k$ steps.

All three training runs use AdamW optimizer and follow a cosine learning-rate decay between $lr_0 = 2e-4$ and $lr_{200k} = 1e-4$, with effective batch size of 32.

VI. DATASET

In Track 1, first phase trained with the full provided training set, with a segment size of 3 seconds. Phase 2 trained with a clean split of the provided training set, with a segment size of 1 seconds. Track 2 system was trained with a clean split of the training set, using full utterances and batch zero-padding. Clean split was obtained by calculating UTMOS score and taking the top 60% of all utterances.

All training runs set the gain of audio to a $dB\ RMS$ level drawn randomly from a $[-18dB, -6dB]$ range. Inputs of the model are degraded by a sequence of data augmentation steps. First, random RIR from the provided set of RIRs is convolved with the input (with probability 25% for Track 1 and 40% for Track 2), then random noise from the provided set of training noises is added with a randomly sampled $dB\ SNR$ ($[6dB - 30dB]$ for Track 1 and $[-6dB - 30dB]$). Lastly a down-sampling is simulated with probability 20% of obtaining $8kHz$ sampling rate, and 50% of obtaining $16kHz$ sampling rate.

VII. EVALUATION

Final model checkpoint has been selected by comparing UTMOS scores calculated on an open testset set, combined with manual listening. Tables I, II, III, IV, V, VI contain UTMOS Results of a submitted checkpoint followed by results of a converged checkpoint (trained for a total of 1.3M steps for Track 1 and 2M steps for Track 2) in parentheses. All UTMOS values are calculated on an open testset.

Model	Clean
baseline (1 kbps)	1.44
proposed (1 kbps)	2.49 ± 0.5 (2.69 ± 0.51)
baseline (6 kbps)	3.23
proposed (6 kbps)	3.14 ± 0.57 (3.33 ± 0.57)

TABLE I
TRACK 1 UTMOS CLEAN

Model	Noisy
baseline (1 kbps)	1.33
proposed (1 kbps)	2.47 ± 0.47 (2.65 ± 0.48)
baseline (6 kbps)	2.7
proposed (6 kbps)	3.05 ± 0.54 (3.21 ± 0.54)

TABLE II
TRACK 1 UTMOS NOISY

Model	Reverb
baseline (1 kbps)	1.26
proposed (1 kbps)	2.16 ± 0.43 (2.28 ± 0.42)
baseline (6 kbps)	1.32
proposed (6 kbps)	2.58 ± 0.52 (2.7 ± 0.49)

TABLE III
TRACK 1 UTMOS REVERB

Model	Clean
baseline (1 kbps)	1.37
proposed (1 kbps)	2.48 ± 0.48 (2.69 ± 0.48)
baseline (6 kbps)	2.97
proposed (6 kbps)	3.14 ± 0.56 (3.36 ± 0.55)

TABLE IV
TRACK 2 UTMOS CLEAN

Model	Noisy
baseline (1 kbps)	1.35
proposed (1 kbps)	2.35 ± 0.51 (2.71 ± 0.53)
baseline (6 kbps)	2.56
proposed (6 kbps)	2.85 ± 0.6 (3.24 ± 0.56)

TABLE V
TRACK 2 UTMOS NOISY

Model	Reverb
baseline (1 kbps)	1.32
proposed (1 kbps)	2.27 ± 0.47 (2.63 ± 0.49)
baseline (6 kbps)	1.79
proposed (6 kbps)	2.66 ± 0.55 (3.23 ± 0.55)

TABLE VI
TRACK 2 UTMOS REVERB

REFERENCES

[1] Liu, Z., Mao, H., Wu, C., Feichtenhofer, C., Darrell, T. & Xie, S. A ConvNet for the 2020s. (2022), https://arxiv.org/abs/2201.03545

[2] Qin, D., Leichner, C., Delakis, M., Fornoni, M., Luo, S., Yang, F., Wang, W., Banbury, C., Ye, C., Akin, B., Aggarwal, V., Zhu, T., Moro, D. & Howard, A. MobileNetV4 – Universal Models for the Mobile Ecosystem. (2024), https://arxiv.org/abs/2404.10518
[3] Shazeer, N. GLU Variants Improve Transformer. (2020), https://arxiv.org/abs/2002.05202
[4] Zeghidour, N., Luebs, A., Omran, A., Skoglund, J. & Tagliasacchi, M. SoundStream: An End-to-End Neural Audio Codec. (2021), https://arxiv.org/abs/2107.03312
[5] Mentzer, F., Minnen, D., Agustsson, E. & Tschannen, M. Finite Scalar Quantization: VQ-VAE Made Simple. (2023), https://arxiv.org/abs/2309.15505
[6] Siuzdak, H. Vocos: Closing the gap between time-domain and Fourier-based neural vocoders for high-quality audio synthesis. (2024), https://arxiv.org/abs/2306.00814
[7] Yoneyama, R., Miyashita, A., Yamamoto, R. & Toda, T. Wavehax: Aliasing-Free Neural Waveform Synthesis Based on 2D Convolution and Harmonic Prior for Reliable Complex Spectrogram Estimation. (2024), https://arxiv.org/abs/2411.06807

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

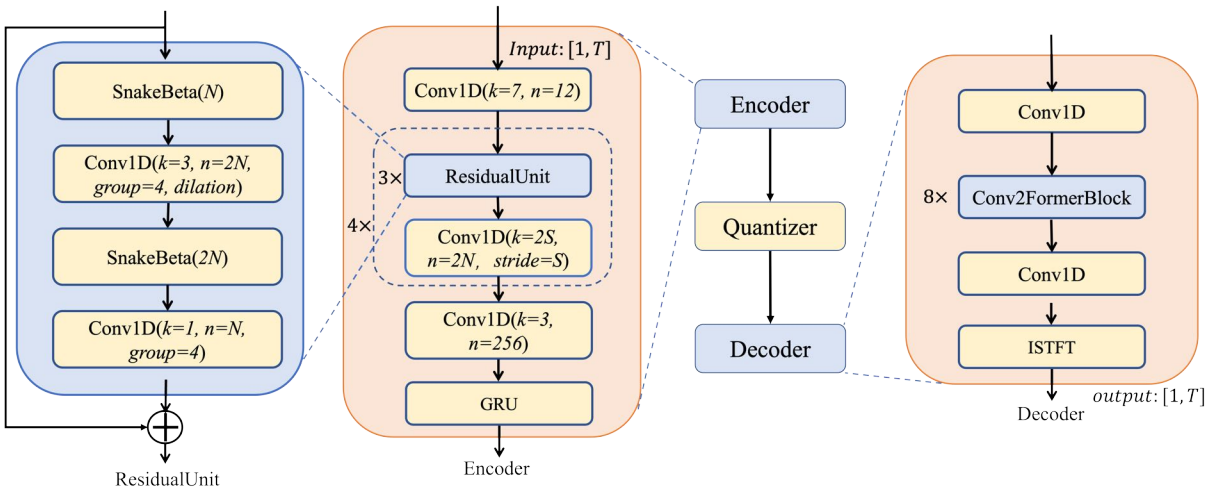


Fig. 1. Schematic diagram of the model architecture

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.

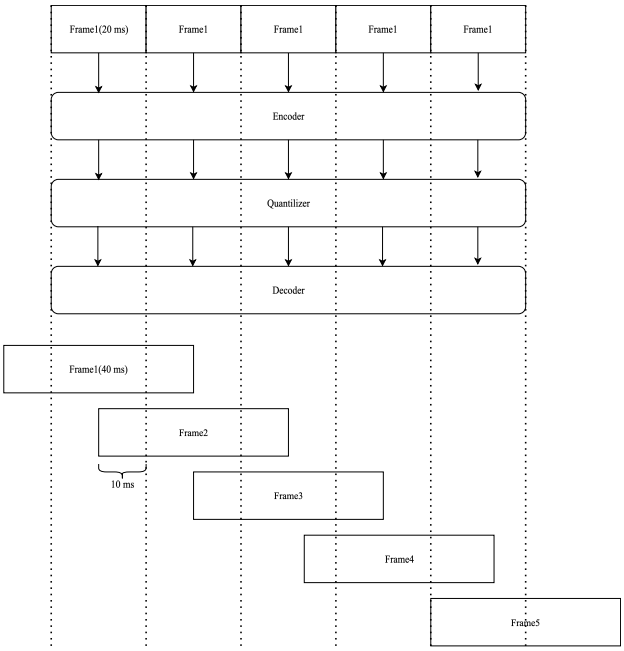


Fig. 2. System latency description

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.

HORCODEC: HORNET BASED NEURAL AUDIO CODEC FOR THE LRAC 2025 CHALLENGE TRACK 1

Qingbo Huang, Weihao Xiong, Congxin Zhang, Xinmin Yan

ByteDance

ABSTRACT

This paper is a description of our team's submission model for LRAC track 1, introducing the HORCODEC based on HORNET, including model structure, training methods, and other details. By introducing Horunit into classic methods such as soundstreaemDAC model and RVQ, our model can consistently improve dense prediction performance with less computation, achieving transparent sound quality as much as possible within the low complexity requirement by LRAC.

***Index Terms*—** neural audio codec, residual vector quantilization

1. INTRODUCTION

High quality and low latency audio encoding algorithms are crucial in real-time communication field. With the rapid development of deep learning technology in recent years, audio codec based on deep neural networks, represented by soundstream[1], DAC[2], have significantly improved compression efficiency compared to traditional audio encoders such as AAC and OPUS. However, the high latency and high complexity of encoding and decoding are fatal flaws of deep neural network-based audio codecs, which prevent them from being widely used in real-time communication. Regarding this issue, LRAC competition track 1 has made clear regulations on the complexity and delay of encoder encoding and decoding. This is extremely challenging for deep neural networks-based audio encoders. To achieve the ultimate goal of low complexity, low latency, and transparent sound quality, we have researched the current mainstream audio encoding methods based on deep neural networks and have referred to the forefront of deep learning in computer vision. Based on the DAC and VOCOS frameworks, we have added the improvement of the basic modules in the transformers described in HORNET[3]. Under the premise of satisfying the requirements of complexity and latency in LRAC, the sound quality of our proposed codec is as close as possible to the original high-complexity DAC.

2. MODEL STRUCTURE

The over view of the proposed codec is shown in Figure 1. The input audio is divided into frames with a frame length of 20ms. The output feature of the encoder network is coded by RVQ, with 0.5kbps for each layer. On the decoder side, the input feature is transformed to the frequency domain, and then audio signals in the time domain are generated by ISTFT.

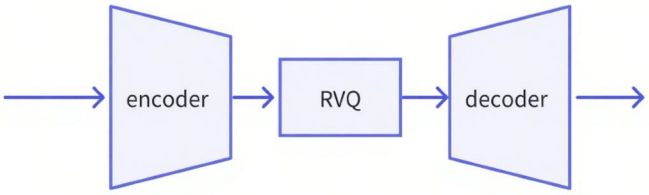


Fig. 1. overview

2.1. Encoder Block

The encoder structure is shown in Figure 2. The first 1D convolution module is set to kernel size k=7. Then four residual convolution modules are applied in sequence with each stride = 3, 4, 5, 8. For each residual convolution module, there are 3 residual units in it and each residual unit's dilation is 1, 3, 9 respectively.

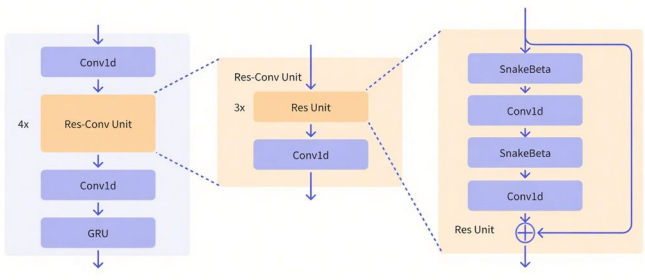


Fig. 2. encoder block

2.2. Decoder Block

Inspired by Hornet in computer vision, we modified the module based on 2D convolution design in Hornet and applied it to audio signal processing. The decoding end receives the quantized feature vectors, which are sequentially processed through one 1D convolution module, 6 horunit modules, and one 1D convolution module before being converted to the frequency domain. The frequency domain signal is then transformed back to the time domain through ISTFT. Each horunit module contains one gConv gating module and one FNN module in sequence, with the gConv gating order set to 3, as shown in Figure 3.

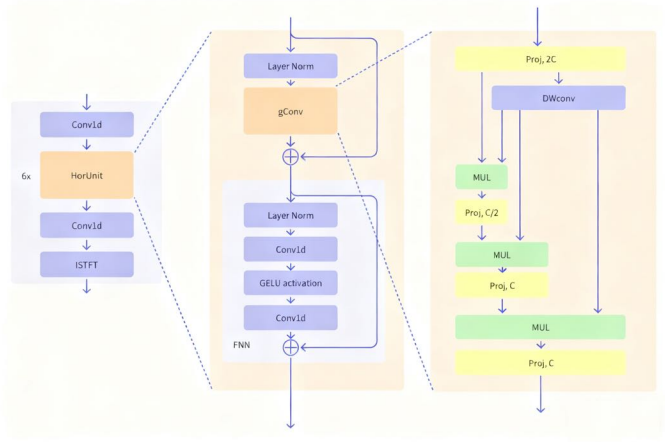


Fig. 3. decoder block

2.3. Quantizer

The features outputted from each frame undergo RVQ (Residual Vector Quantization) hierarchical residual layer coding, consisting of 12 layers. Each layer has 1024 codeword candidates, which requires 10 bits per layer during encoding. Given that the encoding segment is divided into 20ms frames, the bit rate for one layer of RVQ quantization stands at 0.5kbps. If the target bit rate is 6kbps, all 12 layers of RVQ are employed; whereas if the target bit rate is 1kbps, only the first two layers of RVQ are utilized.

2.4. Computational Complexity

The parameter count and computational complexity of each module in the model are shown in Table 1

2.5. System Latency

Since the encoder frame length is 480 points (i.e. 20ms) and there are 240 points (i.e. 10ms) frame overlapping in the decoder, the system latency is 30ms, satisfying the challenge requirement.

	Parameter Count	Model Complexity
Enocder	0.98M	172.97MMACs
Quantizer	0.19M	1.06MMACs
Decoder	3.01M	154.78MMACs
Total	4.18M	328.82MMACs

Table 1. Computational Complexity

3. TRAINING

3.1. Data Processing

On the premise of complying with the competition requirements, we have pre-processed the data provided by the official to achieve data augmentation. When generating noisy frequencies, randomly select SNR within a preset interval. When generating reverberation data, follow the official method and generate it with an appropriate reverberation ratio. For multispeaker data, randomly adjust the volume of a certain speaker.

3.2. Loss Setups

Training is carried out in the form of a generative adversary mode,which is the same as SoundStream and DAC. As described in Equation 1, the total loss of the model is consist of GAN-based loss $\mathcal{L}_g$, RVQ commit loss $\mathcal{L}_c$, RVQ code book loss $\mathcal{L}_r$ related to RVQ to improve the efficiency of code book utilization. The reconstruct loss $\mathcal{L}_{re}$ is set to ensure that reconstructed signal is as consistent as possible with the reference input.

$$\mathcal{L} = \lambda_g * \mathcal{L}_g + \lambda_c * \mathcal{L}_c + \lambda_r * \mathcal{L}_r + \lambda_{re} * \mathcal{L}_{re} \quad (1)$$

Since the reconstruction loss does not occupy the complexity of encoding and decoding, we set a loss function as detailed as possible to evaluate the quality of the reconstructed signal, although this may slow down the training process. The reconstruct loss is set with multiscale STFT loss $\mathcal{L}_{stft}$, multiscale MEL loss $\mathcal{L}_{mel}$, PESQ[4] loss $\mathcal{L}_{peaq}$. The multiscale STFT loss is set with window lengths of 256, 512, 1024 and 2048. The multiscale MEL loss is set with window lengths of 32, 64, 128, 256, 512, 1024 and 2048, corresponding mel bin counts of 5, 10, 20, 40, 80, 160, and 320, respectively. All loss functions are weighted with appropriate coefficients as part of the final loss.

$$\mathcal{L}_{recon} = \lambda_s * \mathcal{L}_{stft} + \lambda_m * \mathcal{L}_{mel} + \lambda_p * \mathcal{L}_{peaq} \quad (2)$$

All the weight coefficients are described in Table 2.

3.3. Network Training Configurations

The learning rate is initialized at 0.0001 and decays by a factor of 0.999996 every epoch, as described in the exponential

loss weight	value
λ_g	1.0
λ_c	0.25
λ_r	1.0
λ_{re}	1.0
λ_s	10.0
λ_m	15.0
λ_p	5.0

Table 2. loss weight configuration

learning rate scheduling technique. The Optimization is performed with Adam, using betas of 0.8 and 0.99.

We train the networks with a batch size of 16 per GPU, and 8 GPUs were used in training progress for Track 1 in total. The model is trained for 500 epochs. We use the checkpoint with the lowest reconstruction loss on the validation set. The reconstruction loss configuration is described in subsection 3.2.

4. REFERENCES

[1] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2021.

[2] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar, "High-fidelity audio compression with improved rvqgan," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[3] Yongming Rao, Wenliang Zhao, Yansong Tang, Jie Zhou, Ser-Nam Lim, and Jiwen Lu, "Hornet: Efficient high-order spatial interactions with recursive gated convolutions," *arXiv preprint arXiv:2207.14284*, 2022.

[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.

NANOCODEC: TOWARDS LOW BITRATE AND LOW COMPLEXITY REAL-TIME NEURAL AUDIO CODEC

Andong Li[⋆*], *Pinglin Xu*[†], *Zhe Han*[†], *Lingling Dai*[⋆*], *Yiqing Guo*[†], *Hua Gao*[†], *Xiaodong Li*[⋆*], Chengshi Zheng[⋆*]

⋆ Institute of Acoustics, Chinese Academy of Sciences, Beijing, China
†ByteDance, China
*University of Chinese Academy of Sciences, Beijing, China

ABSTRACT

In this report, we present NanoCodec, our submitted system for the LRAC Challenge Track 1, which can effectively reconstruct target waveform under ultra-low and low bitrates conditions. Specifically, our architecture operates in the time-frequency (T-F) domain, where we drop the phase and only encode the magnitude feature in the encoder side, and both are estimated in the receiver side. In addition, we propose an efficient convolution-style attention block as the core modeling unit. Given the strict constraint on the decoder complexity, the omnidirectional phase and real-imaginary losses are introduced to enable the effective joint optimization of target magnitude and phase. The submitted system achieves a total latency of 30 ms and a computational complexity of 685 MFlops (390M for the encoder and 295M for the decoder), satisfying the challenge requirements.

Index Terms— Neural audio codec, low-complexity, low bitrate, real-time, speech transmission

1. INTRODUCTION

Audio codecs are designed to convert original waveforms into compact bitstreams for transmission, followed by target decoding at the receiver. In recent years, neural audio codecs (NACs) have surged in popularity, propelled by the advancement of large language models (LLMs). Compared to traditional methods, NACs offer both higher compression ratios and reconstruction quality over [1, 2]. However, while most studies leverage NACs as audio tokenizers for generation tasks, real-time audio transmission remains underexplored [1, 3], where computational cost, causality, and algorithmic delay are regarded as significant factors to hinder the deployment of NACs in practical transmission scenarios.

LRAC Chalenge 2025 aims to gather research attention in real-time (RT) audio transmission under strict constraints on training dataset, calculation complexity and processing delay[1]. Specifically, Track 1 is devised for transparent transmission, with a maximum complexity of 700 MFlops (400 M for the encoder and 300 M for the decoder), and a total latency $\leq$ 30ms. To our best knowledge, existing literature rarely satisfies these requirements, thus posing a significant challenge for neural audio codec design.

To this remedy, in this paper, we present the proposed **NanoCodec**, which contributes in both architecture design and optimization regime. First, the proposed codec is based on time-frequency (T-F) domain, where we ignore the phase and only magnitude is utilized for feature encoding, and both targets are reconstructed in the decoder. The rationale lies in that given the limited calculation resource, it seems challenging for target coding or estimation in the time domain. As such, we employ the Fourier prior to alleviate the learning difficulty. Besides, given the limited bit resource, separate phase encoding can be trivial due to the wrapping effect of phase component. Second, we adopt a convolution-style attention block for spectral modeling, where the attention distribution is generated via large convolution kernels to effectively aggregate the contextual information. Third, it remains an open question for joint magnitude and phase estimation, especially under limited calculation resource. Motivated by [4], we employ an omnidirectional phase loss for phase optimization, efficiently capturing differential relations between centering and neighboring phase bins. we further generalize it into the real and imaginary (RI) parts of the spectrum, and propose an omnidirectional RI loss. By incorporating the above-mentioned tactics together, NanoCodec can reconstruct waveforms with high-quality under both low complexity and low bitrate scenarios.

[1]https://crowdsourcing.cisco.com/lrac-challenge/2025/

2. METHOD ILLUSTRATIONS

2.1. Overall Architecture

The overall diagram of the proposed NanoCodec is presented in Fig. 1(a), where both encoder and decoder are operated in the T-F domain. Given the input waveform $\mathbf{x} \in \mathbb{R}^L$, it is first transformed into the spectrum $\mathbf{X} \in \mathbb{C}^{F\times T}$ via the short-time Fourier transform (STFT), where $\{F, T\}$ denote the frequency and time axes, respectively. Different from previous literature where magnitude and phase are separately encoded [5], here we drop the phase and only preserve the magnitude for feature extraction. The reasons are two-fold. First, due to the restricted computational complexity in the encoder, as well as limited bit resource, the modeling priority should be provided to the magnitude as it exhibits more clear structural patterns over phase. Besides, phase usually exhibits random distribution due to the intrinsic wrapping effect, and it can be trivial for separate feature extraction from phase. Motivated by [6], the energy-content decoupling (ECD) layer is utilized to decouple the spectral energy and content, which is reported to mitigate the extra input energy normalization operation, given by:

$$\mathbf{I}_t = \text{Concat}\left(\log\left(E_t\right), \frac{|\mathbf{X}_t|}{E_t}\right) \in \mathbb{R}^{F+1}, \tag{1}$$

where E_t denotes the calculated energy for the t-th input frame, and Concat $(\cdot)$ is the concatenation operation along the feature dimension. After that, N_e modeling units are stacked for modeing.

For the decoder, similar to the encoder, N_d modeling units are stacked, and separate magnitude and RI heads are adopted for magnitude and phase estimation, respectively. After that, the inverse STFT operation is utilized for target waveform generation.

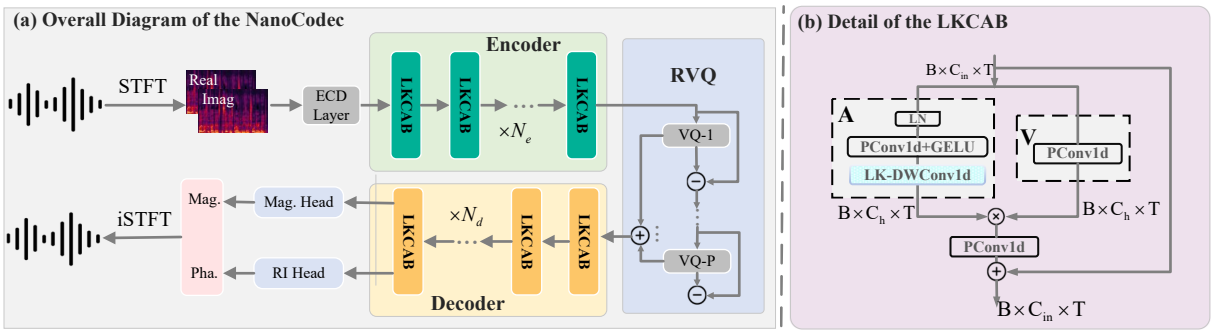


Fig. 1. (a) Overall structure of the proposed NanoCodec; (b) Internal structure of the adopted LKCAB.

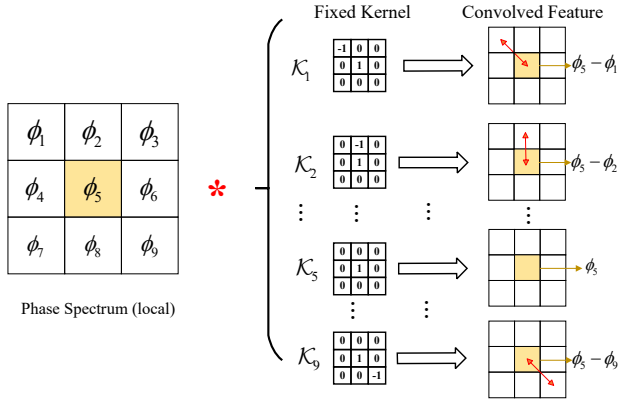


Fig. 2. Illustration of the omnidirectional phase loss.

2.2. Large Kernel Convolution-Style Attention Block

We share the same modeling unit for both encoder and decoder, and detailed internal structure is shown in Fig. 1(b). Given the input $\mathbf{H}_{i-1} \in \mathbb{R}^{B\times C_{in}\times T}$ of the i-th block, where C_{in} represents the input feature channel, it passes the attention branch and value branch to obtain the attention and value feature maps$\{\mathbf{A}_i, \mathbf{V}_i\} \in \mathbb{R}^{B\times C_h\times T}$, respectively. Here C_h indicates the hidden channel size. Motivated by [7], instead of adopting self-attention by calculating the pair-wise similarity scores, we enable it via a depth-wise convolution operation with large kernels (LD-DWConv1d) to enhance the processing efficiency. After that, a point-wise convolution (PConv1d) is adopted to return to the original input space, followed by residual connection. Note that, to reduce the overall computational complexity, we use the group-convolution for PConv1d. The causal setting is adopted, *i.e.*, the padding operation is only applied along the past frames, and no future information is involved. Formally, the process of the LKCAB can be formulated as:

$$\mathbf{A}_i = \text{LK-DWConv1d}\left(\text{GELU}\left(\text{PConv1d}\left(\text{LN}\left(\mathbf{H}_{i-1}\right)\right)\right)\right), \quad (2)$$

$$\mathbf{V}_i = \text{PConv1d}\left(\mathbf{H}_{i-1}\right), \quad (3)$$

$$\mathbf{H}_i = \mathbf{H}_{i-1} + \text{PConv1d}\left(\mathbf{A}_i \otimes \mathbf{V}_i\right), \quad (4)$$

where "$\otimes$" denotes the element-wise multiplication operation.

3. MISCELLANEOUS CONFIGURATIONS

3.1. Loss Setups

We incorporate the reconstruction, adversarial, and perceptual losses for training. For the first term, we include the log-spectral amplitude loss $\mathcal{L}_a$, multi-resolution Mel loss $\mathcal{L}_m$, consistency loss $\mathcal{L}_{cons}$, phase loss $\mathcal{L}_p$, and RI loss $\mathcal{L}_{ri}$.

The amplitude loss evaluates the mean-square error (MSE) between $\left|\tilde{\mathbf{X}}\right|$ and $|\mathbf{X}|$ in the log-domain:

$$\mathcal{L}_a = \frac{1}{FT}\sum_{f,t}\left\|\log\left|\tilde{\mathbf{X}}_{f,t}\right| - \log\left|\mathbf{X}_{f,t}\right|\right\|_2^2. \quad (5)$$

Inconsistency can arise when the generated spectrum in the T-F domain is not necessarily equal to the STFT of it time-domain counterpart [8]. To mitigate this issue, the consistent spectrum is defined as $\hat{\mathbf{S}} = \text{STFT}\left(\text{iSTFT}\left(\tilde{\mathbf{S}}\right)\right)$, and consistency loss is given by:

$$\mathcal{L}_c = \frac{1}{FT}\sum_{f,t}\left(\left\|\mathcal{R}(\tilde{\mathbf{S}}_{f,t}) - \mathcal{R}\left(\hat{\mathbf{S}}_{f,t}\right)\right\|_2^2 + \left\|\mathcal{I}(\tilde{\mathbf{S}}_{f,t}) - \mathcal{I}(\hat{\mathbf{S}}_{f,t})\right\|_2^2\right). \quad (6)$$

Motivated by [9], we use multi-resolution Mel loss, which was reported to yield better performance over the single-resolution version, given by:

$$\mathcal{L}_{mel} = \frac{1}{FTS}\sum_{f,t}\sum_{s}\left\|\tilde{\mathbf{X}}_{f,t}^{mel,(s)} - \mathbf{X}_{f,t}^{mel,(s)}\right\|_1, \quad (7)$$

where $\left\{\tilde{\mathbf{X}}^{mel}, \mathbf{X}^{mel}\right\}$ are the estimated and target Mel spectra, respectively. $(\cdot)^{(s)}$ denotes the Mel spectrum under the s-th resolution scale. Here seven window sizes are adopted: {32, 64, 128, 256, 512, 1024, 2048}, and hop length set to window_length / 4. Besides, we use mel bin sizes $\{5, 10, 20, 40, 80, 160, 320\}$.

Motivated by [4], we employ an omnidirectional phase loss, as shown in Fig. 2. To be specific, a specially devised kernel $\mathcal{K} \in \mathbb{R}^{9\times3\times3}$ is applied to the estimated and target phase, to obtain the omnidirectional differential between the centering and neighboring phase bins:

$$\hat{\tilde{\mathbf{\Phi}}}_{est} = \tilde{\mathbf{\Phi}} * \mathcal{K}, \hat{\mathbf{\Phi}} = \mathbf{\Phi} * \mathcal{K}, \quad (8)$$

where "*" denotes the convolution operation, and $\left\{\hat{\tilde{\mathbf{\Phi}}}, \hat{\mathbf{\Phi}}\right\} \in \mathbb{R}^{9\times F\times T}$ are the convolved results for estimated and target phase, respectively. The phase loss can be calculated as:

$$\mathcal{L}_p = \frac{1}{FT}\sum_{f}\sum_{t}\left\|\hat{\tilde{\mathbf{\Phi}}} - \hat{\mathbf{\Phi}}\right\|_1. \quad (9)$$

We further generalize it into the RI loss. Concretely, we first decouple the magnitude and phase, then the omnidirectional operation is employed to extract the differential phase representation, *i.e.*,

$\left\{\hat{\tilde{\mathbf{\Phi}}}, \hat{\mathbf{\Phi}}\right\}$. The corresponding omnidirectional RI loss can be defined as:

$$\mathcal{L}_{ri} = \frac{1}{FT}\sum_{f}\sum_{t}\Bigg(\left\|\text{Rep}\left(\left|\tilde{\mathbf{X}}\right|\right)\cos\left(\hat{\tilde{\mathbf{\Phi}}}\right) - \text{Rep}\left(|\mathbf{X}|\right)\cos\left(\hat{\mathbf{\Phi}}\right)\right\|_1 + \left\|\text{Rep}\left(\left|\tilde{\mathbf{X}}\right|\right)\sin\left(\hat{\tilde{\mathbf{\Phi}}}\right) - \text{Rep}\left(|\mathbf{X}|\right)\sin\left(\hat{\mathbf{\Phi}}\right)\right\|_1\Bigg), \tag{10}$$

where $\text{Rep}\left(\cdot\right)$ denotes the tensor repeat operation, *i.e.*, $\mathbb{R}^{1\times F\times T} \rightarrow \mathbb{R}^{9\times F\times T}$. The overall reconstruction loss $\mathcal{L}_{recon}$ can be defined as:

$$\mathcal{L}_{recon} = \lambda_a\mathcal{L}_a + \lambda_c\mathcal{L}_c + \lambda_{mel}\mathcal{L}_{mel} + \lambda_p\mathcal{L}_p + \lambda_{ri}\mathcal{L}_{ri}, \tag{11}$$

where $\{\lambda_a, \lambda_c, \lambda_{mel}, \lambda_p, \lambda_{ri}\}$ are the corresponding weighting hyper-parameters, and set to $\{45.0, 20.0, 45.0, 50.0, 45.0\}$, respectively.

For adversarial loss, we incorporate the multi-period discriminator (MPD) [10], multi-resolution STFT discriminator (MRSTFTD) [5], and multi-band discriminator (MBD) [9], and the hinge loss is adopted to calculate the adversarial loss. For each sub-discriminator in MPD, the 1-D raw audio waveform is reshaped into 2-D format with period p, then processed through consecutive Conv2D layers and leaky ReLU for score computation. The periods are set to $\{2, 3\}$[2]. For MRD, three sub-discriminators process magnitude spectra via stacked Conv2d layers to calculate the discriminative score. The {window_size, hop_size, nfft} are set to $(128, 32, 128)$, $(256, 64, 256)$, $(512, 128, 512)$, $(1024, 256, 1024)$, and $(2048, 512, 2048)$, respectively. For MBD, we divide the overall spectrum into five band regions: {(0, 0.1), (0.1, 0.25), (0.25, 0.5), (0.5, 0.75), (0.75, 1.0)}. The {window_size, hop_size, nfft} are set to $(256, 64, 256)$, $(512, 128, 512)$, $(1024, 256, 1024)$, and $(2048, 512, 2048)$, respectively. The trainable parameters of the three discriminators are 3.4 M, 6.3 M, and 7.5 M, respectively. The weighting hyper-parameters of the adversarial and feature-matching losses are set to 1.0, 2.0, respectively.

Besides, the feature matching loss is also incorporated. For perceptual-based loss, to promote the performance on objective metrics, we include the PESQ loss[3] and UTMOS loss[4] for optimization. We also utilize the pre-trained SCOREQ model[5] and maximize the output similarity score between the estimation and target waveforms. Note that, to accelerate the network training, we only add the perceptual loss in the finetune stage, and the weighting hyper-parameters $\{\lambda_{pesq}, \lambda_{utmos}, \lambda_{scoreq}\}$ are set to $\{5.0, 5.0, 5.0\}$, respectively.

3.2. Dataset Setups

For codec training, we use the speech clips from LibriSpeech [11], DNS-Challenge [12], VCTK [13] and EARS [14]. Note that we did not use the CommonVoices [15] due to its relatively low quality. For noise set, we include DNS-Challenge noise set[6], WHAM! [16] and FSD50K [17]. For reverberation generation, we include the RIRs from Open SLR 28[7] and our synthesized 100 k RIR clips. To adapt to practical acoustic scenarios, we adopt the on-the-fly (OTF) training strategy, that is, we randomly combine noise and reverberation during the training process. For noise, the average SNR value is 15 dB, with the variance of 7.5 dB. The probability to include noise and reverberation are 0.15 and 0.15, respectively. We also include the multi-speaker case[8] with the overlap ratio randomly sampled in the range of $[0.5, 0.95]$, and the probability is set to 0.15. To mitigate the possible audio clip, we randomly rescale the waveform value from the range of $[0.218, 0.917]$. No other data augmentation strategies are adopted. All training clips are chunked to 2.0 second to stabilize the training.

3.3. Network Setups

For both STFT and iSTFT, the target sampling rate is 24 kHz. The window size is set to 30 ms, with 10 ms overlap between adjacent frames. 720-point FFT is adopted, leading to 361-D input features. Thus, the overall system latency is 10 + 20 = 30 ms, which satisfies the challenge rule. For network encoder, the input and hidden channel $\{C_{in}, C_h\}$ are set to $\{372, 372\}$, and $N_e = 6$ blocks are adopted. For the decoder, the input and hidden channel $\{C_{in}, C_h\}$ are set to $\{260, 360\}$, and $N_d = 6$ are adopted. For both sides, we set the kernel size of the LK-DWConv1d to 7, and the number of groups for PConv1d is set to 2 to reduce the computational complexity. For the quantization process, motivated by [9], we adopt the factorized quantizer, and the codebook dimension is set to 8. For 1 kbps and 6 kbps settings, $\{1, 6\}$ codebooks are utilized, respectively, and the codebook size is set to 1024. As a result, the average computational complexity of the encoder and decoder are around 390.18 MFlops (including 8.76 MFlops for quantization) and 295.12 MFlops. The trainable parameters of the encoder and decoder are 2.04 M and 1.48 M, respectively.

3.4. Training Setups

The training is based on the Pytorch-Lightning platform, and Two NVIDIA A100 are employed. The total batch size is 32, and we train the network for 1.5 M steps in total, where the discriminators are updated per three steps to reduce the GPU assumption. For the first 1.2 M steps, only reconstruction loss and adversarial loss are adopted. After that, we incorporate the perceptual loss in the remaining finetune stage. The AdamW optimizer [18] is employed, and the learning rate is initialized at 2e-4, with the exponential decay in the batch level, and the decay rate is set to 0.999996. Besides, the exponential moving average (EMA) strategy for generator update, and the decay rate is set to 0.999.

4. REFERENCES

[1] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[2] Haohe Liu, Xuenan Xu, Yi Yuan, Mengyue Wu, Wenwu Wang, and Mark D Plumbley, "Semanticodec: An ultra low bitrate semantic audio codec for general sound," *IEEE J. Sel. Top. Signal Process.*, 2024.

[3] Yi-Chiao Wu, Israel D Gebru, Dejan Marković, and Alexander Richard, "Audiodec: An open-source streaming high-fidelity neural audio codec," in *Proc. ICASSP*. IEEE, 2023, pp. 1–5.

[4] Andong Li, Tong Lei, Zhihang Sun, Rilin Chen, Erwei Yin, Xiaodong Li, and Chengshi Zheng, "Learning neural vocoder from range-null space decomposition," *arXiv preprint arXiv:2507.20731*, 2025.

[2] We empirically observe that more period settings can damage the performance in the light-weight audio codec design.

[3] https://github.com/audiolabs/torch-pesq

[4] https://github.com/tarepan/SpeechMOS/tree/main

[5] https://github.com/alessandroragano/scoreq

[6] https://github.com/microsoft/DNS-Challenge

[7] https://www.openslr.org/28/

[8] In practical synthesis, we only consider the 2-speakers remixing case.

[5] Yang Ai, Xiao-Hang Jiang, Ye-Xin Lu, Hui-Peng Du, and Zhen-Hua Ling, "Apcodec: A neural audio codec with parallel amplitude and phase spectrum encoding and decoding," *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 32, pp. 3256–3269, 2024.

[6] Yi Luo, Jianwei Yu, Hangting Chen, Rongzhi Gu, and Chao Weng, "Gull: A generative multifunctional audio codec," *arXiv preprint arXiv:2404.04947*, 2024.

[7] Qibin Hou, Cheng-Ze Lu, Ming-Ming Cheng, and Jiashi Feng, "Conv2former: A simple transformer-style convnet for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 12, pp. 8274–8283, 2024.

[8] Scott Wisdom, John R Hershey, Kevin Wilson, Jeremy Thorpe, Michael Chinen, Brian Patton, and Rif A Saurous, "Differentiable consistency constraints for improved deep speech enhancement," in *Proc. ICASSP*. IEEE, 2019, pp. 900–904.

[9] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar, "High-fidelity audio compression with improved rvqgan," *Proc. NeurIPS*, vol. 36, pp. 27980–27993, 2023.

[10] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Proc. NeurIPS*, vol. 33, pp. 17022–17033, 2020.

[11] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J. Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu, "LibriTTS: A Corpus Derived from LibriSpeech for Text-to-Speech," in *Proc. Interspeech*, 2019, pp. 1526–1530.

[12] Jianwei Yu, Hangting Chen, Yi Luo, Rongzhi Gu, Weihua Li, and Chao Weng, "Tspeech-ai system description to the 5th deep noise suppression (dns) challenge," in *Proc. ICASSP*. IEEE, 2023, pp. 1–2.

[13] Junichi Yamagishi, "English multi-speaker corpus for CSTR voice cloning toolkit," `Lhttp://homepages.inf.ed.ac.uk/jyamagis/page3/page58/page58.html/`, 2012.

[14] Julius Richter, Yi-Chiao Wu, Steven Krenn, Simon Welker, Bunlong Lay, Shinji Watanabe, Alexander Richard, and Timo Gerkmann, "Ears: An anechoic fullband speech dataset benchmarked for speech enhancement and dereverberation," in *Proc. Interspeech*, 2024, pp. 4873–4877.

[15] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," in *Proceedings of the 12th Conference on Language Resources and Evaluation (LREC 2020)*, 2020, pp. 4211–4215.

[16] Gordon Wichern, Joe Antognini, Michael Flynn, Licheng Richard Zhu, Emmett McQuinn, Dwight Crow, Ethan Manilow, and Jonathan Le Roux, "Wham!: Extending speech separation to noisy environments," in *Proc. Interspeech*, 2019, pp. 1368–1372.

[17] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font, and Xavier Serra, "FSD50K: an open dataset of human-labeled sound events," *arXiv preprint arXiv:2010.00475*, 2020.

[18] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

VOCODEC: AN EFFICIENT LIGHTWEIGHT LOW-BITRATES SPEECH CODEC

Leyan Yang[1,2,†], *Ronghui Hu*[1,2,†], *Yang Xu*[1,2,†], *Jing Lu*[1,2,*]

[1]Key Laboratory of Modern Acoustics, Nanjing University, Nanjing 210093, Jiangsu, China
[2]NJU-Horizon Intelligent Audio Lab, Horizon Robotics, Beijing 100094, China

ABSTRACT

Recent advancements in end-to-end neural speech codecs enable compressing audio at extremely low bitrates while maintaining high-fidelity reconstruction. However, low computational complexity and low latency remain crucial for real-time communication. In this paper, we propose VoCodec, an audio codec model featuring a computational complexity of only 349.29 M multiply-accumulate operations per second (MAC/s) and a latency of 30 ms. Additionally, we cascade a neural network for speech enhancement at the front end to extend its capabilities of noise reduction and dereverberation. Experimental results demonstrate that the two systems deliver superior performance across multiple evaluation metrics.

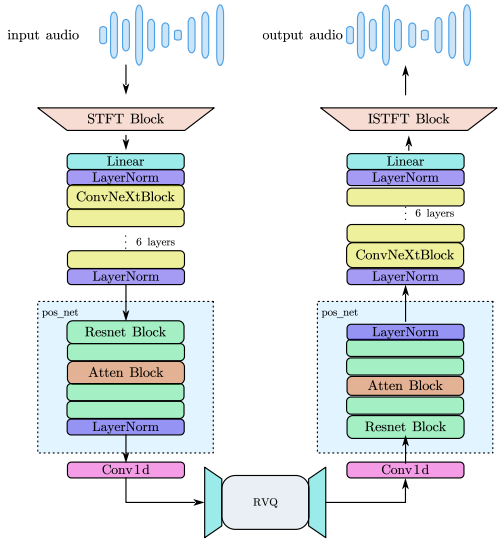
***Index Terms*—** audio codec, low computational resource, low bitrate, generative adversarial network, speech enhancement

1. INTRODUCTION

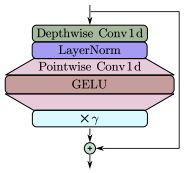
Recently, audio codec models have achieved significant progress in recovering high-quality speech at low bitrates [1]. However, existing models with excellent performance often suffer from two critical drawbacks: high computational complexity and non-causality, rendering them unsuitable for real-time communication [2, 3]. Track 1 of the 2025 Low-Resource Audio Codec (LRAC) Challenge focuses on audio compression that balances low latency, low bitrate, and high speech quality under constrained computational resources. Track 2 further takes the interference from noise and reverberation in real-world scenarios into account.

In this paper, we introduce our two systems submitted to the challenge. The system for Track 1, named VoCodec, is an audio codec model requiring low computational resources. Based on Vocos [4], we construct VoCodec's encoder and decoder. To reduce computational overhead and latency, audio codec is performed directly in the time-frequency domain and all upsampling and downsampling operations are eliminated in the encoder and decoder. For Track 2, UL-UNAS [5], a lightweight model for speech enhancement is cascaded at the front end of VoCodec to equip our whole system with speech enhancement capability.

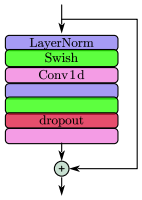
†Equal contribution.
*Corresponding author: lujing@nju.edu.cn



(a) VoCodec



(b) ConvNeXt Block



(c) ResNet Block

Fig. 1: The architecture of the proposed model

2. PROPOSED MODEL

2.1. Architecture

Based on the VQ-GANs framework [6], the architecture of the proposed VoCodec is depicted in Figure 1. Given a speech signal **x**, it is first passed through the Short-Time Fourier Transform (STFT). Then the logarithmic magnitude and phase of the complex spectrogram are concatenated along the frequency dimension. To reduce the computational complexity, a fully connected layer is employed to reduce the input's frequency dimension to 192.

The subsequent encoder follows the improved Vocos architecture in WavTokenizer[7], which consists of 6 stacked

ConvNeXt blocks and a PosNet module. 1D convolution with causal padding is used and the inverted structure from ConvNeXt is retained with an intermediate dimension of 216. A kernel size of 7 is employed for the depth-wise convolutional layers. The PosNet incorporates 4 basic ResNet blocks using a kernel size of 3 and a causal self-attention block is added after the second ResNet block. For the decoder, it is almost a mirror-symmetric structure of the encoder. However, to constrain the receiver-side's computational complexity, the inverted design from the ConvNeXt is removed, and the number of groups in the convolutions of the ResNet blocks is set to 2.

VoCodec's quantizer uses the Residual Vector Quantization (RVQ) strategy [8]. Following the improved RVQ proposed in DAC [2], the quantizer of our model is applied with 6 layers, each containing 1024 codewords. With an encoder frame rate of 100 Hz, this corresponds to 1 kbps per layer, and 6 kbps in total.

Since we perform audio codec in the time-frequency domain, intuitively, the multi-scale STFT discriminator [9] can further improve the quality of the output audio. A set of window lengths [128, 256, 512, 1024, 2048] is used, and the hop length is fixed to the window length / 4. Moreover, only this discriminator is employed throughout the training process, while other types (e.g. MSD and MPD) are not used.

2.2. Loss Functions

When training UL-UNAS, we apply the negative scale invariant SNR (SI-SNR) [10] loss and the power-compressed specturm loss as the loss functions.

For VoCodec, the generator loss $L_{generator}$ comprises three components: the multi-scale mel-spectrogram L1 loss [2] as the reconstruction loss L_{rec}, the adversarial loss L_g with the L1 feature matching loss L_{feat} involved, the same codebook L_{code} and commitment loss L_c in VQ-VAE [11] for codebook updates. The discriminator is trained separately with the adversarial loss L_d. Formally,

$$L_{rec} = \|\mathcal{M}(x) - \mathcal{M}(\hat{x})\|_1 \tag{1}$$

$$L_g = \|1 - D(\hat{x})\|_2^2 \tag{2}$$

$$L_{feat} = 2\sum_l \left\|D^l(x) - D^l(\hat{x})\right\|_1 \tag{3}$$

$$\begin{aligned} L_{\text{generator}} =& \lambda_{\text{rec}} L_{\text{rec}} + \lambda_g L_g + \lambda_{\text{feat}} L_{\text{feat}} \\ &+ \lambda_{\text{code}} \underbrace{\|\text{sg}[\mathbf{z}_e] - \mathbf{e}_k\|_2^2}_{L_{\text{code}}} + \lambda_c \underbrace{\|\mathbf{z}_e - \text{sg}[\mathbf{e}_k]\|_2^2}_{L_c} \end{aligned} \tag{4}$$

$$L_d = \|1 - D(x)\|_2^2 + \|D(\hat{x})\|_2^2 \tag{5}$$

where x and $\hat{x}$ denote the target and reconstructed speech, respectively, $\mathcal{M}(\cdot)$ is the mel-spectrogram transform, $D(\cdot)$ is the discriminator output, $D^l(\cdot)$ represents the feature map of the l-th discriminator layer, $\mathbf{z}_e$ is the quantizer output, and $\mathbf{e}_k$ is the codebook vector. During training, the mel-spectrograms are computed with multiple window lengths of [32, 64, 128, 256, 512, 1024, 2048] and a fixed hop length set to window length / 4. Meanwhile, different mel bin sizes of [5, 10, 20, 40, 80, 160, 320] are employed.

Finally, we assign loss weights of 15.0 for the multi-scale mel-spectrogram loss, 1.0 for the feature matching loss, 2.0 for the adversarial loss, and 1.0, 0.25 for the codebook and commitment loss, respectively.

Table 1: Latency and computational complexity of the Track 1 baseline system.

	Transmit Side		Receive Side	Overall
	Encoder	RVQ	Decoder	
Buffering Latency (ms)	10	0	0	10
Algorithmic Latency (ms)	0	0	20	20
Compute Complexity (MMACs)	194.56	1.96	144.82	349.29

Table 2: Latency and computational complexity of the system for Track 2.

	Transmit Side			Receive Side	Overall
	SE	Encoder	RVQ	Decoder	
Buffering Latency (ms)	0	10	0	0	10
Algorithmic Latency (ms)	20	0	0	20	40
Compute Complexity (MMACs)	935.36	194.56	1.96	144.82	1284.66

2.3. Two Training Stages

We first train the speech enhancement network UL-UNAS and the codec model VoCodec independently. After their training is completed, we use UL-UNAS as the front end to perform noise suppression and dereverberation on the input audio. Subsequently, we freeze the parameters of UL-UNAS and only update the parameters of the entire codec model to enable it to compensate for the spectral distortion caused by the enhancement network.

3. EXPERIMENTAL AND RESULTS

3.1. Training Data Preparation

All training data follow the cleaning and preprocessing procedures defined in the baseline of the Challenge[1]. Considering the attention module in the codec model, we extract 3-second speech segments for training VoCodec.

During the training process of UL-UNAS and the final cascade system, each speech sample is combined with background noise, where signal-to-noise ratio (SNR) is uniformly distributed between -5 dB and 30 dB. In addition, reverberation is randomly introduced, and the final training target is speech signals with early reverberation.

3.2. Implementation Details

STFT is computed using a square root Hanning window of a length of 30 ms, a hop length of 10 ms, and an FFT length of 720, resulting in a buffering latency of 10 ms and an algorithmic latency of 20 ms due to the inverse STFT processing.

[1] https://github.com/cisco-open/lrac_data_generation

Table 3: Performance Comparison on the Open Test Set for Track 1

Bitrate	Model	Condition	ScoreQ-ref ↓	UTMOS ↑	Sheet-SSQA ↑	PESQ ↑	Audiobox AE-CE ↑
6 kbps	Baseline	Clean	0.35	3.23	3.84	2.67	5.28
		Noisy	0.82	2.76	3.12	1.81	4.37
		Reverb	1.13	1.32	2.22	1.18	3.43
	VoCodec	Clean	**0.17**	**3.73**	**4.22**	**3.20**	**5.66**
		Noisy	**0.70**	**3.10**	**3.43**	**2.03**	**4.82**
		Reverb	**0.94**	**1.55**	**2.80**	**1.21**	**3.98**
1 kbps	Baseline	Clean	1.15	1.44	1.84	1.15	3.90
		Noisy	1.29	1.33	1.72	1.11	3.40
		Reverb	1.36	1.26	1.85	1.07	2.94
	VoCodec	Clean	**0.40**	**3.24**	**3.55**	**1.95**	**5.31**
		Noisy	**0.83**	**2.67**	**2.93**	**1.56**	**4.43**
		Reverb	**1.10**	**1.48**	**2.19**	**1.17**	**3.59**

Table 4: Performance Comparison on the Open Test Set for Track 2

Bitrate	Model	Condition	ScoreQ-ref ↓	UTMOS ↑	Sheet-SSQA ↑	PESQ ↑	Audiobox AE-CE ↑
6 kbps	Baseline	Clean	0.43	2.97	3.55	2.13	2.97
		Noisy	0.75	2.56	2.92	1.73	4.60
		Reverb	0.92	1.79	2.67	1.29	4.25
	VoCodec	Clean	**0.18**	**3.74**	**4.21**	**3.06**	**5.68**
		Noisy	**0.50**	**3.26**	**3.62**	**2.19**	**5.00**
		Reverb	**0.88**	**2.02**	**2.70**	**1.38**	4.20
1 kbps	Baseline	Clean	1.01	1.37	2.07	1.21	3.96
		Noisy	1.15	1.35	1.95	1.18	3.70
		Reverb	1.12	1.32	2.43	1.15	3.55
	VoCodec	Clean	**0.41**	**3.21**	**3.50**	**1.92**	**5.29**
		Noisy	**0.68**	**2.81**	**3.00**	**1.63**	**4.75**
		Reverb	**1.04**	**1.75**	2.20	**1.26**	**3.95**

As shown in Table 1, our proposed VoCodec comprises 3.47 M parameters and has a computational complexity of 349.29 MMAC/s [2], with the receiver-side computation accounting for only 144.82 MMAC/s. In Track 2, we adopt the same network architecture from UL-UNAS [5], and scale up the number of intermediate channels to [48, 96, 108, 108, 64], which results in a computational complexity of 935.36 MMAC/s. The whole system has a computational complexity of 1.28 GMAC/s with 5.34 M parameters shown in Table 2. The total latency is 50 ms, of which 20 ms is attributed to the additional look-ahead in UL-UNAS.

We train UL-UNAS and VoCodec independently on 8 NVIDIA RTX 4090 GPUs. The batch size for UL-UNAS is set to 4 per GPU, while that for VoCodec is 24 per GPU. UL-UNAS and VoCodec are trained for 400 and 1000 epochs, with 1250 and 500 iterations per epoch, respectively. During training, we use the AdamW optimizer [12] and employ a linear warmup scheduler followed by cosine annealing. In the joint training phase, we adopt the same configuration as used in the training of VoCodec, and conduct the training process for a total of 500 epochs.

Meanwhile, we employ a systematic strategy to select the final checkpoint of the model. The validation objective metrics are evaluated at regular intervals during training and the checkpoint with the best performance is selected.

3.3. Results

Evaluation on the test set is conducted using the official metrics provided by the Challenge. Scoreq-ref [13], UTMOS [14], Audiobox AE-CE [15], PESQ [16], and Sheet-SSQA [17, 18] are selected to compare the quality and naturalness of speech recovered by the decoder of the codec.

The experimental results are summarized in Table 3 and Table 4. It can be seen that our two systems outperform the official baseline models[3] across all metrics, particularly on the

[2]The computational complexity is calculated by ptflops: `https://github.com/tel-0s/ptflops`.

[3]`https://github.com/cisco-open/espnet/tree/master/egs2/lrac`

clean and noisy test sets.

4. CONCLUSION

This paper introduces VoCodec, an efficient lightweight speech codec, and our two systems in the LRAC 2025 Challenge. Experiments show that our systems achieve superior performance over the baseline models.

5. REFERENCES

[1] Yiwei Guo, Zhihan Li, Hankun Wang, Bohan Li, Chongtian Shao, Hanglei Zhang, Chenpeng Du, Xie Chen, Shujie Liu, and Kai Yu, "Recent advances in discrete speech tokens: A review," *CoRR*, vol. abs/2502.06490, February 2025.

[2] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar, "High-fidelity audio compression with improved RVQGAN," in *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[3] Detai Xin, Xu Tan, Shinnosuke Takamichi, and Hiroshi Saruwatari, "Bigcodec: Pushing the limits of low-bitrate neural speech codec," *arXiv preprint arXiv:2409.05377*, 2024.

[4] Hubert Siuzdak, "Vocos: Closing the gap between time-domain and fourier-based neural vocoders for high-quality audio synthesis," *arXiv preprint arXiv:2306.00814*, 2023.

[5] Xiaobin Rong, Dahan Wang, Yuxiang Hu, Changbao Zhu, Kai Chen, and Jing Lu, "Ul-unas: Ultra-lightweight u-nets for real-time speech enhancement via network architecture search," 2025.

[6] Patrick Esser, Robin Rombach, and Bjorn Ommer, "Taming transformers for high-resolution image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 12873–12883.

[7] Shengpeng Ji, Ziyue Jiang, et al., "Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling," *arXiv preprint arXiv:2408.16532*, 2024.

[8] Biing-Hwang Juang and A. Gray, "Multiple stage vector quantization for speech coding," in *ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1982, vol. 7, pp. 597–600.

[9] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[10] Jonathan Le Roux, Scott Wisdom, Hakan Erdogan, and John R. Hershey, "Sdr – half-baked or well done?," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 626–630.

[11] Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu, "Neural discrete representation learning," in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[12] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2017.

[13] Alessandro Ragano, Jan Skoglund, and Andrew Hines, "Scoreq: Speech quality assessment with contrastive regression," in *Advances in Neural Information Processing Systems*. 2024, vol. 37, pp. 105702–105729, Curran Associates, Inc.

[14] Takaaki Saeki and Detai Xin and Wataru Nakata and Tomoki Koriyama and Shinnosuke Takamichi and Hiroshi Saruwatari, "UTMOS: UTokyo-SaruLab System for VoiceMOS Challenge 2022," in *Interspeech 2022*, 2022, pp. 4521–4525.

[15] Andros Tjandra, Yi-Chiao Wu, Baishan Guo, John Hoffman, Brian Ellis, Apoorv Vyas, Bowen Shi, Sanyuan Chen, Matt Le, Nick Zacharov, Carleigh Wood, Ann Lee, and Wei-Ning Hsu, "Meta audiobox aesthetics: Unified automatic quality assessment for speech, music, and sound," 2025.

[16] 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.

[17] Wen-Chin Huang, Erica Cooper, and Tomoki Toda, "SHEET: A Multi-purpose Open-source Speech Human Evaluation Estimation Toolkit," in *Proc. Interspeech*, 2025, pp. 2355–2359.

[18] Wen-Chin Huang, Erica Cooper, and Tomoki Toda, "Mos-bench: Benchmarking generalization abilities of subjective speech quality assessment models," 2024.

LOW RESOURCE AUDIO CODEC CHALLENGE TRACK2: DENOISING CODEC

Haoran Zhao, Zixiang Wan, Guochang Zhang, Runqiang Han, Jianqiang Wei

Anker Innovations, Beijing, China

ABSTRACT

We propose a frequency-domain *Denoising Codec* for the 2025 Low-Resource Audio Codec (LRAC) Challenge that jointly performs speech coding and noise suppression under strict constraints on complexity, latency, and bitrate. By integrating enhancement into the coding pipeline and employing residual vector quantization (RVQ), the system allocates bits to perceptually important speech components while reducing the noise. A three-stage training process combines spectral reconstruction with adversarial objectives to ensure stable optimization and high-quality output. Experiments across clean, noisy, and reverberant conditions demonstrate consistent improvements in both coding fidelity and robustness.

***Index Terms*—** speech codec, noise suppression, low resource, LRAC

1. INTRODUCTION

Neural audio codecs are emerging as powerful alternatives to traditional speech coders such as AMR-WB and Opus, delivering improved perceptual quality and flexible bitrate adaptation. Recent advances, SoundStream [1], Encodec [2], and DAC [3] employ autoencoder-based architectures with RVQ and adversarial training, achieving high-quality reconstruction at low bitrates.

In parallel, neural speech enhancement has advanced rapidly. Architectures such as U-Net [4], DCCRN [5], and DeepFilterNet [6] demonstrate robust noise suppression across diverse acoustic environments. Leveraging convolutional encoder–decoder backbones, recurrent layers, and attention mechanisms, these models effectively disentangle clean speech from noise and reverberation.

However, most codecs and enhancement systems are designed and optimized independently: codecs focus primarily on compression efficiency and reconstruction fidelity, while enhancement models target noise reduction and dereverberation. Under realistic constraints on complexity, latency, and bitrate, separating enhancement from coding can be suboptimal. A unified approach enables efficient bit allocation for perceptual speech quality and effective noise suppression.

The 2025 LRAC Challenge provides an ideal platform for such integrated solutions, emphasizing neural speech codecs operating under realistic noise and reverberation with strict limits on complexity, latency, and bitrate. It encourages unified designs that jointly address speech coding and enhancement within a low-resource framework. Motivated by this, we propose a frequency-domain *Denoising Codec* jointly optimized for noise suppression and speech coding.

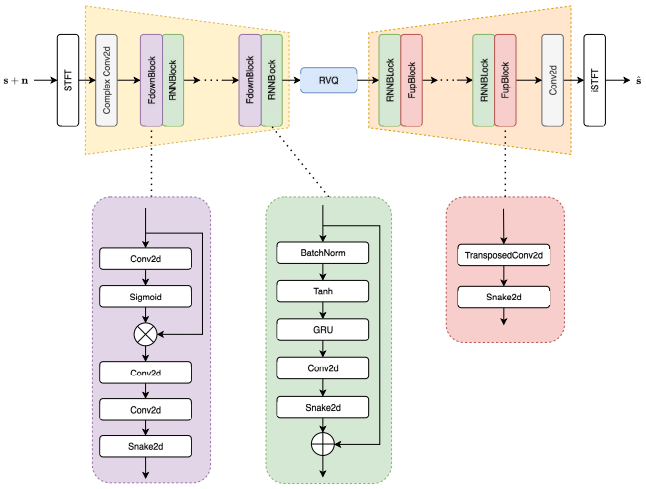


Fig. 1. The proposed model architecture.

2. METHOD

2.1. Architecture

The proposed end-to-end *Denoising Codec* is illustrated in Figure 1. It comprises an encoder, an RVQ module [1, 2], and a decoder, and operates entirely in the frequency domain. The noisy input signal is first transformed into a spectrogram via STFT, which is processed by the encoder to generate downsampled latent vectors; these vectors are quantized by RVQ and then reconstructed by the decoder through upsampling. The resulting spectrogram is finally converted back to the time domain using iSTFT. Noise suppression is implicitly achieved throughout the encoding–decoding process.

The encoder consists of a complex convolutional layer followed by 4 FdownBlocks and RNNBlocks, which perform downsampling, feature extraction, and implicit denoising.

Table 1. Evaluation results for different bitrates and acoustic conditions.

Bitrate	Method	Clean					Noisy					Reverb				
		sheet ssqa	scoreq ref	audiobox AE_CE	utmos	pesq	sheet ssqa	scoreq ref	audiobox AE_CE	utmos	pesq	sheet ssqa	scoreq ref	audiobox AE_CE	utmos	pesq
1kbps	Baseline	2.07	1.01	3.96	1.37	1.21	1.95	1.15	3.70	1.35	1.18	2.43	1.12	3.55	1.32	1.15
	Proposed	3.44	0.43	5.23	3.18	2.07	3.11	0.61	4.9	2.94	1.8	2.27	0.95	4.32	2.05	1.38
6kbps	Baseline	3.55	0.43	5.25	2.97	2.13	2.92	0.75	4.60	2.56	1.73	2.67	0.92	4.25	1.79	1.29
	Proposed	4.22	0.18	5.62	3.80	3.34	3.78	0.41	5.17	3.47	2.41	2.96	0.74	4.62	2.30	1.61

Each FdownBlock includes two 1×1 convolutions and one downsampling convolution, incorporates a gating mechanism to enhance feature extraction, and adopts a Snake2D activation [7] to improve harmonic structure modeling. Each RNNBlock contains batch normalization, a GRU, and a 1×1 convolution, with residual connections to preserve gradient flow. The decoder mirrors the encoder with 4 RNNBlocks and FupBlocks for upsampling, followed by a final convolutional layer for spectrogram reconstruction. Due to computational constraints, each FupBlock contains only a transposed convolution and a Snake2D activation.

The model is trained using a loss function that combines complex spectrogram loss, multi-scale Mel-spectrogram loss, and a GAN-based loss, where Multi-Period (MPD) and Multi-Resolution (MRD) discriminators [8] are employed to capture both fine-grained temporal details and spectral characteristics.

2.2. Training Stages

We employ a three-stage training pipeline. (1) A quantizer-free encoder–decoder model is trained exclusively for the denoising task, optimized solely with a complex spectral loss to establish a clean and stable representation space for subsequent quantization. (2) Quantizer Integration: A quantizer is then incorporated into the pre-trained denoising model, and the entire system is jointly optimized for both denoising and codec objectives while still employing the complex spectral loss. This staged integration stabilizes training by initializing the quantizer within a well-structured representation space, thereby maintaining denoising performance while enabling effective quantization. (3) Perceptual Fine-tuning: Finally, the model is fine-tuned to enhance the perceptual audio quality by replacing the loss function with a multi-scale Mel-spectrogram reconstruction loss and introducing adversarial objectives via MPD and MRD. This combination further improves the naturalness and fidelity of the reconstructed audio.

3. EXPERIMENTS

3.1. Datasets

The 2025 LRAC Challenge provides training datasets comprising speech, noise, and room impulse response (RIR) subsets. All datasets are first resampled to 24 kHz, followed by curation to form finalized training subsets. Specifically for the noise dataset, we utilize a pre-trained audio understanding model to predict audio labels, and further filter out "dirty" data samples bearing speech labels.

To further enhance the generalization capability of the model, training data is synthesized online using randomly sampled parameters at each training step. The data augmentation is detailed as follows: Noisy conditions are simulated by mixing speech and noise at a probability of 0.75, using a signal-to-noise ratio (SNR) uniformly sampled from the range of -5 to 15 dB. Reverberation is simulated by convolving the speech signal with a RIR at a probability of 0.4. For the corresponding target speech, the RIR undergoes truncation commencing 1 ms after its peak amplitude prior to convolution.

3.2. Implementation Details

The proposed model has a total computational complexity of 2595M FLOPs and 3.9M parameters. Specifically, the encoder together with the RVQ module accounts for 1997M FLOPs and 2.5M parameters, while the decoder requires 598M FLOPs and 1.4M parameters. The system is designed for a sampling rate of 24kHz, with a frame length of 720 samples and a frame shift of 312 samples. No future frames are utilized, resulting in an algorithmic latency of only 30ms. The RVQ module contains 6 codebooks, each with a size of 8192 entries (equivalent to 13 bits) and a vector dimension of 16. During inference, the RVQ can dynamically select between using 1 to 6 codebooks, enabling bitrate scalability from 1kbps to 6kbps.

Due to computational constraints, the channel configurations of the encoder and decoder are asymmetric. The FdownBlocks in the encoder have channel sizes of $[48, 144, 192, 288]$ with strides $[6, 5, 4, 3]$, while the upsampling layers in the decoder have channel sizes of $[24, 48, 124, 288]$.

The MPD employs period settings of $[2, 3, 5, 7, 11]$, while the MRD adopts $[3072, 1536, 768, 384, 206, 126, 78]$ as window sizes [9]. Additionally, The generator is trained with a learning rate of 3×10^{-4}, while the discriminator uses 1×10^{-4}. The Adam optimizer is employed throughout all training stages.

3.3. Results

Our evaluation employs Versa [10], the official evaluation toolkit recommended by the 2025 LRAC Challenge, which provides standardized implementations of metrics such as *sheet_ssqa*, *score_ref*, *audiobox AE_CE*, *UTMO*, and *PESQ*. Experiments are conducted under three acoustic conditions: clean, noisy, and reverberant. Leveraging the RVQ module, which supports variable-bitrate operation by selectively discarding codebooks during inference, we further assess the model's performance at 1 kbps and 6 kbps.

The evaluation results are summarized in Table 1, where the proposed method demonstrates significant improvements over the baseline across all metrics under all three acoustic conditions and at both 1 kbps and 6 kbps bitrates. The evaluation across different acoustic conditions reveals distinct characteristics of *Denoising Codec*. In clean scenarios, the model demonstrates superior performance in speech compression and reconstruction. For noisy and reverberant conditions, it exhibits strong robustness by effectively suppressing background noise and reverberation. However, the audio quality under reverberant conditions is somewhat compromised compared to other scenarios.

4. CONCLUSION

We presented a unified *Denoising Codec* that integrates speech coding and noise suppression in the frequency domain, enabling scalable bitrate via RVQ and delivering high perceptual quality across diverse acoustic conditions. The staged training strategy stabilizes optimization and enhances overall performance while meeting strict low-resource constraints. Future work will focus on further improving reconstruction quality under strict low-resource constraints.

5. REFERENCES

[1] Neil Zeghidour, Anatoly Luebs, Mohammad Omran, Jan Skoglund, and Marco Tagliasacchi, "Soundstream: An end-to-end neural audio codec," in *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2021, vol. 30, pp. 495–507.

[2] Alexandre Défossez, Neil Zeghidour, Nicolas Usunier, and Gabriel Synnaeve, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[3] Fabian Mentzer, Eirikur Agustsson, Michael Tschannen, Radu Timofte, Tim Salimans, and Marco Tagliasacchi, "Fully neural audio coding using variational autoencoders," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14266–14276.

[4] Andreas Jansson, Eric Humphrey, Nicola Montecchio, Rachel Bittner, Aparna Kumar, and Till Weyde, "Singing voice separation with deep u-net convolutional networks," in *18th International Society for Music Information Retrieval Conference (ISMIR)*, 2017.

[5] Yongxin Hu, Yue Wang, Yao, et al., "Dccrn: Deep complex convolution recurrent network for phase-aware speech enhancement," in *INTERSPEECH*, 2020, pp. 2472–2476.

[6] Hendrik Schröter, Tim Gburrek, Thomas Appel, and Andreas Maier, "Deepfilternet: Lightweight speech enhancement for full-band audio," *arXiv preprint arXiv:2205.07847*, 2022.

[7] Sang gil Lee, Wei Ping, Boris Ginsburg, Bryan Catanzaro, and Sungroh Yoon, "Bigvgan: A universal neural vocoder with large-scale training," 2023.

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

[9] Julian D Parker, Anton Smirnov, Jordi Pons, CJ Carr, Zack Zukowski, Zach Evans, and Xubo Liu, "Scaling transformers for low-bitrate high-quality speech coding," 2024.

[10] Jiatong Shi, Hye jin Shim, Jinchuan Tian, Siddhant Arora, Haibin Wu, Darius Petermann, Jia Qi Yip, You Zhang, Yuxun Tang, Wangyou Zhang, Dareen Safar Alharthi, Yichen Huang, Koichi Saito, Jionghao Han, Yiwen Zhao, Chris Donahue, and Shinji Watanabe, "Versa: A versatile evaluation toolkit for speech, audio, and music," 2025.

EFFICIENT REAL-TIME AUDIO CODEC WITH INTEGRATED SPEECH ENHANCEMENT TECHNIQUES

Weihao Xiong, Congxin Zhang, Xinming Yan, Qingbo Huang

ByteDance

ABSTRACT

This paper presents our submission model for the 2025 Low-Resource Audio Codec (LRAC) Challenge, which is an efficient real-time audio codec with integrated speech enhancement techniques. The model is composed of three primary components: an encoder, a quantizer, and a decoder. To achieve better performance, the encoder module encodes the noisy audio into clean embeddings with the constraint of a pretrained codebook. Then the decoder decoders the clean embedding to audio wavforms. This system operates with a 50ms latency and a computational complexity of 2.68 GFLOPS, with the decoder contributing 0.58 GFLOPS.

***Index Terms*—** Speech Enhancement, audio codec

1. OVERVIEW OF OUR SYSTEM

This model is primarily built upon advancements from previous codecs and vocoders [1, 2, 3, 4]. The model operates in the frequency domain, where an STFT (Short-Time Fourier Transform) is applied before the encoder and an iSTFT (Inverse STFT) is performed after the decoder. Within the model, only the magnitude of the STFT is processed.

The STFT and iSTFT processes utilize a 50ms window length (1200 samples at 24000hz) and a 12.5ms hop size (300 samples), resulting in a total latency of 50ms. The 1kbps codebook consists of 5,792 numbers, producing a bitrate of 1kbps, calculated as $\frac{1}{12.5} \cdot \log_2(5792)$. For the 6kbps configuration, the model extends the 1kbps codebook with six additional codebooks, each containing 1,024 numbers. This setup results in a total bitrate of 5.8kbps, computed as $\frac{1}{12.5} \cdot (\log_2(5792) + \log_2(1024) * 6))$.

The model includes a total of 11.96 million trainable parameters and has a computational complexity of 1.34 GMacs (2.68 GFLOPS)."

2. ENCODER

The encoder module, referred to as the **FullBandEncoder**, processes input data through three main components: an input feature extractor (`in_fc`), a series of eight sequential blocks (`blocks`), and an output projection head (`out_head`). The overall architecture has **8.2M parameters** and a computational complexity of **662.66MMACs (Million Multiply-Accumulate operations)**, which accounts for **50.829% of the total parameters** and **39.592% of the total MACs** in the network. The encoder aims to capture temporal and spatial dependencies in sequential data while maintaining high computational efficiency.

2.1. Input Feature Extraction (`in_fc`)

The first component of the encoder is the input feature extractor (`in_fc`), which processes the raw input data. This module has **1.34M parameters** and contributes **108.31MMACs** (**8.294% Params, 6.471% MACs**). It consists of the following layers:

- **ChannelNormalization**: This normalization layer stabilizes the input data by re-scaling the channel distributions. As a computationally free module (*0% Params, 0% MACs*), it improves model training and convergence behavior.
- **Conv1d Sequential Block**:
  - **ConstantPad1d**: Padding is applied (**padding=(2, 0)**) to ensure dimensional alignment prior to convolution. This layer does not add any computational cost.
  - **Conv1d**: A convolutional layer with **1.34M parameters**, configured with **602 input channels** (in consistancy with stft freq bins), **740 output channels**, a **kernel size of 3**, and a **stride of 1**. This operation extracts local features while increasing dimensionality to match the hidden size.

2.2. Sequential Blocks (`blocks`)

Motivated by [5], the second component consists of eight **Large Kernel Convolution-Style Attention Blocks (LKCABs)**, which are implemented through a `ModuleList`. Each block has **833.98k parameters** and a computational complexity of **67.37MMACs** (**5.170% Params, 4.025%**

MACs). Collectively, the blocks account for the primary processing in the encoder.

Each `LKCAB` focuses on capturing temporal and spatial dependencies through its **attention module**, **value module**, and **output projection layer**. These are described in detail below:

2.2.1. Attention Module (`attn`)

The attention module processes sequential data through a combination of normalization, convolutional operations, and non-linear activations in the following pipeline:

- **ChannelNormalization**: A normalization layer prepares the input channels for processing without adding to the computational complexity (*0% Params, 0% MACs*).
- **First Conv1d Sequential Block**:
  - **ConstantPad1d**: Padding ensures consistent input dimensions without adding parameters or MACs.
  - **Conv1d**: This convolutional layer has **275.28k parameters** and operates on **740 input and output channels**. It uses a **kernel size of 1**, **stride of 1**, and **groups=2**, enabling separable convolution for efficient feature extraction. It accounts for **22.24MMACs**, or **1.706% Params, 1.329% MACs**.
- **GELU Activation**: A **GELU (Gaussian Error Linear Unit)** introduces non-linearity into the pipeline. GELU has no parameters and a negligible computational cost of **59.94KMACs** (**0.004% MACs**), but it provides smooth and continuous activation for improved gradient flow and feature learning.
- **Second Conv1d Sequential Block**:
  - **ConstantPad1d**: Padding aligns the input sequence for the subsequent convolutional layer.
  - **Conv1d**: A depthwise convolutional layer configured with **8.14k parameters**. It operates on **740 input and output channels**, with a **kernel size of 9**, **stride of 1**, and **groups=740**, allowing each channel to be processed independently. This layer consumes **599.4KMACs** (**0.050% Params, 0.036% MACs**) and captures channel-specific features over a larger receptive field.

The attention module has a total of **283.42k parameters** and contributes **22.9MMACs** (**1.757% Params, 1.368% MACs**). By combining separable convolutions, non-linear activation, and depthwise operations, it efficiently extracts both local and global features from sequential data.

2.2.2. Value Module (`v`)

The value module processes the input in parallel to the attention module and comprises:

- **ConstantPad1d**: Padding ensures dimensional consistency without adding computational cost (*0% Params, 0% MACs*).
- **Conv1d**: A convolutional layer with **275.28k parameters** configured identically to the first Conv1d layer in the attention module (**740 input and output channels**, **kernel size 1**, **stride 1**, **groups=2**). It contributes **22.24MMACs**, or **1.706% Params, 1.329% MACs**.

The combined output of the attention module and the value module is connected by a **residual connection**, ensuring stable training and strong information flow.

2.2.3. Projection Layer (`proj`)

The output projection layer refines features by projecting the channels back to the hidden dimensionality. This layer includes:

- **ConstantPad1d**: Padding is applied to preserve spatial consistency.
- **Conv1d**: Another convolutional layer with **275.28k parameters** identical to those in the value module. It contributes **22.24MMACs** (**1.706% Params, 1.329% MACs**).

2.3. Output Projection Head (`out_head`)

The final component of the encoder processes the output of the eight sequential blocks to produce the desired feature representation. The output projection head has **189.95k parameters** and contributes **15.37MMACs** (**1.177% Params, 0.918% MACs**). It consists of:

- **ConstantPad1d**: Padding ensures dimensional alignment.
- **Conv1d**: A convolutional layer with **740 input channels**, projecting down to **256 output channels**. It uses a **kernel size of 1** and **stride of 1**. This operation reduces dimensionality while retaining relevant features for downstream tasks.

2.4. Overall Design and Applications

The encoder module leverages **channel normalization**, **separable convolutions**, **depthwise operations**, and **residual connections** to achieve efficient and expressive feature extraction. Its modular design makes it suitable for sequential data tasks such as **audio signal processing**. By balancing computational complexity (**662.66MMACs**) and parameter

count (**8.2M**), the encoder strikes an excellent trade-off between performance and resource efficiency.

3. QUANTIZATION

As previously mentioned, both the 1kbps and 6kbps configurations share a base codebook consisting of 5,792 entries, which is quantized by a vector quantizer (FactorizedVectorQuantize) [6] with a computational complexity of 381.07 MMacs.

The 6kbps configuration introduces additional codebooks, which are quantized in two groups using another quantizer called SimVQ1D[7] with a computational complexity of approximately 1.2 MMacs.

The encoder always produces encodings at 6kbps, while in the training phase, the quantizer randomly drops the outputs from the second codebook. During inference, the decoder reconstructs the waveform using the specified codebook configuration.

4. DECODER

The decoder shares the same fundamental building blocks as the encoder but adopts two distinct configurations depending on the training stage. During stage 1, the hidden dimension is set to 600 to improve the performance of codebook training. In stage 2 and during inference, the hidden dimension is reduced to 530 to prioritize computational efficiency while maintaining strong performance.

To further address computational complexity constraints, the block dimensionality in the decoder is fixed at 530, and the number of blocks is limited to 6. This optimization reduces the computational complexity of the decoder to **296.96 MMACs**, meeting the competition's requirements while ensuring robust performance.

Apart from the hidden dimension and the number of blocks, the input dimension of the decoder is set to 256, which differs from that of the encoder.

For waveform reconstruction, the decoder incorporates multiple head modules to recover both the magnitude and phase components of the frequency-domain signal. These include:

- **Mag Head**: Outputs the magnitude of the STFT.
- **R Head and I Head**: Jointly output the phase information of the STFT.

Each head (Mag Head, R Head, and I Head) consists of a `ConstantPad1d` layer followed by a `Conv1d` layer. These modules share the same structure, with a 530-channel input, a 601-channel output, and a kernel size and stride of 1.

The final step in the decoder is the `ISTFT` (Inverse Short-Time Fourier Transform) layer, which reconstructs the time-domain signal from its frequency-domain representation. This layer introduces no additional trainable parameters or computational overhead, ensuring an efficient mapping back to the audio waveform.

5. TRAINING SETUP

We trained the model using data provided by the LRAC challenge requirements, with augmentation applied on the fly during training. The augmentation module simulates audio degradation by adding noise, reverberation, and other artifacts. It generates corresponding pairs of noisy and clean audio for training purposes.

The training process consists of two stages. In the first stage, the model is trained using clean speech as both input and output. During this phase, we aim to simultaneously learn the codebooks for both 1kbps and 6kbps configurations. In the second stage, the quantization module is frozen, and the encoder and decoder are retrained using noisy speech as the input and clean speech as the output.

This approach allows the codebooks to constrain the encoder to focus exclusively on encoding clean speech. On one hand, the denoising functionality is embedded within the encoder, which is a larger component of the model. On the other hand, the codebooks are specifically designed to support clean speech transmission, making them highly effective for this task.

Several loss functions are employed during the training process, including the multi-resolution STFT loss, multi-resolution Mel loss, and phase loss. Additionally, for adversarial loss, we utilize the Multi-Period Discriminator (MPD), Multi-Resolution STFT Discriminator (MRSTFTD), and Multi-Band Discriminator (MBD) to improve audio quality and realism.

6. REFERENCES

[1] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019.

[2] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar, "High-fidelity audio compression with improved rvqgan," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[3] Andong Li, Tong Lei, Zhihang Sun, Rilin Chen, Erwei Yin, Xiaodong Li, and Chengshi Zheng, "Learning neural vocoder from range-null space decomposition," *arXiv preprint arXiv:2507.20731*, 2025.

[4] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2021.

[5] Ming-Ming Cheng Qibin Hou, Cheng-Ze Lu and Jiashi Feng, "Conv2former: A simple transformer-style convnet for visual recognition," in *IEEE Trans. Pattern Anal. Mach. Intell., vol. 46, no. 12, pp. 8274– 8283*, 2024.

[6] Chi-Min Chan Xinsheng Wang Xu Tan Jiahe Lei Yi Peng Haohe Liu Yizhu Jin Zheqi Dai Hongzhan Lin Jianyi Chen Xingjian Du Liumeng Xue Yunlin Chen Zhifei Li Lei Xie Qiuqiang Kong Yike Guo Wei Xue Zhen Ye, Xinfa Zhu, "Llasa: Scaling train-time and inference-time compute for llama-based speech synthesis," *arXiv preprint arXiv:2502.04128*, 2025.

[7] Yifei Xin Zhihua Xia Linli Xu Yongxin Zhu, Bocheng Li, "Addressing representation collapse in vector quantized models with one linear layer," *arXiv preprint arXiv:2411.02038*, 2025.

ENHANCE-NANOCODEC: ENHANCEMENT NEURALAUDIO CODEC FOR THE LRAC 2025 CHALLENGE TRACK 2

Lingling Dai[⋆∗], *Zhe Han*[†], *Andong Li*[⋆∗], *Yiqing Guo*[†], *Linping Xu*[†],
Hua Gao[†], Xiaodong Li[⋆∗], Chengshi Zheng[⋆∗]

⋆ Institute of Acoustics, Chinese Academy of Sciences, Beijing, China
†ByteDance, China
∗University of Chinese Academy of Sciences, Beijing, China

ABSTRACT

This paper presents Enhance-NanoCodec, which is designed to perform codec transmission in conjunction with simultaneous denoising and dereverberation under the constraints of low complexity, low bitrate and real-time processing. Our architecture operates in the time-frequency (T-F) domain, where we discard the phase and only encode the magnitude features on the encoder side—both the magnitude and phase are estimated on the receiver side. To scientifically allocate the complexity ratio of the model between the encoder and decoder, and to utilize the codebook more efficiently, we designed a multi-stage training scheme, which excellently accomplishes the joint task of speech enhancement and coding. In addition, we propose an efficient convolution-style attention block as the core modeling unit. Enhance-NanoCodec achieves a total latency of 50 ms and a computational complexity of 1.86 GFlops (0.58 for the decoder), and is submitted to the LRAC Challenge Track 2.

Index Terms— Neural audio codec, speech enhancement, low-complexity, low bitrate, real-time

1. INTRODUCTION

Audio codec technologies are foundational to on-demand streaming. End-to-end Neural audio codecs (NACs) with learnable encoders, including SoundStream [1] and DAC [2], have attracted significant research interest. They stand out for high-quality audio at very low bitrates, a performance target conventional audio coding struggles to achieve. However, several critical issues persist as key focus areas for advancing the practical deployment of NACs in real-world transmission scenarios, including high computational cost, strict causality constraints, non-negligible algorithmic delay, and the ongoing challenge of ensuring clear speech transmission amid complex background noise.

The objective of Track 2 in the LRAC 2025 Challenge [1] is to achieve the integration of speech enhancement and coding under the joint constraints of low latency, low computational complexity, real-time processing, and high quality. To this end, we propose the Enhance-NanoCodec architecture. This system is engineered to fulfill the challenge constraints while maintaining robust performance, and its capabilities are fully optimized through a multi-stage training scheme for high-quality speech enhancement and coding.

First, Enhance-NanoCodec operates in the time-frequency (T-F) domain for high-fidelity spectral detail reconstruction. As target coding or estimation in the time domain becomes especially challenging when computational resources are limited, we disregard the phase and utilize only the magnitude for feature encoding, with both magnitude and phase reconstructed in the decoder, leveraging a Fourier prior to ease the learning process. Second, we adopt a convolution-style attention block for spectral modeling. It uses large convolution kernels to generate the attention distribution, effectively aggregating contextual information. Third, joint magnitude and phase estimation under limited resources remains an open challenge. Following [3], we use an omnidirectional phase loss for phase optimization, which captures differential relations between center and neighboring phase bins. We further extend this to the spectrum's real and imaginary (RI) parts, proposing an omnidirectional RI loss. Finally, inspired by [4], we design a multi-stage training scheme to further enhance the codebook's efficiency in leveraging clean speech data within Track 2, while optimizing task allocation between the encoder and decoder. This comprehensive training strategy enables the model to accomplish the dual objectives of high-quality speech enhancement and reconstruction, all while fully complying with the challenge requirements.

[1] https://crowdsourcing.cisco.com/lrac-challenge/2025/

2. METHOD ILLUSTRATIONS

2.1. Overall Architecture

The overall structure of the proposed Enhance-NanoCodec is presented in Fig. 1(a). Given the input waveform $x \in \mathbb{R}^L$, we first transform it into the time-frequency (T-F) domain using the short-time Fourier transform (STFT), obtaining the complex spectrogram $X \in \mathbb{C}^{F\times T}$, where F and T denote the number of frequency bins and time frames, respectively. For the encoder input, we drop the phase counterpart and use the normalized magnitude spectrogram $|X| \in \mathbb{R}^{F\times T}$ along with the spectral energy, which is extracted via the energy-content decoupling (ECD) layer. Then the encoder extracts the frequency information and obtains highly compressed hidden representations, which are matched with a sequence of discrete codes $C \in \mathbb{R}^{N_q\times D\times T}$ through residual vector quantization (RVQ), where N_q is the codebook number and D is the feature dimension. The decoder takes the quantized codes as input and reconstructs both the magnitude spectrogram and the phase spectrogram. Finally, we recover the enhanced waveform $\hat{x} \in \mathbb{R}^L$ by applying the inverse STFT (iSTFT). Both encoder and decoder share the same modeling unit, which is composed of a stack of Large Kernel Convolution-Style Attention Block (LKCAB) as shown in Fig. 1(b). The proposed LKCAB uses large convolution kernels to generate the attention distribution, effectively aggregating contextual information.

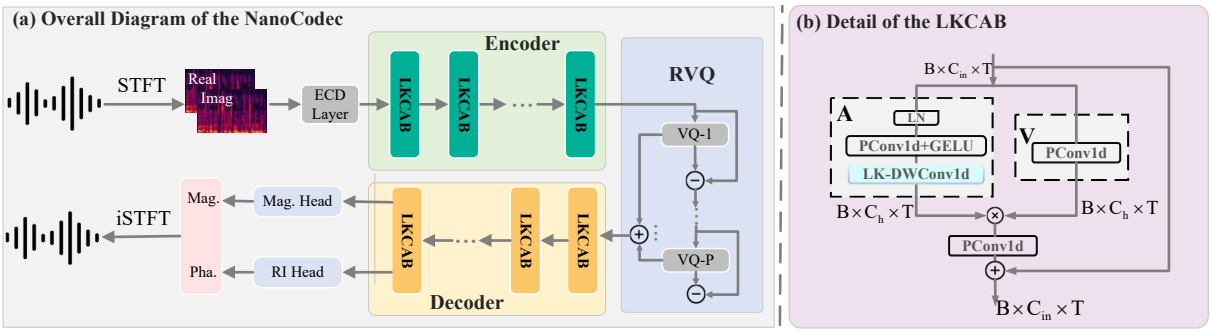


Fig. 1. (a) Overall structure of the proposed NanoCodec; (b) Internal structure of the adopted LKCAB.

2.2. Multi-stage Training Scheme

In Track 2 of the LRAC 2025 Challenge, the computational budget is intentionally biased toward the transmitter side, with a substantially higher complexity allocation compared to the receiver. Since the track specifically focuses on enhanced speech, we introduce a multi-stage training strategy aimed at improving the codebook's efficiency in representing clean speech while achieving a more balanced computational distribution between the encoder and decoder. The detailed training procedure is elaborated as follows.

2.2.1. Stage 1: Training codebook of clean speech

To ensure that all information in the codebook is dedicated to transmitting valid clean speech, only clean speech is used during the training process. It is important to note that the codebook for both 1 kbps and 6 kbps were finalized in this stage. To better guide the encoder's performance and avoid being constrained by the decoding bottleneck of the decoder, a decoder with a complexity exceeding that required by the challenge is employed for speech encoding during this stage.

2.2.2. Stage 2: Training a speech-enhancement encoder

In Stage 2, an encoder with noise reduction capability is trained. At this stage, various speech augmentations were applied to the data, including the addition of noise and reverberation, as well as other augmentations mentioned in 3.3. During this phase, the codebook and decoder learned in Stage 1 were fixed, with only the encoder undergoing training.

2.2.3. Stage 3: Training a low complexity decoder

In Stage 3, the objective was to train a decoder that meets the computational complexity requirements and is compatible with the encoder and codebook obtained in the previous two stages. During this stage, both the codebook and the encoder were fixed.

3. MISCELLANEOUS CONFIGURATIONS

3.1. Network Setups

For both STFT and iSTFT, the window length is set to 50 ms with a hop size of 12.5 ms. No auxiliary look-ahead nor algorithmic delay is introduced, resulting in a total system latency of 50 ms. The number of LKCABs used in the encoder is set to 12 with a hidden dimension of 600, while the decoder uses 10 LKCABs with a hidden dimension of 420. The number of codebooks is set to 1 with a codebook size of 5792 for the 1 kbps transmission rate. For the 6 kbps transmission rate, we reuse the codebook from the 1 kbps setup. Additionally, we introduce the grouped RVQ, where the codebooks are divided into two groups, with each group containing 3 codebooks and a codebook size of 1024. The theoretical transmission rate is 1.00 kbps for 1 kbps transmission and 5.80 kbps for 6 kbps transmission. The total trainable parameter count for Enhance-NanoCodec is 11.51 M, and the total computational complexity is 1.86 GFlops, where the decoder accounts for 0.58 GFlops. The detailed model parameters and computational complexity of each module are presented in Table 1.

Module	Para. (M)	Complexity (MFlops)
Encoder	7.84	1266.66
Quantizer	0.08	16.32
Decoder	3.59	578.63

Table 1. Model parameter and computational complexity.

3.2. Loss Setups

We use both reconstruction and adversarial losses during Stage 1 and Stage 2 training. The reconstruction loss consists of multi-resolution STFT loss, multi-resolution Mel loss, as well as our proposed omni-directional phase loss, which captures differential relations between center and neighboring phase bins. For adversarial training, we employ a multi-period discriminator (MPD), multi-resolution STFT discriminator (MRSTFTD), and multi-band discriminator (MBD), along with a feature matching loss. In Stage 3, to further improve the performance of the low-complexity decoder, we additionally incorporate PESQ loss, UTMOS loss, as well as our proposed omnidirectional RI loss for optimization, where the former two provide perceptual supervision and the latter enables finer joint magnitude-and-phase reconstruction.

3.3. Dataset Setups

The training corpus employed in this study is sourced from the LRAC 2025 Challenge. For speech, we use the speech clips from LibriSpeech [5], LibriVox [6], VCTK [7], EARS [8] and Multilingual Librispeech [9]. For noise set, we include Audioset [10], Freesound [11](from the DNS5 challenge[2]), FMA [12],

[2]https://github.com/microsoft/DNS-Challenge

WHAM! [13] and FSD50K [14]. For reverberation generation, we include the room impulse responses (RIRs) from Open SLR 28[3] and Motus [15]. Further refinement was performed by excluding audio segments that were excessively short or exhibited abnormally low energy, thereby ensuring the quality and consistency of the training samples.

During model training after stage 1, noisy and reverberant signals were synthesized on-the-fly via random sampling from the speech, RIR, and noise datasets. Specifically, under noisy speech conditions, the signal-to-noise ratio (SNR) was set to range from -5 dB to 20 dB. To enhance model generalization, we applied additional data augmentation to 20% of the training corpus, implementing specific techniques including bandwidth limitation, amplitude clipping, and packet loss concealment (PLC).

3.4. Evaluation Metrics

In this study, model performance is initially evaluated using both the non-intrusive metric UTMOS [16] and the intrusive metric PESQ [17], which facilitated rapid evaluation and informed iterative adjustments to the model architecture and training procedures. For the final selection of the model, comprehensive human listening tests were conducted to ensure robust perceptual quality.

3.5. Training Settings

We optimized the model using AdamW optimizer [18] with its default betas (0.8, 0.99) and an initial learning rate of 0.0002. The learning rate is scheduled using an ExponentialLR scheduler with a gamma of 0.999998 per epoch. Additionally, we set the batch size to 16 and the duration of each sample to 5 seconds. For each training stage, the number of training steps was set to a range of 500,000 to 1,000,000, depending on the convergence of the evaluation metrics.

4. REFERENCES

[1] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi, "SoundStream: An End-to-End Neural Audio Codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2021.

[2] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar, "High-fidelity audio compression with improved rvqgan," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[3] Andong Li, Tong Lei, Zhihang Sun, Rilin Chen, Erwei Yin, Xiaodong Li, and Chengshi Zheng, "Learning Neural Vocoder from Range-Null Space Decomposition," *arXiv preprint arXiv:2507.20731*, 2025.

[4] Yunlong Liu, Tao Huang, Weisheng Dong, Fangfang Wu, Xin Li, and Guangming Shi, "Low-light image enhancement with multi-stage residue quantization and brightness-aware attention," in *Proc. ICCV*, 2023, pp. 12140–12149.

[5] Heiga Zen, Viet Dang, Rob Clark, Yu Zhang, Ron J Weiss, Ye Jia, Zhifeng Chen, and Yonghui Wu, "Libritts: A corpus derived from librispeech for text-to-speech," *arXiv preprint arXiv:1904.02882*, 2019.

[6] Jodi Kearns, "Librivox: Free public domain audiobooks," 2014.

[7] Christophe; MacDonald Kirsten Yamagishi, Junichi; Veaux, "CSTR VCTK Corpus: English Multi-speaker Corpus for CSTR Voice Cloning Toolkit,," .

[8] Julius Richter, Yi-Chiao Wu, Steven Krenn, Simon Welker, Bunlong Lay, Shinji Watanabe, Alexander Richard, and Timo Gerkmann, "EARS: An anechoic fullband speech dataset benchmarked for speech enhancement and dereverberation," *arXiv preprint arXiv:2406.06185*, 2024.

[9] Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert, "Mls: A large-scale multilingual dataset for speech research," *arXiv preprint arXiv:2012.03411*, 2020.

[10] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. ICASSP*. IEEE, 2017, pp. 776–780.

[11] Eduardo Fonseca, Jordi Pons Puig, Xavier Favory, Frederic Font Corbera, Dmitry Bogdanov, Andres Ferraro, Sergio Oramas, Alastair Porter, and Xavier Serra, "Freesound datasets: a platform for the creation of open audio datasets," 2017.

[12] Michaël Defferrard, Kirell Benzi, Pierre Vandergheynst, and Xavier Bresson, "FMA: A dataset for music analysis," *arXiv preprint arXiv:1612.01840*, 2016.

[13] Gordon Wichern, Joe Antognini, Michael Flynn, Licheng Richard Zhu, Emmett McQuinn, Dwight Crow, Ethan Manilow, and Jonathan Le Roux, "Wham!: Extending speech separation to noisy environments," *arXiv preprint arXiv:1907.01160*, 2019.

[14] Eduardo Fonseca, Xavier Favory, Jordi Pons, Frederic Font, and Xavier Serra, "Fsd50k: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2021.

[15] Georg Götz, Sebastian J Schlecht, and Ville Pulkki, "A dataset of higher-order ambisonic room impulse responses and 3d models measured in a room with varying furniture," in *2021 Immersive and 3D Audio: from Architecture to Automotive (I3DA)*. IEEE, 2021, pp. 1–8.

[16] Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari, "UTMOS: UTokyo-SaruLab System for the VoiceMOS Challenge 2022," in *Proc. Interspeech*, 2022, pp. 4521–4525.

[17] 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. ICASSP*, 2001, vol. 2, pp. 749–752 vol.2.

[18] Ilya Loshchilov and Frank Hutter, "Decoupled Weight Decay Regularization," in *International Conference on Learning Representations*, 2019.

[3] https://www.openslr.org/28/

PROGRESSIVE REFINEMENT TRAINING FOR LOW-RESOURCE NEURAL SPEECH CODING AND ENHANCEMENT

Ronghui Hu[1,2,†], *Leyan Yang*[1,2,†], *Yang Xu*[1,2,†], *Qinwen Hu*[1,2], *Jing Lu*[1,2,*]

[1]Key Laboratory of Modern Acoustics, Nanjing University, Nanjing 210093, Jiangsu, China
[2]NJU-Horizon Intelligent Audio Lab, Horizon Robotics, Beijing 100094, China

ABSTRACT

Speech codec is a key challenge in hands-free communication systems, where on-device deployment requires real-time processing under strict constraints on bitrate and computational complexity. Meanwhile, real-world acoustic conditions demand integrated speech enhancement (SE). In this paper, we propose a novel Progressive Refinement (PR) strategy to build a high-performance codec for joint speech coding and enhancement. With this strategy, we introduce PR-Vocodec, a low-latency, high-fidelity, and low-bitrate codec, which can perform noise reduction and dereverberation simultaneously with low computational overhead. Experimental results demonstrate that the PR-Vocodec delivers superior performance across multiple evaluation metrics.

Index Terms— progressive refinement, audio neural codec, speech enhancement.

1. INTRODUCTION

The 2025 Low-Resource Audio Codec (LRAC) Challenge focuses on codecs with low computational complexity, low latency, and low transmission bandwidth, as well as multi-task codecs coupled with front-end enhancement tasks. In this paper, we introduce PR-Vocodec, our system submitted to the Challenge. The system is built upon the Vocos architecture [1] and employs a six-layer Residual Vector Quantizer (RVQ) [2] in the quantization module, supporting both 1 kbps and 6 kbps bitrates. The training follows a three-stage progressive refinement (PR) strategy. Stage 1 focuses on constructing a high-fidelity teacher model. Stages 2 and 3 progressively train the student model, enhancing its noise suppression and dereverberation capabilities. This progressive refinement framework not only preserves the quality of the codebooks but also significantly improves the speech enhancement performance and generalization of the student model under low-bitrate constraints.

[†]Equal contribution.
[*]Corresponding author: lujing@nju.edu.cn

2. PROPOSED METHOD

2.1. Codec architecture

As illustrated in Fig.1, we design the backbone architecture based on the Vocos [1] framework and employ it as the decoder. The decoder consists of six 1D ConvNeXt [3] blocks with a hidden dimension of 558, followed by a post-processing network comprising four ResNet blocks and a causal self-attention module [4]. The encoder is constructed as a mirror-symmetric counterpart of the decoder, performing feature extraction of the input speech at the transmitting end through a reversed information flow. Since the encoding stage is coupled with the SE task, we adopt an asymmetric parameter configuration to enhance the encoder's feature extraction and multi-task processing capabilities. Specifically, the encoder consists of twelve 1D ConvNeXt blocks with the hidden dimension increased to 1096. In addition, we employ an RVQ module to encode the embeddings extracted by the encoder. The RVQ consists of six quantization layers, each with a codebook size of 1024.

2.2. PR training strategy

The PR strategy enables the model to achieve high-fidelity audio coding while simultaneously performing high-quality speech enhancement, including noise suppression and dereverberation. As illustrated in Fig. 2, the training process consists of three progressive stages.

In Stage 1, the model follows the standard audio codec training paradigm to obtain a low-bitrate, high-fidelity codec, which serves as the teacher model. The training process adopts a generative adversarial network (GAN) framework, where a multi-scale short-time Fourier transform discriminator (MS-STFTD) [5] is employed to impose multi-scale time–frequency constraints on the reconstructed audio, thereby enhancing the accuracy in frequency band reconstruction.

In Stage 2, the encoder of the student model is trained from scratch to perform joint coding and enhancement. Specifically, the clean speech is fed into the encoder of the teacher model to generate target embeddings, while

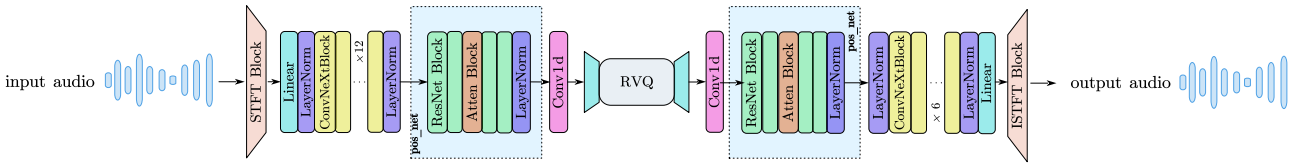


Fig. 1. An overview of the proposed PR-Vocodec backbone.

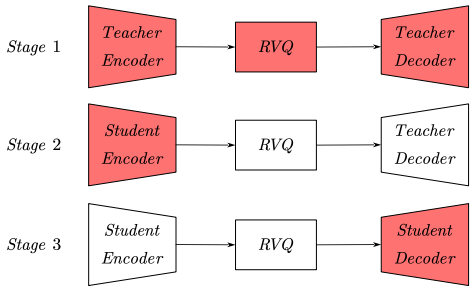


Fig. 2. The Schematic of the PR training strategy. The red blocks are updated during the training stage, while the white blocks are frozen.

the noisy and reverberant speech is passed through the student encoder. During training, the two sets of embeddings are aligned, guiding the student encoder to produce representations that closely match those of the teacher model when processing clean inputs. This alignment effectively implements noise and reverberation suppression within the encoder module. Crucially, the codebook remains frozen throughout this stage, ensuring that the decoder's input space remains consistent with that of the teacher model.

Stage 3 is the dual process of Stage 2, aiming to enhance the robustness of the student decoder and thereby improve the system's generalization to noisy or reverberant inputs. During this stage, the encoders and RVQ modules of both teacher and student models are frozen. Clean and noisy speech pairs are processed in parallel, aligning the decoder outputs and the reconstructed waveforms to promote consistent decoding behavior. This process ensures the output of the student decoder closely approximates the output of the teacher decoder for clean speech, enhancing the robustness against variations in encoder output. Furthermore, adversarial training is incorporated in this stage by employing the discriminator pre-trained during Stage 1 to refine the decoder outputs of the student model. To balance the training progress between the decoder and the discriminator, we update the decoder five times for each discriminator update to ensure balanced convergence and maintain stable adversarial training.

2.3. Loss function

The training of the teacher codec utilizes a composite loss function within a GAN framework. The total generator loss, $L_{generator}$, is a weighted sum of multiple components: the multi-scale mel-spectrogram reconstruction loss L_{rec} [6], the generator adversarial loss L_g, the feature matching loss L_{feat} applied to the discriminator's features, the codebook loss L_{code}, and the commitment loss L_c. It is formulated as:

$$L_{rec} = \|\mathcal{M}(x) - \mathcal{M}(\hat{x})\|_1 \tag{1}$$

$$L_g = \|1 - D(\hat{x})\|_2^2 \tag{2}$$

$$L_{feat} = 2\sum_l \left\|D^l(x) - D^l(\hat{x})\right\|_1 \tag{3}$$

$$\begin{aligned} L_{\text{generator}} =& \lambda_{\text{rec}} L_{\text{rec}} + \lambda_g L_g + \lambda_{\text{feat}} L_{\text{feat}} \\ &+ \lambda_{\text{code}} \underbrace{\|\text{sg}[\mathbf{z}_e] - \mathbf{e}_k\|_2^2}_{L_{\text{code}}} + \lambda_c \underbrace{\|\mathbf{z}_e - \text{sg}[\mathbf{e}_k]\|_2^2}_{L_c} \end{aligned} \tag{4}$$

In the above equations, x and $\hat{x}$ denote the target and reconstructed speech, respectively, $\mathcal{M}(\cdot)$ is the mel-spectrogram transform, $D(\cdot)$ is the discriminator output, $D^l(\cdot)$ represents the feature map of the l-th discriminator layer, $\mathbf{z}_e$ is the quantizer output, and $\mathbf{e}_k$ is the codebook vector. $\text{sg}[\cdot]$ denotes the stop-gradient operation, indicating that its gradients are detached from the computation graph and do not participate in backpropagation. The multi-scale mel-spectrogram loss L_{rec} is computed using window length samples [32, 64, 128, 256, 512, 1024, 2048], with the hop length fixed at 1/4 of each window length. Each scale uses different mel bins of [5, 10, 20, 40, 80, 160, 320]. Loss weights are set as: $\lambda_{\text{rec}} = 15$, $\lambda_g = 2$, $\lambda_{\text{feat}} = 1$, $\lambda_{\text{code}} = 1$, $\lambda_c = 0.25$. The discriminator is trained with adversarial loss L_d, which is formulated as:

$$L_d = \|1 - D(x)\|_2^2 + \|D(\hat{x})\|_2^2 \tag{5}$$

In Stage 2, the loss function $L_{PR-encoder}$ combines the mean squared error (MSE) and cosine distance between the teacher and student embeddings, weighted by 1.0 and 0.2, respectively.

Table 1. Objective Performance Comparison on the Open Test Set.

Bitrate	Model	Condition	ScoreQ-ref	UTMOS	Sheet-SSQA	PESQ	Audiobox AE-CE
6 kbps	Baseline	Clean	0.435	2.972	3.548	2.126	5.381
		Noisy	0.753	2.562	3.122	1.723	4.754
		Reverb	0.913	1.803	3.273	1.295	4.381
	Stage 2	Clean	0.164	3.790	3.917	3.215	5.786
		Noisy	0.348	3.594	3.706	2.428	5.540
		Reverb	0.364	3.517	3.883	2.092	5.597
	Stage 3	Clean	**0.158**	**3.785**	**3.929**	**3.244**	**5.795**
		Noisy	**0.317**	**3.613**	**3.755**	**2.444**	**5.592**
		Reverb	**0.340**	**3.560**	**3.890**	**2.116**	**5.659**
1 kbps	Baseline	Clean	1.008	1.371	2.079	1.207	4.163
		Noisy	1.150	1.351	2.520	1.180	3.918
		Reverb	1.117	1.323	3.065	1.153	3.723
	Stage 2	Clean	0.386	**3.306**	3.609	1.959	5.470
		Noisy	0.470	3.236	3.537	1.753	5.370
		Reverb	0.466	3.202	**3.666**	1.657	**5.392**
	Stage 3	Clean	**0.364**	3.305	**3.648**	**1.991**	**5.490**
		Noisy	**0.463**	**3.242**	**3.541**	**1.786**	**5.383**
		Reverb	**0.465**	**3.211**	3.626	**1.674**	5.391

In Stage 3, the outputs of the decoder's final hidden layer are optimized using the same loss function as in Stage 2, denoted as $L_{PR-decoder}$. Meanwhile, the decoded speech is trained with the same loss formulation as in Stage 1, denoted as $L_{generator}$. The overall training objective for the decoder at this stage is therefore given by:

$$L_{stage-3} = L_{PR-decoder} + L_{generator}. \quad (6)$$

3. EXPERIMENTAL SETUP

3.1. Training data preparation

In Stage 1, the teacher model is trained on the EARS, VCTK, Common Voice, LibriTTS, Multilingual LibriSpeech, and DNS Challenge 5 datasets. All speech data are resampled to 24 kHz. In Stages 2 and 3, we extend the student model's capability in noise suppression and dereverberation by constructing an additional noise dataset derived from VCTK, WHAM, FSD50K, and FMA, covering a diverse range of noise types. During training, each clean speech sample is mixed with background noise with a probability of 80%, where the signal-to-noise ratio (SNR) is uniformly sampled between -5 dB and 30 dB. To simulate reverberant conditions, room impulse responses (RIRs) from the Motus dataset are applied, with each sample augmented with reverberation at a probability of 50%. All training data are processed following the cleaning and preprocessing procedures specified in the official baseline[1]

[1] `https://github.com/cisco-open/lrac_data_generation`

3.2. Implementation Details

In Stage 1, the teacher model is trained for 1000 epochs with a batch size of 192, using the AdamW optimizer with a cosine annealing learning rate scheduler. In Stage 2, the student encoder is trained to replicate the teacher model's embeddings. This stage runs for 500 epochs with a batch size of 40, optimized by RAdam with an exponential decay scheduler. In Stage 3, the student decoder is trained for 200 epochs with a batch size of 192 and optimized by the AdamW optimizer with an exponential learning rate decay scheduler.

3.3. Computational complexity and latency

The computational complexity of the teacher model is 349.29M multiply–accumulate operations per second (MACs/s)[2] (with the decoder accounting for 281.57M MACs/s) and the model contains 3.47M parameters. The overall student model comprises 12.37M parameters and operates with a computational complexity of 1.25G MACs/s (with the decoder accounting for 281.29M MACs)

The teacher model incurs an algorithmic latency of 30 ms due to the 720-point STFT. In contrast, the student model has a total latency of 50 ms, comprising the same 30 ms algorithmic latency and an additional 20 ms buffering latency introduced in the encoder.

[2] The computational complexity is calculated by ptflops: `https://github.com/tel-0s/ptflops`.

4. RESULTS

Evaluation on the open test set is conducted using the official metrics provided by the challenge, which contain five objective metrics: ScoreQ_ref [7], UTMOS [8], Sheet-SSQA [9], PESQ [10], and Audiobox Aesthetics_CE [11].[3] Experimental results are summarized in Table 1. The results show that PR-Vocodec significantly outperforms the baseline across all scenarios at both bitrates, particularly demonstrating strong robustness and generalization for reverberant data. Furthermore, the comparison between Stage 2 and Stage 3 shows that the decoder retraining enhances the model's adaptability and consistency, thereby validating the effectiveness of the PR training strategy in achieving high-fidelity speech coding with strong enhancement capability.

5. CONCLUSION

This paper introduces our proposed PR training strategy designed for joint speech coding and enhancement tasks, and our PR-Vocodec model submitted to the LRAC Challenge. The proposed approach achieves competitive performance in the LRAC challenge, surpassing the baseline by a large margin across different bitrates and input conditions.

6. REFERENCES

[1] Hubert Siuzdak, "Vocos: Closing the gap between time-domain and fourier-based neural vocoders for high-quality audio synthesis," *arXiv preprint arXiv:2306.00814*, 2023.

[2] Biing-Hwang Juang and A Gray, "Multiple stage vector quantization for speech coding," in *ICASSP'82. IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 1982, vol. 7, pp. 597–600.

[3] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11976–11986.

[4] Shengpeng Ji, Ziyue Jiang, Wen Wang, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Xize Cheng, Zehan Wang, Ruiqi Li, et al., "Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling," *arXiv preprint arXiv:2408.16532*, 2024.

[5] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[6] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar, "High-fidelity audio compression with improved rvqgan," *Advances in Neural Information Processing Systems*, vol. 36, pp. 27980–27993, 2023.

[7] Alessandro Ragano, Jan Skoglund, and Andrew Hines, "Scoreq: Speech quality assessment with contrastive regression," *Advances in Neural Information Processing Systems*, vol. 37, pp. 105702–105729, 2024.

[8] Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari, "Utmos: Utokyo-sarulab system for voicemos challenge 2022," *arXiv preprint arXiv:2204.02152*, 2022.

[9] Wen-Chin Huang, Erica Cooper, and Tomoki Toda, "Mos-bench: Benchmarking generalization abilities of subjective speech quality assessment models," *arXiv preprint arXiv:2411.03715*, 2024.

[10] Antony W Rix, John G Beerends, Michael P Hollier, and Andries 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)*. IEEE, 2001, vol. 2, pp. 749–752.

[11] Andros Tjandra, Yi-Chiao Wu, Baishan Guo, John Hoffman, Brian Ellis, Apoorv Vyas, Bowen Shi, Sanyuan Chen, Matt Le, Nick Zacharov, et al., "Meta audiobox aesthetics: Unified automatic quality assessment for speech, music, and sound," *arXiv preprint arXiv:2502.05139*, 2025.

[3] Sheet-SSQA and Audiobox AE-CE scores show some deviations from the official results provided by the LRAC challenge.

LOW-COMPLEXITY END-TO-END SPEECH ENHANCEMENT CODEC FOR REAL-TIME COMMUNICATION IN NOISY AND REVERBERANT CONDITIONS

Pincheng Lu *Peng Zhou* *Xiaojiao Chen* *Jing Wang*

Beijing Institute of Technology, Beijing, China

ABSTRACT

End-to-end speech codecs enable efficient low-bitrate communication, but most existing approaches lack integrated enhancement, which limits performance under noisy and reverberant conditions. While recent work has attempted to combine speech enhancement with neural codecs, these methods are often too complex to be practical in low-resource scenarios. In this paper, we present a lightweight speech enhancement codec specifically designed for resource-constrained settings. The proposed system adopts a three-stage training strategy that first establishes strong compression capability and then progressively improves robustness to noise and reverberation. Experimental results demonstrate that our model achieves superior performance in challenging noisy and reverberant environments while meeting strict constraints on computational complexity, latency, and bitrate.

***Index Terms*—** low complexity, speech codec, speech enhancement

1. INTRODUCTION

Speech codecs compress speech signals while preserving perceptual quality [1]. Recent end-to-end models such as SoundStream [2], DAC [3], and L3AC [4] employ encoder–decoder architectures with quantization modules like RVQ or FSQ [5], achieving high-quality reconstruction. However, as most are trained only on clean speech, they lack robustness to real-world noise, making integrated enhancement essential for practical deployment.

Joint enhancement–compression has thus emerged as an active research direction. Early approaches, such as SoundStream and SEStream [6], were trained directly on noisy–clean pairs. More recent methods have explored the use of domain-specific codebooks [7], masked generative models [8, 9], or latent space regression within pretrained codecs [10, 11]. While these approaches have demonstrated promising performance, they often come with high computational complexity, which hinders their applicability in real-time, resource-constrained scenarios.

To address these limitations, we propose a Lightweight Codec for Joint speech compression and enhancement (LJCodec), an end-to-end framework designed to perform both tasks within a unified system. The main contributions of this work are summarized as follows.

- We propose LJCodec, a Lightweight Codec that jointly performs speech compression and enhancement.
- We propose a three-stage training strategy that strengthens noise robustness by training on clean speech, aligning encoder representations from noisy to clean embeddings, and adapting the decoder with the fixed encoder.

2. METHOD

2.1. Model Architecture

The entire model follows the same structure as the baseline. The **encoder** consists of five EncoderBlocks, each composed of several residual convolutional blocks followed by a strided convolution for downsampling. The downsampling factors across the five blocks are 2, 2, 3, 4, and 5, respectively. The **quantizer** employs Residual Vector Quantization (RVQ), where multiple codebooks are cascaded such that each deeper codebook encodes the residual of the previous one. The **decoder** mirrors the encoder architecture and performs upsampling using transposed convolutions with stride equal to the kernel size, thereby reducing the complexity introduced by the upsampling operations. To reduce the computational burden at the receiver side and satisfy LRAC requirements, the convolutional channel width in the decoder is set to about 3/4 of that in the encoder.

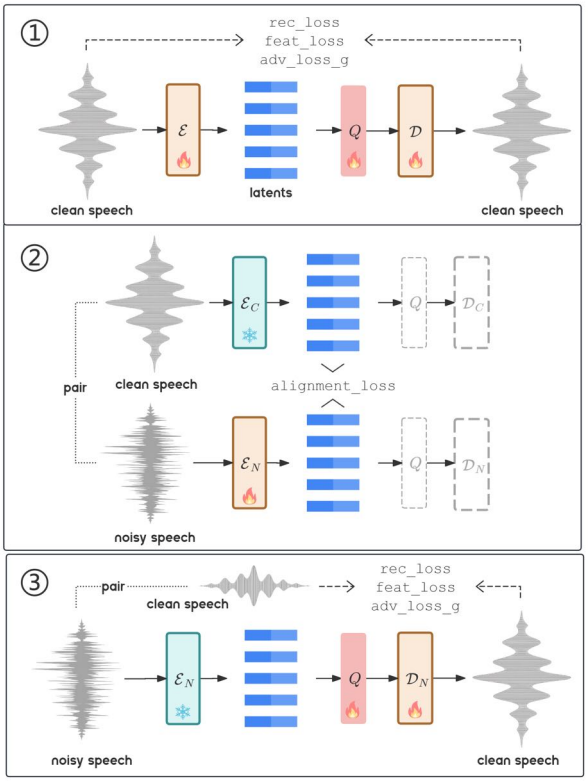


Fig. 1: Proposed stage-wise training strategy.

2.2. Stage-wise Training

To improve robustness against noisy speech, we employ a three-stage training strategy (Fig. 1), starting with clean speech training and followed by independent fine-tuning of the encoder and decoder.

Stage 1. Base Model Training on Clean Speech. In the first stage, we train the codec model exclusively on clean speech using a combination of reconstruction loss, feature loss, commitment loss, and adversarial loss, following the same loss setup and adversarial training strategy as EnCodec.

Let x be the speech to be encoded, and $\hat{x}$ be the speech generated by the decoder. Reconstruction loss is used to measure the difference between $\hat{x}$ and x in both the time domain and the time-frequency domain. The loss in the time and time-frequency domains can be expressed as

$$\ell_t = ||x - \hat{x}||_2^2, \tag{1}$$

and

$$\ell_f = \sum_{s\in\{2^6,\ldots,2^{11}\}} \sum_t ||S_t^s(x) - S_t^s(\hat{x})||_1 + ||\log S_t^s(x) - \log S_t^s(\hat{x})||_2, \tag{2}$$

respectively, where S_t^s represents the t-th frame in the 64-bin mel-spectrogram with window length s and hop length $s/4$. The reconstruction loss ℓ_{rec} is the sum of the time domain loss and the time-frequency domain loss:

$$\ell_{rec} = 100\ell_t(x, \hat{x}) + \ell_f(x, \hat{x}). \tag{3}$$

Feature loss ℓ_{feat} measures the difference between x and $\hat{x}$ in the feature space defined by the discriminators. It is calculated by taking the mean absolute difference between the inner layer output feature maps of the discriminators for the generated speech and the corresponding target speech.

$$\ell_{feat} = E_x \left[\frac{1}{KL} \sum_{k,l} |\mathcal{D}_{k,l}(x) - \mathcal{D}_{k,l}(\hat{x})| \right], \tag{4}$$

where L is the number of intermediate layers, and $\mathcal{D}_{k,l}$ ($l \in \{1, \ldots, L\}$) denotes the output of the l-th layer of discriminator k.

Quantizer commitment loss ℓ_q describes the difference between the input and output of the quantizer. It is used to reduce the discrepancy between the quantizer's embedding space and the encoder's output, which can be expressed by:

$$\ell_q = \sum_{c=1}^{C} ||z_c - q_c(z_c)||_2^2, \tag{5}$$

where q_c represents the c-th vector quantizer.

In adversarial training, the following two adversarial losses are used to optimize the codec and the discriminators. The adversarial loss ℓ_{adv_g} for codec is

$$\ell_{adv\_g} = E_x \left[(1 - \mathcal{D}(\hat{x}))^2 \right], \tag{6}$$

while ℓ_{adv_d} for discriminators is

$$\ell_{adv\_d} = E_x \left[(1 - \mathcal{D}(x))^2 + (1 + \mathcal{D}(\hat{x}))^2 \right]. \tag{7}$$

The total loss for the codec is defined as follows:

$$\ell_g = \lambda_{rec}\ell_{rec} + \lambda_{feat}\ell_{feat} + \lambda_q\ell_q + \lambda_{adv\_g}\ell_{adv\_g}, \tag{8}$$

Table 1: Objective evaluation results at 1 kbps and 6 kbps under clean, noisy, and reverberant conditions.

	ScoreQ	UTMOS	PESQ
1 kbps			
clean	0.39	3.99	1.58
noisy	0.49	3.82	1.44
reverb	0.52	3.61	1.27
6 kbps			
clean	0.27	4.17	2.21
noisy	0.45	3.96	1.77
reverb	0.5	3.63	1.38

Table 2: Computational complexity (MFLOPS) and latency (ms) of different modules.

Component	Compute	Latency
Encoder	1946	20
Quantizer	48	0
Decoder	594	20
Buffering latency	–	10
Total	2588	50

and the discriminator loss ℓ_d is

$$\ell_d = \lambda_{adv\_d}\ell_{adv\_d}. \tag{9}$$

where λ are constant weights used to balance each component.

In our experiments, we trained the model with weights $\lambda_{rec} = \lambda_{feat} = \lambda_{adv\_g} = \lambda_{adv\_d} = 1$, and $\lambda_q = 1000$.

Stage 2. Encoder Alignment Fine-tuning. Inspired by SoundStream, we argue that the enhancement task should be performed before quantization, on the encoder side, to minimize the impact of noisy latent representations on both the quantizer and decoder. Unlike NoiseRobustVRVQ (NRVRVQ) [11], which optimizes the entire model on noisy speech, we perform alignment fine-tuning only on the encoder.

Specifically, we duplicate all modules before the quantizer into a trainable encoder, denoted as $\mathcal{E}_N$, and a frozen encoder, denoted as $\mathcal{E}_C$. Noisy speech x_n is fed into $\mathcal{E}_N$, while clean speech x_c is fed into $\mathcal{E}_C$. The output of $\mathcal{E}_C$ serves as the supervision target for $\mathcal{E}_N$. The $\mathcal{E}_N$ is optimized with a mean squared error loss:

$$\ell_a = \mathbb{E}\left[(\mathcal{E}_N(x_n) - \mathcal{E}_C(x_c))^2 \right]. \tag{10}$$

No additional losses (e.g., reconstruction loss) are introduced, as this design forces the encoder to rapidly adapt to the speech enhancement task on top of its established compression capability.

Stage 3. Decoder Adaptive Fine-tuning. Although the latent distribution after Stage 2 is close to that of Stage 1, slight mismatches remain and lead to reconstruction artifacts. We fine-tune both the quantizer and the decoder to better adapt to these new representations. The encoder $\mathcal{E}_N$ is frozen, while the quantizer $\mathcal{Q}$, decoder $\mathcal{D}_N$, and the discriminators are optimized using the same loss functions as in Stage 1. This strategy improves the overall audio quality with minimal overhead.

3. EXPERIMENT

3.1. Datasets

We trained our codec using the datasets specified by LRAC. In Stage 1, the model was trained on clean speech drawn from LibriVox data from the DNS5 Challenge [12], LibriTTS[13], VCTK[14], EARS[15], CommonVoice[16], and Multilingual LibriSpeech[17].

In Stage 2 and Stage 3, we constructed degraded speech by mixing clean utterances with noise and reverberation. The noise sources included Audioset[18] and FreeSound[19] noises from the DNS5 Challenge, WHAM! noise[20], speech-filtered FSD50K[21], and Free Music Archive[22]. Noisy speech was synthesized by mixing clean utterances with these noises at signal-to-noise ratios (SNR) uniformly sampled between -5 dB and 30 dB. Reverberation was simulated using RIR datasets from OpenSLR28, the DNS5 Challenge, and Motus [23]. All corpora were downsampled to 24 kHz for both training and evaluation. For benchmarking, we used the official LRAC validation and test sets to ensure fair and consistent comparisons.

3.2. Training and Evaluation Settings

Training Settings: The entire model is trained on a single RTX 4090 GPU with a batch size of 32. The number of iterations for Stage 1, Stage 2, and Stage 3 are set to 150k, 50k, and 150k, respectively.

Evaluation Metrics: For preliminary offline testing during the development stage, we adopt PESQ[24], UTMOS[25], and ScoreQ[26] as objective quality metrics. For the official benchmark evaluation, we rely on the toolkit provided by the organizers, which reports a more comprehensive set of metrics, including `sheet_ssqa` [27], `scoreq_ref`, `audiobox_AE_CE` [28], `utmos`, and `pesq`. For model efficiency, we report both the computational complexity and the latency of the proposed codec.

3.3. Speech Quality Metrics

Table 1 summarizes the objective evaluation results. On clean speech compression, LJCodec outperforms the baseline at both 1 kbps and 6 kbps. For degraded speech with additive noise and reverberation, LJCodec also demonstrates consistent improvements over the baseline.

3.4. Model Efficiency

Table 2 presents the computational complexity and latency of our model. The overall complexity is below 2600 MFLOPS, with the receive-side (decoder) complexity under 600 MFLOPS. The end-to-end latency is less than 50 ms, fully meeting the challenge requirements.

4. CONCLUSIONS

We presented **LJCodec**, a low-complexity end-to-end codec that jointly performs speech compression and enhancement. Through a three-stage training strategy, the model achieves robustness to noise and reverberation while maintaining a low bitrate, low latency ($<$50 ms), and low computational complexity ($<$2600 MFLOPS). Experiments on the LRAC benchmark show consistent improvements over the baseline, demonstrating the practicality of LJCodec for real-world low-resource speech communication.

5. REFERENCES

[1] J. Wang, L. Xu, X. Chen *et al.*, "Research review on low bit rate speech coding technology based on neural networks," *Journal of Signal Processing*, vol. 40, no. 12, pp. 2261–2280, 2024.

[2] N. Zeghidour, A. Luebs, A. Omran, J. Skoglund, and M. Tagliasacchi, "Soundstream: An end-to-end neural audio codec," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 495–507, 2021.

[3] R. Kumar, P. Seetharaman, A. Luebs, I. Kumar, and K. Kumar, "High-fidelity audio compression with improved rvqgan," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[4] L. Zhai, H. Ding, C. Zhao, fei wang, G. Wang, W. Zhi, and W. Xi, "L3ac: Towards a lightweight and lossless audio codec," 2025. [Online]. Available: https://arxiv.org/abs/2504.04949

[5] F. Mentzer, D. Minnen, E. Agustsson, and M. Tschannen, "Finite scalar quantization: Vq-vae made simple," *arXiv preprint arXiv:2309.15505*, 2023.

[6] J. Huang, Z. Yan, W. Jiang, and F. Wen, "A two-stage training framework for joint speech compression and enhancement," *arXiv preprint arXiv:2309.04132*, 2023.

[7] X. Bie, X. Liu, and G. Richard, "Learning source disentanglement in neural audio codec," in *IEEE International Conference on Acoustic, Speech and Signal Procssing (ICASSP)*, 2025.

[8] H. Xue, X. Peng, and Y. Lu, "Low-latency speech enhancement via speech token generation," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 661–665.

[9] H. Yang, J. Su, M. Kim, and Z. Jin, "Genhancer: High-fidelity speech enhancement via generative modeling on discrete codec tokens," in *Proc. Interspeech*, vol. 2024, 2024, pp. 1170–1174.

[10] H. Li, J. Q. Yip, T. Fan, and E. S. Chng, "Speech enhancement using continuous embeddings of neural audio codec," in *ICASSP 2025-2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2025, pp. 1–5.

[11] Y. Chae and K. Lee, "Towards bitrate-efficient and noise-robust speech coding with variable bitrate rvq," *arXiv preprint arXiv:2506.16538*, 2025.

[12] H. Dubey, A. Aazami, V. Gopal, B. Naderi, S. Braun, R. Cutler, H. Gamper, M. Golestaneh, and R. Aichner, "Icassp 2023 deep noise suppression challenge," in *ICASSP*, 2023.

[13] H. Zen, V. Dang, R. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu, "Libritts: A corpus derived from librispeech for text-to-speech," *arXiv preprint arXiv:1904.02882*, 2019.

[14] J. Yamagishi, C. Veaux, K. MacDonald *et al.*, "Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit (version 0.92)," 2019.

[15] J. Richter, Y.-C. Wu, S. Krenn, S. Welker, B. Lay, S. Watanabe, A. Richard, and T. Gerkmann, "Ears: An anechoic fullband speech dataset benchmarked for speech enhancement and dereverberation," *arXiv preprint arXiv:2406.06185*, 2024.

[16] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," *arXiv preprint arXiv:1912.06670*, 2019.

[17] V. Pratap, Q. Xu, A. Sriram, G. Synnaeve, and R. Collobert, "Mls: A large-scale multilingual dataset for speech research," *arXiv preprint arXiv:2012.03411*, 2020.

[18] J. F. Gemmeke, D. P. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[19] E. Fonseca, J. Pons Puig, X. Favory, F. Font Corbera, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound datasets: a platform for the creation of open audio datasets," 2017.

[20] G. Wichern, J. Antognini, M. Flynn, L. R. Zhu, E. McQuinn, D. Crow, E. Manilow, and J. L. Roux, "Wham!: Extending speech separation to noisy environments," *arXiv preprint arXiv:1907.01160*, 2019.

[21] E. Fonseca, X. Favory, J. Pons, F. Font, and X. Serra, "Fsd50k: an open dataset of human-labeled sound events," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 829–852, 2021.

[22] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "Fma: A dataset for music analysis," *arXiv preprint arXiv:1612.01840*, 2016.

[23] G. Götz, S. J. Schlecht, and V. Pulkki, "A dataset of higher-order ambisonic room impulse responses and 3d models measured in a room with varying furniture," in *2021 Immersive and 3D Audio: from Architecture to Automotive (I3DA)*. IEEE, 2021, pp. 1–8.

[24] I.-T. Recommendation, "Perceptual evaluation of speech quality (pesq): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs," *Rec. ITU-T P. 862*, 2001.

[25] T. Saeki, D. Xin, W. Nakata, T. Koriyama, S. Takamichi, and H. Saruwatari, "Utmos: Utokyo-sarulab system for voicemos challenge 2022," in *Proc. Interspeech*, 2022, pp. 4521–4525.

[26] A. Ragano, J. Skoglund, and A. Hines, "Scoreq: Speech quality assessment with contrastive regression," *arXiv preprint arXiv:2410.06675*, 2024.

[27] W.-C. Huang, E. Cooper, and T. Toda, "Mos-bench: Benchmarking generalization abilities of subjective speech quality assessment models," *arXiv preprint arXiv:2411.03715*, 2024.

[28] A. Tjandra, Y.-C. Wu, B. Guo, J. Hoffman, B. Ellis, A. Vyas, B. Shi, S. Chen, M. Le, N. Zacharov *et al.*, "Meta audiobox aesthetics: Unified automatic quality assessment for speech, music, and sound," *arXiv preprint arXiv:2502.05139*, 2025.

A LOW-LATENCY VQ-GAN-BASED CODEC WITH KNOWLEDGE DISTILLATION FOR JOINT SPEECH CODING AND ENHANCEMENT

Yang Xu[1,2,†], *Ronghui Hu*[1,2,†], *Leyan Yang*[1,2,†], *Jing Lu*[1,2,*]

[1]Key Laboratory of Modern Acoustics, Nanjing University, Nanjing, 210093, Jiangsu, China
[2]NJU-Horizon Intelligent Audio Lab, Horizon Robotics, Beijing, 100094, Beijing, China

ABSTRACT

The advancement of speech interfaces operating in resource-constrained environments drives the need for neural speech codecs that achieve a critical balance among computational efficiency, minimized bitrate, and low latency. These codecs must also maintain high speech quality under challenging acoustic conditions, integrating robust enhancement capabilities to counteract real-world noise and reverberation. To address these challenges, we present KD-Vocodec, an efficient knowledge distillation (KD) framework for joint speech coding and enhancement. The proposed system achieves superior performance by training a student model to replicate the intermediate representations of a high-fidelity teacher model through feature-level knowledge distillation, thereby delivering high-quality audio at a latency of 30 ms and scalable bitrates from 1 to 6 kbps. Rigorous evaluation on a public test set confirms the superior capability of KD-Vocodec.

***Index Terms*—** neural speech codec, knowledge distillation, speech enhancement

1. INTRODUCTION

The deployment of neural speech codecs on devices with constrained resources requires balancing critical trade-offs between bitrate, computational complexity, latency, and robustness to acoustic noise. The 2025 Low-Resource Audio Codec (LRAC) Challenge focuses on this problem, calling for codecs that perform effectively under realistic and noisy conditions. Motivated by this challenge, a novel framework called KD-Vocodec is proposed in this paper for joint speech coding and enhancement. Its key innovation is a feature-level knowledge distillation technique, which enables the system to learn compact and noise-invariant representations. The resulting codec achieves a low algorithmic latency of 30 ms and supports variable bitrates, delivering enhanced performance without a significant increase in computational complexity.

2. PROPOSED METHOD

Our proposed framework leverages feature-level knowledge distillation to achieve joint speech coding and enhancement under strict latency and bitrate constraints. The system architecture, depicted in Fig.1, is built upon a VQ-GAN-based [1] clean teacher codec. The overarching design employs a teacher-student paradigm wherein a student encoder is trained to replicate the intermediate representations of a pre-trained teacher encoder, facilitating the learning of clean features. However, the final system retains the original decoder weights without fine-tuning.

2.1. Teacher codec architecture

We adopt Vocos [2] as the backbone of our teacher codec architecture, due to its superior performance in speech synthesis. Specifically, a mirrored variant of the Vocos structure is employed as the encoder–decoder backbone. The input waveform is first converted into a time–frequency representation via STFT. The complex spectrogram is split into magnitude and phase components, which are concatenated along the frequency dimension and fed into the network. This combined input is projected into a latent space with dimension D via a linear layer. The encoder consists of multiple convolutional blocks inspired by ConvNeXt [3], aiming to extract deep hierarchical features. Each block contains a 1D depthwise convolution with weight normalization, followed by a pointwise convolution. To ensure strict causality and avoid algorithmic delay, all temporal padding is causal. To enhance sequence modeling, ResNet blocks are incorporated. Inspired by WavTokenizer [4], a causal self-attention mechanism is inserted after the second convolutional block. The resulting features are passed to the quantizer, which uses a Residual Vector Quantizer (RVQ) [5] with 6 layers and gradient-based codebook updates. Linear layers before and after quantization map features between the quantization dimension and a lower-dimensional space.

The encoder configuration is as follows: STFT window size is 720 samples with a hop size of 180; hidden dimension D is 256; the encoder stack contains 12 ConvNeXt layers, each with an expansion channel size of 896. The decoder mirrors the encoder's structure but with reduced capacity to meet receiver-side computational constraints: D is 252, the number of ConvNeXt layers is 4, and the expansion channel size is 256. The projection dimension for RVQ is set to 8.

†Equal contribution.
*Corresponding author: lujing@nju.edu.cn

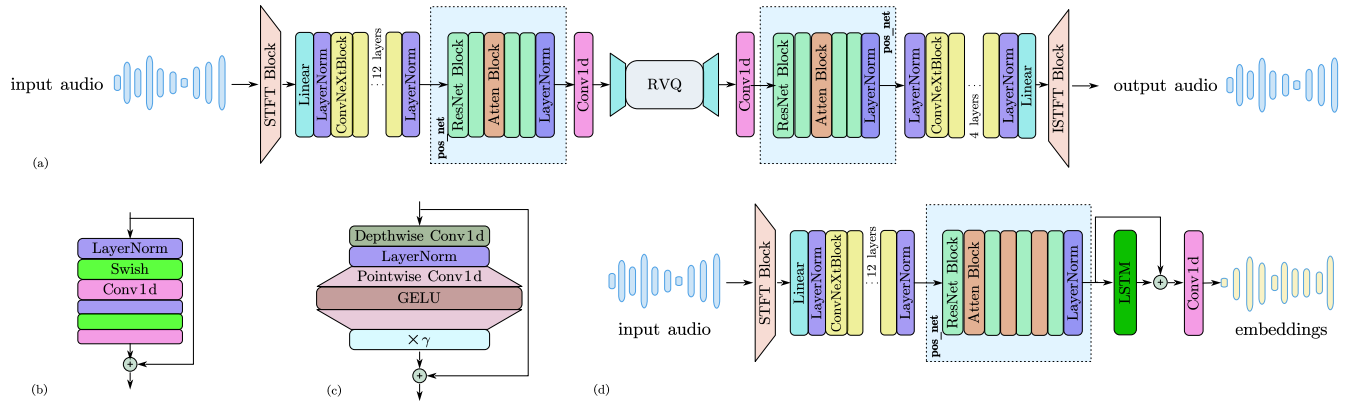


Fig. 1. Architecture of the proposed KD-Vocodec framework. (a) Overall pipeline of the teacher codec; (b) Detailed structure of the ResNet Block; (c) Detailed structure of the ConvNeXt Block; (d) Architecture of the student encoder.

2.2. Student encoder

The student encoder is designed by augmenting the encoder with several key components. This design is motivated by the hypothesis that these augmentations will enable a more robust derivation of clean embeddings from distorted speech inputs. Specifically, causal self-attention modules are incorporated after each ResNet block, except for the final one, to capture long-range contextual dependencies under causal constraints. Furthermore, a two-layer LSTM layer is introduced immediately preceding the final convolutional layer to enhance temporal sequence modeling. A skip connection is also employed between the input and output of this LSTM to facilitate gradient flow and preserve fine-grained temporal information.

2.3. Discriminator

Given that the input to our model is derived from the STFT time-frequency representation, it is advantageous to employ a Multi-Scale STFT Discriminator (MSSTFTD) [6] to assess the reconstruction quality directly in the spectral domain. A set of window lengths [128, 256, 512, 1024, 2048] is used, and the hop length is fixed to one-fourth of the window length. Accordingly, we introduce adversarial training solely using the MSSTFTD to refine the output of the teacher codec.

2.4. Loss function

The training of the teacher codec utilizes a composite loss function within a GAN framework. The total generator loss, $L_{generator}$, is a weighted sum of multiple components: the multi-scale mel-spectrogram reconstruction loss L_{rec} [7], the generator adversarial loss L_g, the feature matching loss L_{feat} applied to the discriminator's features, the codebook loss L_{code}, and the commitment loss L_c. It is formulated as:

$$L_{rec} = \|\mathcal{M}(x) - \mathcal{M}(\hat{x})\|_1 \tag{1}$$

$$L_g = \|1 - D(\hat{x})\|_2^2 \tag{2}$$

$$L_{feat} = 2\sum_l \left\|D^l(x) - D^l(\hat{x})\right\|_1 \tag{3}$$

$$\begin{aligned} L_{\text{generator}} =& \lambda_{\text{rec}} L_{\text{rec}} + \lambda_g L_g + \lambda_{\text{feat}} L_{\text{feat}} \\ &+ \lambda_{\text{code}} \underbrace{\|\text{sg}[\mathbf{z}_e] - \mathbf{e}_k\|_2^2}_{L_{\text{code}}} + \lambda_c \underbrace{\|\mathbf{z}_e - \text{sg}[\mathbf{e}_k]\|_2^2}_{L_c} \end{aligned} \tag{4}$$

In the above equations, x and $\hat{x}$ denote the target and reconstructed speech, respectively, $\mathcal{M}(\cdot)$ is the mel-spectrogram transform, $D(\cdot)$ is the discriminator output, $D^l(\cdot)$ represents the feature map of the l-th discriminator layer, $\mathbf{z}_e$ is the quantizer output, and $\mathbf{e}_k$ is the codebook vector. The multi-scale mel-spectrogram loss L_{rec} is computed using window length samples [32, 64, 128, 256, 512, 1024, 2048], with the hop length fixed at $1/4$ of each window length. Each scale uses different mel bins of [5, 10, 20, 40, 80, 160, 320]. Loss weights are set as: $\lambda_{\text{rec}} = 15$, $\lambda_g = 2$, $\lambda_{\text{feat}} = 1$, $\lambda_{\text{code}} = 1$, $\lambda_c = 0.25$. The discriminator is trained separately with the adversarial loss L_d.

$$L_d = \|1 - D(x)\|_2^2 + \|D(\hat{x})\|_2^2 \tag{5}$$

For knowledge distillation in the student encoder, the loss combines the MSE and cosine distance between the teacher and student embeddings, weighted by 1.0 and 0.1, respectively.

3. EXPERIMENTAL SETUP

3.1. Training data preparation

We trained our model on a large-scale speech dataset curated from high-quality speech samples obtained from the EARS, VCTK, Common Voice, LibriTTS, Multilingual LibriSpeech datasets, and DNS Challenge 5 dataset. All speech signals are resampled to 24 kHz. To extend the noise suppression and dereverberation capabilities of the model, we further constructed a noise data set that includes noise from the VCTK, WHAM, FSD50K, and FMA datasets, encompassing various noise types. During training, each speech sample is combined with background noise with an 80% probability, where signal-to-noise ratios (SNRs) are uniformly distributed between -5 dB and 30 dB. For reverberation, we use room impulse responses (RIRs) from the Motus dataset, and each sample is

Table 1. Objective Performance Comparison on the Open Test Set

Bitrate	Model	Condition	ScoreQ-ref	UTMOS	Sheet-SSQA	PESQ	Audiobox AE-CE
6 kbps	Baseline	Clean	0.43	2.97	3.55	2.13	5.25
		Noisy	0.75	2.56	2.92	1.73	4.6
		Reverb	0.92	1.79	2.67	1.29	4.25
	Proposed	Clean	**0.15**	**3.74**	**4.26**	**3.22**	**5.69**
		Noisy	**0.40**	**3.36**	**3.73**	**2.23**	**5.29**
		Reverb	**0.48**	**3.08**	**3.51**	**1.80**	**5.27**
1 kbps	Baseline	Clean	1.01	1.37	2.07	1.21	3.96
		Noisy	1.15	1.35	1.95	1.18	3.7
		Reverb	1.12	1.32	2.43	1.15	3.55
	Proposed	Clean	**0.38**	**3.26**	**3.60**	**1.94**	**5.37**
		Noisy	**0.53**	**3.00**	**3.30**	**1.61**	**5.14**
		Reverb	**0.62**	**2.74**	**3.06**	**1.43**	**5.01**

augmented with reverberation with a probability of 50% during training. All training data follow the cleaning and preprocessing procedures defined in the baseline[1].

3.2. Implementation Details

Notably, our approach avoids using any pre-trained models throughout the training and inference pipeline. The training procedure consists of two distinct stages. The first stage involves training the teacher codec using a GAN-based reconstruction objective. This model is trained for 1000 epochs with a batch size of 128, using the AdamW optimizer with a cosine annealing learning rate scheduler. In the subsequent distillation stage, the student encoder is trained to replicate the teacher's embeddings. This stage runs for 500 epochs with a batch size of 384, optimized by RAdam with an exponential decay scheduler.

3.3. Computational complexity

The teacher codec operates with 1.11G multiply–accumulate operations per second (MACs) computational complexity (with the decoder accounting for 281.57M MACs) and contains 11.07M parameters. By integrating the student encoder (979.18M MACs), the complete system achieves a complexity of 1.28G MACs with 12.65M parameters. The system maintains strict causality without look-ahead. Consequently, the algorithmic latency is determined solely by the 30-ms STFT analysis window at a 24 kHz sampling rate. The system supports variable bitrates via its RVQ module, where each quantizer layer provides approximately 1 kbps (using a 1024-codebook at 100 fps), allowing operational modes of 1 kbps (1-layer) and 6 kbps (6-layers).

3.4. Checkpoint selection strategy

We employ a systematic strategy for selecting the final model checkpoint. The validation objective metrics are evaluated at regular intervals during training. Should a consistent and pronounced degradation in these metrics be observed, the early stopping strategy is triggered, and the checkpoint with the best performance up to that point is selected. Otherwise, the model checkpoint achieving the lowest training loss at the end of the training process was chosen as the final model.

[1] https://github.com/cisco-open/lrac_data_generation

4. EVALUATION RESULTS

The proposed approach is evaluated using the official challenge metrics and compared against the official baseline system [8]. As shown in Table 1, the KD-Vocodec framework demonstrates consistent performance improvements at both operational bitrates of 1 kbps and 6 kbps. The evaluation employs five objective metrics—ScoreQ_ref [9], UTMOS [10], Sheet-SSQA [11], PESQ [12], and Audiobox Aesthetics_CE [13]—selected for their high correlation with subjective quality assessments, as confirmed by Pearson correlation analysis, thereby providing a reliable measure of decompressed speech quality.

5. CONCLUSION

This paper introduces KD-Vocodec, our submission to Track 2 of the 2025 Low-Resource Audio Codec (LRAC) Challenge. Experimental evaluations demonstrate that KD-Vocodec delivers superior performance over the baseline under diverse acoustic conditions. The system provides an effective solution for real-world speech coding applications that require efficient processing on resource-constrained devices.

6. REFERENCES

[1] Patrick Esser, Robin Rombach, and Bjorn Ommer, "Taming transformers for high-resolution image synthesis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 12873–12883.

[2] Hubert Siuzdak, "Vocos: Closing the gap between time-domain and fourier-based neural vocoders

for high-quality audio synthesis," *arXiv preprint arXiv:2306.00814*, 2023.

[3] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie, "A convnet for the 2020s," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 11976–11986.

[4] Shengpeng Ji, Ziyue Jiang, Wen Wang, Yifu Chen, Minghui Fang, Jialong Zuo, Qian Yang, Xize Cheng, Zehan Wang, Ruiqi Li, et al., "Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling," *arXiv preprint arXiv:2408.16532*, 2024.

[5] Biing-Hwang Juang and A Gray, "Multiple stage vector quantization for speech coding," in *ICASSP'82. IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 1982, vol. 7, pp. 597–600.

[6] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.

[7] Rithesh Kumar, Prem Seetharaman, Alejandro Luebs, Ishaan Kumar, and Kundan Kumar, "High-fidelity audio compression with improved rvqgan," *Advances in Neural Information Processing Systems*, vol. 36, pp. 27980–27993, 2023.

[8] Yusuf Ziya Isik and Rafał Łaganowski, "Low resource audio codec challenge baseline systems," *arXiv preprint arXiv:2510.00264*, 2025.

[9] Alessandro Ragano, Jan Skoglund, and Andrew Hines, "Scoreq: Speech quality assessment with contrastive regression," *Advances in Neural Information Processing Systems*, vol. 37, pp. 105702–105729, 2024.

[10] Takaaki Saeki, Detai Xin, Wataru Nakata, Tomoki Koriyama, Shinnosuke Takamichi, and Hiroshi Saruwatari, "Utmos: Utokyo-sarulab system for voicemos challenge 2022," *arXiv preprint arXiv:2204.02152*, 2022.

[11] Wen-Chin Huang, Erica Cooper, and Tomoki Toda, "Mos-bench: Benchmarking generalization abilities of subjective speech quality assessment models," *arXiv preprint arXiv:2411.03715*, 2024.

[12] Antony W Rix, John G Beerends, Michael P Hollier, and Andries 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)*. IEEE, 2001, vol. 2, pp. 749–752.

[13] Andros Tjandra, Yi-Chiao Wu, Baishan Guo, John Hoffman, Brian Ellis, Apoorv Vyas, Bowen Shi, Sanyuan Chen, Matt Le, Nick Zacharov, et al., "Meta audiobox aesthetics: Unified automatic quality assessment for speech, music, and sound," *arXiv preprint arXiv:2502.05139*, 2025.

BASELINE SYSTEMS FOR THE 2025 LOW-RESOURCE AUDIO CODEC CHALLENGE

Yusuf Ziya Isik, Rafał Łaganowski

Collaboration AI, Cisco Systems, Inc.

ABSTRACT

The Low-Resource Audio Codec (LRAC) Challenge aims to advance neural audio coding for deployment in resource-constrained environments. The first edition focuses on low-resource neural speech codecs that must operate reliably under everyday noise and reverberation, while satisfying strict constraints on computational complexity, latency, and bitrate. Track 1 targets *transparency codecs*, which aim to preserve the perceptual transparency of input speech under mild noise and reverberation. Track 2 addresses *enhancement codecs*, which combine coding and compression with denoising and dereverberation. This paper presents the official baseline systems for both tracks in the 2025 LRAC Challenge. The baselines are convolutional neural codec models with Residual Vector Quantization, trained end-to-end using a combination of adversarial and reconstruction objectives. We detail the data filtering and augmentation strategies, model architectures, optimization procedures, and checkpoint selection criteria.

***Index Terms*—** LRAC 2025, baseline, transparency codecs, enhancement codecs, residual vector quantizer, generative adversarial networks

1. INTRODUCTION

This paper presents the design and training of the baseline models for the two tracks of the 2025 LRAC Challenge.[1] The challenge imposes strict constraints on latency, computational complexity, and transmission bandwidth. All participating codec systems must operate at a 24 kHz sampling rate and support both an ultra-low bitrate mode (up to 1 kbps) and a low-bitrate mode (up to 6 kbps) within a single system. Track 1, the transparency codec track, permits up to 30 ms of latency, including buffering but excluding processing latency. Track 2, the enhancement codec track, allows up to 50 ms of latency. The computational complexity limits are 700 MFLOPS for Track 1 (with no more than 300 MFLOPS on the receive side) and 2600 MFLOPS for Track 2 (with no more than 600 MFLOPS on the receive side).

The baseline systems are designed to demonstrate codec implementations that meet the challenge constraints, provide a benchmark for participants, and facilitate entry into the competition. They are made available through two separate public repositories.

The LRAC data generation repository [1] contains scripts to download public datasets, apply preprocessing (such as sampling rate conversion), and curate data using pre-generated file lists. It also handles splitting the data into training, validation, and open test sets to use during the development phase. The actual test phase relies on a blind test set, which will be released at the end of the development phase.

The LRAC baseline development repository [2] is a public fork of the End-to-End Speech Processing (ESPnet) toolkit [3]. It enhances the existing GAN-based neural speech codec training recipes, providing greater flexibility in model and loss function design, and improves the vector quantization implementation. The repository includes designs and configurations for models, loss functions, data loaders, and optimizers. The trained baseline model weights are also provided in the repository under a Creative Commons Attribution-NonCommercial license.

It should be noted that these baseline neural codecs were developed exclusively for the 2025 LRAC Challenge and are not intended for, nor deployed in, any commercial products.

2. DATASETS AND AUGMENTATIONS

To ensure fair comparison across submissions and to facilitate analysis of factors influencing system performance, the LRAC Challenge is conducted on a closed set of publicly available training data for both speech and noise. Publicly available room impulse responses (RIRs) are included in the training data; however, participants may additionally use other RIRs, either recorded or synthetically generated.

For the baseline systems, data preparation involves filtering a curated subset of the original speech files provided by Collaboration AI, Cisco Systems. File selection is guided by estimated quality metrics for signal-to-noise ratio (SNR), reverberation, and effective speech bandwidth. To promote diversity and balance, we further stratify the dataset according to speaker gender, speaker identity, and per-speaker recording durations, using ground-truth annotations when available or estimated values otherwise. Files reserved for the open test set are excluded from training, ensuring no speaker overlap between training and evaluation data. The baseline data gen-

[1] https://lrac.short.gy/2025-lrac-challenge

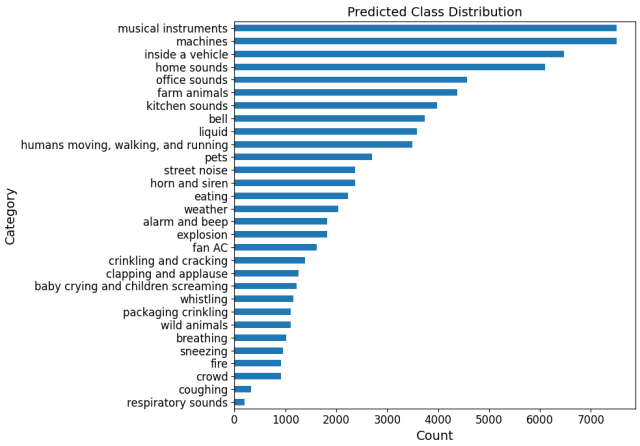
Table 1. Training speech data curation by dataset.

Dataset	Kept (h)	Original (h)	Retention
LibriTTS	46.3	191.3	24.20%
VCTK	22.3	78.8	28.30%
EARS	86.8	86.8	100.00%
Librivox (DNS5)	85.3	313.9	27.17%
MLS (FR, DE, ES)	275.6	450.0	61.24%
GLOBE	186.4	520.9	35.78%
Total	**702.7**	**1641.7**	**42.80%**

eration recipe further sets aside part of the data as validation dataset to be used in hyperparameter tuning and checkpoint selection. Table 1 summarizes the speech datasets and their total durations before and after curation.

The baseline data preparation pipeline also filters noise files based on the curated selection provided by Collaboration AI, Cisco Systems. This curation ensures a diverse and balanced noise dataset spanning major noise categories. To achieve this, we classify all noise files using an ontology derived from AudioSet, simplified to emphasize broad noise types and human vocal sounds. Noise classification is performed with CLAP [4], and files from the most frequent categories are downsampled to balance the distribution. A subset of the noise data is reserved for constructing the open test set used during development, while additional portions of the training noise and reverberation data are held out to form a validation set for hyperparameter tuning and checkpoint selection. Figure 2 illustrates the final distribution of noise data used for training the baseline models.

For the Track 1 baseline model, no data augmentation is applied; the model is trained exclusively on the curated speech data. Training inputs are extracted as sliding windows of 62,400 samples per utterance with 50% overlap. In contrast, the Track 2 baseline employs on-the-fly data augmentation using the EnhPreprocessor class from the ESPnet framework. Reverberation is applied with a probability of 0.5, and additive noise with a probability of 0.8, with signal-to-noise ratios (SNRs) uniformly sampled between -5 and 30 dB. Since EnhPreprocessor supports only a single noise source per utterance, we adopt that constraint. For reverberation, we exclusively use real room impulse responses (RIRs) from the public datasets provided in the LRAC Challenge and do not include synthetically generated RIRs. When reverberation is added to an input utterance, the early reflection component of the room impulse response is also applied to the reference speech. The early reflections are defined as the 50 ms segment following the direct path.



The final training noise data distribution.

3. MODEL ARCHITECTURES

The ESPnet repository includes implementations of several neural audio codecs, including Soundstream [5] and Encodec [6], from which we derive our baseline systems for the LRAC Challenge. These codecs follow a convolutional encoder-decoder architecture with a quantizer in the middle. Both Soundstream and Encodec employ a Residual Vector Quantizer (RVQ) to compress encoder embeddings. Our baseline systems adopt this design and they are trained with a generative adversarial network (GAN) approach.

3.1. Track 1 Baseline Model

The Track 1 baseline model employs an encoder operating directly on the raw audio waveform. It begins with a convolutional input layer (kernel size 7, 8 output channels), followed by four convolutional blocks. Each block consists of three residual convolutional sub-blocks and a strided convolution for temporal downsampling. The block strides are 3, 4, 4, and 5, yielding an overall stride of 240 samples (10 ms). Within each residual sub-block, two dilated convolutions with ELU nonlinearities are wrapped by skip connections, enabling a larger receptive field. All convolutional layers use weight normalization. The embedding dimension increases progressively to 16, 32, 64, and finally 160 after each strided convolution. To minimize computational cost, the encoder omits a dedicated output layer. The third block includes two center-aligned convolutions, introducing 10 ms latency; combined with encoder buffering, this results in 20 ms total transmit-side latency. The encoder receptive field spans 14,085 samples.

RVQ is applied with 6 layers, each containing 1,024 codewords, contributing 10 bits per frame. With an encoder frame rate of 100 Hz, this corresponds to 1 kbps per layer, or 6 kbps in total. Each RVQ layer uses projection layers to reduce the 160-dimensional encoder output to 12 dimensions, and then project the selected codeword back to 160 dimensions, with residuals computed in the original space. The RVQ complexity is 19.35 MFLOPS. Post-training, the output projections can be absorbed into the codebooks, storing separate transmit-

Table 2. Latency and computational complexity of the Track 1 baseline system.

	Transmit Side		Receive Side	Overall
	Encoder	RVQ	Decoder	
Buffering Latency (ms)	10	0	0	10
Algorithmic Latency (ms)	10	0	10	20
Compute Complexity (MFLOPS)	377.5	17.05	296.8	691.35

Table 3. Latency and computational complexity of the Track 2 baseline system.

	Transmit Side		Receive Side	Overall
	Encoder	RVQ	Decoder	
Buffering Latency (ms)	10	0	0	10
Algorithmic Latency (ms)	20	0	20	40
Compute Complexity (MFLOPS)	1944.2	38.7	563.3	2546.2

and receive-side versions, which reduces complexity to 17.05 MFLOPS at the cost of increased binary size.

The decoder is a convolutional network consisting of four blocks and a final output layer. Each block begins with a transposed convolution for upsampling, followed by three residual sub-blocks. The strides of the transposed convolutions are 5, 4, 3, and 4, yielding an overall stride of 240. The kernel sizes are set equal to the strides, preventing implicit overlap-add in the transposed convolutions that could otherwise introduce additional latency. As in the encoder, each residual sub-block contains two dilated convolutions with ELU nonlinearities, wrapped by skip connections. The final output layer is a convolution with kernel size 21 and a tanh activation, producing waveform samples in the range [-1, 1]. All convolutional layers use weight normalization. The decoder introduces 10 ms algorithmic latency due to the center-aligned convolution in the first block, and its overall computational complexity is 296.8 MFLOPS (excluding nonlinearities).

We provide a summary of the latency and computational complexity of the Track 1 Baseline system in Table 2. We also provide a detailed design sheet for both Track 1 and Track 2 baseline models with all the hyperparameters, latency and computational complexity calculations in [7]. For a guideline on buffering and algorithmic latency calculations, see the guidance on the challenge website [8].

3.2. Track 2 Baseline Model

The Track 2 baseline model follows the same architectural principles as Track 1 but is trained as a joint codec and enhancement network. It takes noisy and reverberant audio as input and aims to reconstruct the clean reference signal. For reverberant inputs, the clean reference retains the early reverberation components.

The encoder starts with a convolutional input layer (kernel size 7, 8 output channels), followed by five convolutional blocks. Each block contains multiple residual convolutional sub-blocks and a strided convolution for temporal downsampling. The first two blocks include 4 residual sub-blocks each, while the last three contain 3 sub-blocks. The block strides are 2, 2, 3, 4, and 5, resulting in an overall stride of 240 samples (10 ms). Within each residual sub-block, two dilated convolutions with ELU activations are wrapped by skip connections. The embedding dimension increases progressively to 16, 32, 64, 128, and 320 after each strided convolution. The encoder exhibits a buffering latency of 10 ms, an algorithmic latency of 20 ms due to center-aligned convolutions, and a total computational complexity of 1944.2 MFLOPS.

Similar to the Track 1 system, we employ a 6-layer RVQ, with each layer containing 1,024 codewords, contributing 10 bits per frame. Each layer first projects the 320-dimensional encoder output to a 24-dimensional space, selects a codeword, and then projects it back to 320 dimensions, with residuals computed in the original space. The RVQ has a computational complexity of 48 MFLOPS. By absorbing the output projections into the codebooks after training, the complexity can be reduced to 38.7 MFLOPS at the expense of increased binary size.

The Track 2 decoder is a convolutional network composed of five blocks followed by a final output layer. Each block starts with a transposed convolution for upsampling, followed by three residual sub-blocks. The strides of the transposed convolutions are 5, 4, 3, 2, and 2, resulting in an overall stride of 240. Kernel sizes match the strides, as in the Track 1 decoder. The embedding dimension decreases progressively to 96, 48, 24, 12, and 8 after each transposed convolution. The final output layer is a convolution with a kernel size of 21, a single output channel, and a tanh activation, producing waveform samples in the range [–1, 1]. The decoder introduces 20 ms of algorithmic latency due to two center-aligned convolutions in the first block and has an overall computational complexity of 563.3 MFLOPS (excluding nonlinearities).

We provide a summary of the latency and computational complexity of the Track 2 Baseline system in Table 3. For a detailed description of the hyperparameters, as well as the latency and computational complexity calculations for both Track 1 and Track 2 baseline models, please refer to the design sheet [7].

4. TRAINING

We train both systems end-to-end using a combination of adversarial and reconstruction losses. The RVQ codebooks are updated with exponential moving averages, while the projection matrices are optimized via backpropagation. For the codebooks, straight-through gradient estimation is applied. We use Euclidean distance in codeword selection. To stabilize training and prevent rapid fluctuations in the encoder embeddings and codeword selections, we include a commit-

Table 4. Objective evaluation results for Track 1 baseline under clean, noisy, and reverberant conditions.

Bitrate	Clean					Noisy					Reverberant				
	sheet_ssqa	scoreq_ref	audiobox_AE_CE	utmos	pesq	sheet_ssqa	scoreq_ref	audiobox_AE_CE	utmos	pesq	sheet_ssqa	scoreq_ref	audiobox_AE_CE	utmos	pesq
1 kbps	1.84	1.15	3.90	1.44	1.15	1.72	1.29	3.40	1.33	1.11	1.85	1.36	2.94	1.26	1.07
6 kbps	3.84	0.35	5.28	3.23	2.67	3.12	0.82	4.37	2.70	1.81	2.22	1.13	3.43	1.32	1.18

Table 5. Objective evaluation results for Track 2 baseline under clean, noisy, and reverberant conditions.

Bitrate	Clean					Noisy					Reverberant				
	sheet_ssqa	scoreq_ref	audiobox_AE_CE	utmos	pesq	sheet_ssqa	scoreq_ref	audiobox_AE_CE	utmos	pesq	sheet_ssqa	scoreq_ref	audiobox_AE_CE	utmos	pesq
1 kbps	2.07	1.01	3.96	1.37	1.21	1.95	1.15	3.70	1.35	1.18	2.43	1.12	3.55	1.32	1.15
6 kbps	3.55	0.43	5.25	2.97	2.13	2.92	0.75	4.6	2.56	1.73	2.67	0.92	4.25	1.79	1.29

ment loss [9]. During training, we uniformly sample between 1 kbps and 6 kbps using random quantizer dropout, enabling the decoder to operate robustly at both bitrates.

For reconstruction, we employ a multi-scale mel spectrogram loss [10] with window lengths of 64, 128, 256, 512, 1024, and 2048 samples, and corresponding mel bin counts of 10, 20, 40, 80, 160, and 320, respectively.

The adversarial objective follows Encodec [6], using multi-scale feature discriminators operating in the complex STFT domain. We compute STFTs with window lengths of 128, 256, 512, 1024, and 2048 samples, with hop sizes equal to one quarter of the window length. Each discriminator is a convolutional network with weight normalization and Leaky ReLU activations (slope 0.1), using 16 channels in its internal layers. Hinge loss is applied at the output layer. In addition, we apply a feature matching loss on the intermediate discriminator representations.

The loss weights are set to 10 for the commitment loss, 5 for the multi-scale mel-spectrogram loss, 1 for the adversarial loss, and 2 for the feature matching loss.

Each training epoch consists of 10,000 randomly selected utterances from the training set. From these utterances, sliding windows of 62,400 samples are extracted with 50% overlap. Training within an epoch continues until all windows are consumed, so the number of iterations per epoch is not fixed but remains approximately constant. We reserve 1,000 utterances for validation. For Track 2, on-the-fly noise and reverberation augmentation is applied during validation. Although offline augmentation of the validation set could help reduce variance in the validation losses, we did not adopt this approach for simplicity.

We train with a batch size of 64 per GPU, using distributed data parallelism with 6 GPUs for Track 1 and 8 GPUs for Track 2. The learning rate is initialized at 3e-4 and decays at each step by a factor of 0.998. Optimization is performed with RAdam, using betas of 0.9 and 0.999. The Track 1 model is trained for 1,150 epochs and the Track 2 model for 1,325 epochs. For model selection, we use the checkpoint with the lowest multi-scale mel-spectrogram loss on the validation set. While this choice prioritizes ease of implementation, more robust strategies—such as combining objective metrics that correlate better with subjective listening tests—are likely to yield improved results.

We present the baseline results for Track 1 and Track 2 on the open test set in Table 4 and Table 5, respectively. The reported metrics are the official objective measures of the LRAC challenge: SHEET_SSQA, SCOREQ_Ref, Audiobox_AE_CE, UTMOS, and PESQ. Further details on the open test set and these evaluation metrics are available on the 2025 LRAC Challenge objective evaluation page [11].

5. ACKNOWLEDGEMENTS

We thank the Data Team at Cisco Collaboration AI for their support in curating and augmenting the training datasets used in the baseline development. In particular, we acknowledge the contributions of Ivana Balic, Laura Lechler, Daniel Arismendi, Ayoub Zaidour, and James Taylor.

6. REFERENCES

[1] Collaboration AI, Cisco Systems. 2025 LRAC Challenge data generation repository, 2025. Available: `https://github.com/cisco-open/lrac_data_generation` (accessed: 2025-09-30).

[2] Collaboration AI, Cisco Systems. 2025 LRAC baseline development repository, 2025. Available: `https://github.com/cisco-open/espnet` (accessed: 2025-09-30).

[3] Shinji Watanabe, Takaaki Hori, Shigeki Karita, Tomoki Hayashi, Jiro Nishitoba, Yuya Unno, Nelson Enrique Yalta Soplin, Jahn Heymann, Matthew Wiesner, Nanxin Chen, Adithya Renduchintala, and Tsubasa Ochiai. ESPnet: End-to-end speech processing toolkit. In *Proceedings of Interspeech*, pages 2207–2211, 2018.

[4] Benjamin Elizalde, Soham Deshmukh, Mahmoud Al Ismail, and Huaming Wang. Clap learning audio concepts from natural language supervision. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023.

[5] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. Soundstream: An end-to-end neural audio codec. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:495–507, 2022.

[6] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *Transactions on Machine Learning Research*, 2023. Featured Certification, Reproducibility Certification.

[7] Collaboration AI, Cisco Systems. 2025 LRAC baseline model design sheet, 2025. Available: `https://github.com/cisco-open/espnet/blob/master/egs2/lrac/LRAC-Challenge-Baseline-Models-Design-Sheet.xlsx` (accessed: 2025-09-30).

[8] Collaboration AI, Cisco Systems. 2025 LRAC challenge latency calculation guidelines, 2025. Available: `https://lrac.short.gy/latency-guidelines` (accessed: 2025-09-30).

[9] Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[10] Ryuichi Yamamoto, Eunwoo Song, and Jae-Min Kim. Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6199–6203, 2020.

[11] Collaboration AI, Cisco Systems. 2025 LRAC challenge - objective evaluation, 2025. Available: `https://lrac.short.gy/evaluation#objective-evaluation` (accessed: 2025-09-30).