



TS3-Codec: Transformer-Based Simple Streaming Single Codec

Haibin Wu¹, Naoyuki Kanda¹, Sefik Emre Eskimez¹, Jinyu Li¹

¹Microsoft, USA

Abstract

Neural audio codecs (NACs) have garnered significant attention as key technologies for audio compression as well as audio representation for speech language models. While mainstream NACs are predominantly convolution-based, the performance of NACs with a purely transformer-based, and convolution-free architecture remains unexplored. This paper introduces TS3-Codec, a **T**ransformer-Based **S**imple **S**treaming **S**ingle Codec. TS3-Codec consists of only a stack of transformer layers and linear layers, offering greater simplicity and expressiveness by fully eliminating convolution layers that require careful hyperparameter tuning and large computations. Under the streaming setup, TS3-Codec achieves comparable or superior performance compared to the codec with state-of-the-art convolution-based architecture while requiring only 12% of the computation and 77% of bitrate. Furthermore, it significantly outperforms the convolution-based codec when using similar computational resources.

Index Terms: Transformer, neural audio codec, streaming

1. Introduction

Neural audio codec (NAC) is a technique to compress audio signals into a sequence of discretized codes for efficient data storage and transmission [1–5]. More recently, NAC has also gained significant attention as a key technology for speech language modeling (SLM) [6–9]. By converting continuous audio into discrete codes, large language modeling (LLM) techniques—already highly successful in text processing—is able to be applied to versatile speech processing [10].

Numerous high-performance NACs have been proposed, addressing various aspects, e.g. better audio quality, bitrate efficiency, and low computational cost. Most models rely on convolutional layers as the dominant architecture, with only a few [11, 12] incorporating transformers (or self-attention mechanism) [13] as intermediate layers within the convolutional encoder-decoder framework. However, the performance of a purely transformer-based and convolution-free architecture in NACs remains unexplored. This study aims to fill the existing gap by developing a NAC exclusively based on transformer models. It leverages the benefits of transformers, such as simplicity in model design and enhanced computational efficiency when compared to convolution-based models.

When the NAC is used for SLMs, the following properties are particularly important. (1). **Streaming:** Full-duplex communication, where users and machines can speak and respond simultaneously [12, 14]. To enable seamless real-time interactions, the codec must support streaming processing, allowing it to encode user speech and generate speech response with low latency. (2). **Single codebook:** A single codebook-

based model is preferable to a multiple-codebook-based model, such as residual vector quantization (RVQ) [15, 16], because the latter introduces additional complexity to the architecture of SLMs [7, 17, 18]. (3). **Low computation:** Low-computation NACs enable faster encoding and decoding, reducing computational demands and leaving more computation resources available for SLMs. (4). **Low token rate:** Long sequences generally make LLM training slow and unstable. Therefore, it is preferable to use low-token-rate NAC models for SLM.

This paper introduces TS3-Codec (Transformer-Based Simple Streaming Single Codec), the first attempt to develop a convolution-free, transformer-only NAC. TS3-Codec consists of only a stack of transformer layers with a few linear layers, offering greater simplicity and expressiveness by fully eliminating convolution layers that require careful hyperparameter tuning and large computations. The proposed TS3-Codec offers several advantages, namely, streaming capability, low computational requirements, low bitrate, and a single codebook design while maintaining high audio quality. In the streaming setup, TS3-Codec delivers comparable or superior performance than convolution-based codecs with just 12% of the computation and 77% of bitrate. TS3-Codec also achieves significantly better audio quality when using the same computational resources.

2. Related work

Neural audio codec models typically consist of an encoder, a vector quantization (VQ) module, and a decoder. The encoder downsamples the time-domain audio to extract frame-wise audio features. The vector quantization module converts each frame-wise audio feature into discrete tokens. These discrete tokens can be used for efficient transmission or as input for SLMs. Finally, the decoder reconstructs the time-domain audio signal from the discrete tokens. To meet the requirements of SLMs for real-time conversational agents with full-duplex mode, a suitable codec is better to support streaming, low computational complexity, a single codebook, and a low token rate.

Streaming: Encodec [16] is a pioneer in achieving real-time, high-quality audio coding by utilizing a streaming encoder-decoder architecture with residual vector quantization. To enable full-duplex operation in SLMs, Mimi [12] is proposed as a streaming RVQ codec with a 12.5 Hz token rate. It employs techniques, e.g. semantic distillation, discriminator-only training without reconstruction loss, and the integration of transformer layers between the encoder and decoder backbone to enhance feature modeling. Mimi achieves a low bitrate of 1.1 kbps while maintaining very high speech quality.

Single codebook A single codebook-based model is preferable to a multiple-codebook-based model, because it offer the simplicity design of the SLMs. TiCodec [19] and SingleCodec [20]

are designed to encode speech using fewer tokens by disentangling time-invariant global information (e.g., speaker timbre and acoustic environment) into a single utterance-level vector, while representing time-varying information (e.g., phonetic content) with a single stream of frame-level tokens. The time-invariant utterance embeddings and frame-level token sequences are then combined to reconstruct the audio. However, one limitation is their reliance on global utterance-level features for decoding, which restricts their streaming capability. WavTokenizer [11] employs several techniques, including codebook size optimization, k-means initialization for codebook embeddings, the Vocos decoder [21], to achieve high fidelity in a single-codebook low-bitrate codec. BigCodec [22] achieves exceptionally high reconstruction quality at a low bitrate of 1.04 kbps by scaling the convolutional model size to 160M parameters, making it a strong convolutional baseline.

Uniqueness and contributions of this work: This paper presents the first transformer-only codec architecture designed for streaming, featuring low computational complexity and a simple model design. Previous works have utilized transformers [11, 12] solely as intermediate layers within predominantly convolutional backbones for feature engineering. Furthermore, previous streaming codecs [12, 16, 23] rely on residual vector quantization (RVQ), and single-codebook codecs [11, 19, 20, 22] have lacked a streaming design. Our work is the first to introduce a single-codebook codec specifically designed for streaming. This work is also the first to explore single codec designs with 65k and 130k codebook sizes.

3. TS3-Codec

3.1. Model architecture

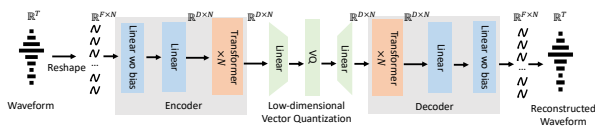


Figure 1: The framework of the proposed TS3-Codec. The transformer layer employs a sliding window attention on the left context, enabling the model to function in a streaming manner with linear (rather than quadratic) complexity relative to the waveform length.

TS3-Codec follows the conventional NAC structure, which includes an encoder, a quantizer, and a decoder, as shown in Figure 1. The model is trained using the generative adversarial network (GAN) framework [24].

The encoder consists of two linear layers and a stack of transformer layers with sliding window attention only on the left context. The input audio signal with a shape of \mathbb{R}^T , where T is the signal length, is first reshaped into a two-dimensional tensor with a shape of $\mathbb{R}^{F \times N}$, where F is a window-size and $N = \frac{T}{F}$ is a number of windowed frames.¹ Two linear layers without activations (the one close to the waveform is without bias, and the second linear is with bias) are then applied to convert the reshaped windowed frames into a shape of $\mathbb{R}^{D \times N}$. Finally, a stack of Transformer layers with an embedding size of D is applied to output an encoder embedding with a shape of $\mathbb{R}^{D \times N}$. The Transformer layer employs a sliding window with the size of either 16 or 32 on the left-context for the self-attention operation, meaning only a fixed number of previous frames are considered. This design not only ensures that the

¹ For simplicity, we assume T is divisible by F . We can pad the original signal to satisfy this assumption.

transformer’s computational complexity does not blow up to the quadratic with respect to N , but also improves the generalization ability of the model for an audio longer than the training data. If not specified, the encoder has 8 transformer layers, 16 attention heads, and a feed-forward dimension of 4096.

After the encoder module, a factorized VQ layer [25] discretizes the encoder output in a low-dimensional space. We used a codebook with a codebook dimension of 8 or 16, and a codebook size of 65,536 or 131,072.

The decoder has a symmetric structure with the encoder, consisting of a stack of transformer layers with two linear layers. Unless otherwise specified, the decoder uses the same parameters as the encoder, namely, 8 transformer layers, 16 attention heads, and a feed-forward dimension of 4096. The output from the transformer layer has a shape of $\mathbb{R}^{D \times N}$. It is then converted into a tensor with a shape of $\mathbb{R}^{F \times N}$ based on two linear layers without activations (the first one is with bias, and the second one doesn’t have bias). Finally, the tensor is reshaped into a single-dimensional tensor with a shape of \mathbb{R}^T .²

3.2. Design principle

We chose transformers as the backbone for several potential advantages over convolution-based architectures. (1). Convolutional layers are with parameter efficiency and reusability. On the other hand, for models of similar parameter sizes, convolutions typically require significantly more computation than transformers [27]. Surprisingly, we discovered that the state-of-the-art neural audio codec model, BigCodec [22], with 160M parameters, has a computational cost comparable to that of a **1-billion-parameter** transformer.³ The computational cost increases significantly as convolutional neural audio codec models are scaled up, making it impractical to scale these models further. (2). Convolutions have inherent biases. Convolutions apply fixed weighted-sum weights across all intermediate feature maps across different time stamps, whereas transformers use self-attention, dynamically determining weights tailored to each feature map. Transformers also incorporate positional embeddings, enabling distinct embeddings to be added to different feature maps at different time stamps. (3). Transformers offer simplicity in model design. Unlike convolutions, which require careful selection of kernels and up- and down-sampling mechanisms due to their inherent biases (they are sensitive to hyperparameter settings), transformers could avoid these complexities.

3.3. Training objectives

Similar to most NACs, TS3-Codec is trained based on the GAN framework. We use multiple losses following BigCodec [22]. (1). Reconstruction Loss: We adopt a multi-scale mel-spectrogram reconstruction loss, calculated using the L_1 distance in the spectral domain across multiple scales. The mel-spectrogram is closely related to perceptual audio quality. (2). Least-square GAN loss [28]: Following the BigCodec [22], we employ two types of discriminators to train our model. The first is the Multi-Period Discriminator (MPD), adapted from HiFi-GAN [29], which captures various periodic patterns in speech signals. The second is the Multi-Scale Short-Time Fourier Transform (MS-STFT) Discriminator, which is used in En-

² Our decoder is largely inspired by the WaveNet vocoder [26], which demonstrated that up-sampling at the final layer using a simple linear layer without overlapping for reconstructed signals is sufficient to reconstruct high-quality speech, as opposed to progressive up-sampling through stacked convolution layers. We refer to the implementation in https://github.com/wetdog/wavenext_pytorch

³ At a bitrate of 1k, TS3-Codec (1.6B parameters) requires 60.52G MACs, while BigCodec (160M parameters) requires 61.1G MACs. At a bitrate of 0.6k, TS3-Codec (1.2B parameters) uses 36.34G MACs, while BigCodec (160M parameters) uses 39.6G MACs.

Codec [16]. We use the same discriminator configurations as BigCodec. (3). Feature loss: we use the L_1 feature matching loss for the discriminator features. (4). VQ loss: The codebook is trained using the L_1 loss, which is calculated between the features before and after quantization, employing a stop-gradient operation [25]. Following the approach used in BigCodec, we do not use a moving average to update the codebook. To prevent the encoder output from becoming excessively large, a commitment loss with a loss weight as 0.25 is introduced.

The loss weights for the reconstruction, GAN, feature losses are set to 15.0, 1.0, and 1.0, respectively. Unlike the official BigCodec implementation, we set the VQ loss proportional to the codebook size: 4.0 for a codebook size of 8192, 32.0 for a codebook size of 65,536. This configuration yielded the best results in our preliminary experiments.

4. Experimental setup

4.1. Data

We used Libri-light [30], which contains 60K hours of speech, for training. For evaluation, we used the Librispeech ‘test-clean’ [31], which consists of 2620 utterances from 40 speakers.

4.2. Evaluation Metrics

Complexity and the extracted tokens’ property: (1). Paras: The number of parameters of the codec model. (2). Multiply-Accumulate Operations (MACs): MACs refer to the fundamental computational operations in NAVs. In this paper, we get MACs using a 1-second audio utterance with PyFlops⁴. For modules not supported by PyFlops, such as streaming transformers, we calculated them manually. (3). Bits Per Second (BPS): measures the number of bits transmitted per second. (4). Token rate: A metric that indicates the number of tokens required for each second of encoded audio. (5). Frame rate: A metric that shows the number of frames needed to encode each second of audio.

Intelligibility: (1). Word Error Rate (WER): We employed a HuBERT-based [32] speech recognition model⁵ to calculate the WER. (2). Short-Time Objective Intelligibility (STOI): assess speech intelligibility by comparing short-time temporal envelopes of clean and degraded speech signals.

Distortion: (1). Mel Cepstral Distortion (MCD): MCD measures the difference between two mel-frequency cepstral coefficients (MFCCs) sequences. It is commonly employed in speech synthesis to evaluate the quality of generated speech. (2). Perceptual Evaluation of Speech Quality (PESQ): PESQ assesses the quality of speech signals by comparing a degraded speech signal with a reference signal, providing a score that predicts human perception of speech quality. Wide-band PESQ is used.

Speaker similarity (SPK SIM): We used the WavLM-based speaker verification model⁶ [33] to get the cosine similarity between original and reconstructed utterances.

Naturalness We used UTMOS⁷ [34]. UTMOS is a neural network-based metric that predicts the Mean Opinion Score (MOS) for the naturalness of speech by learning from pairs of human subjective scores and corresponding utterances. UTMOS is widely used as a metric that is highly correlated with human preferences on codec reconstructed utterances [3, 11].

⁴ <https://github.com/sovrasov/flops-counter.pytorch/tree/master>

⁵ <https://huggingface.co/facebook/hubert-large-ls960-ft>

⁶ https://github.com/microsoft/UniSpeech/tree/main/downstreams/speaker_verification

⁷ <https://github.com/sarulab-speech/UTMOS22>

Table 1: Comparison between baseline codecs. *SEM* represents semantic distillation. *RVQ* and *single* means residual and single vector quantization, respectively. *SA* means self-attention.

Codec	SEM	Streaming	VQ type	Architecture
Encodec [16]	✗	✓	RVQ	Conv + LSTM
DAC [35]	✗	✗	RVQ	Conv
SpeechTokenizer [36]	✓	✗	RVQ	Conv + LSTM
Mimi [12]	✓	✓	RVQ	Conv + Transformer
BigCodec [22]	✗	✗	Single	Conv + LSTM
WavTokenizer [11]	✗	✗	Single	Conv + LSTM + SA

4.3. Baselines

Official checkpoints: The comparison of these 6 codec models with official checkpoints is shown in Table 1. (1). Encodec [16]: We used the default 24k Hz model⁸, with a bitrate setting as 1.5 kbps, which aligns closely with our model’s bitrate. During reconstruction, we first upsampled the utterance to 24 kHz, then performed reconstruction using Encodec, and finally downsampled it to obtain the reconstructed utterances at 16 kHz. (2). DAC [35]: We used the 24k Hz model⁹, and adopted similar procedures as for Encodec. (3). SpeechTokenizer [36]: We used the speechookenizer.snake model¹⁰. We got two bitrate settings, 1k and 0.5k bps, to align with our bitrate settings. (4). Mimi [12]: We used the official checkpoint¹¹ to obtain two bitrate settings: 1.1 kbps and 0.6875 kbps, which align with our model’s bitrate settings. (5). BigCodec [22]: The official checkpoint¹² with a bitrate of 1.04k bps was used. (6). WavTokenizer [11]: Two official checkpoints¹³, WavTokenizer-large-600-24k-4096 (0.52k bps) and WavTokenizer-large-320-24k-4096 (0.975k bps) were used.

Reproduced models based on official implementation Our primary baseline is BigCodec, as it is the current state-of-the-art single-codebook low-bps codec. We reproduced BigCodec based on its official implementation and further developed a streaming version (denoted as BigCodec-S) by replacing all the non-causal convolution operations with causal convolution operations, and removing the upsampling/downsampling operations at the snake activation function¹⁴. BigCodec-S was trained on the same datasets as TS3-Codec for a fair comparison. We conducted extensive hyperparameter tuning (e.g., codebook size, loss weights) on BigCodec-S. As a result, at a similar bitrate, BigCodec-S trained using only Libri-light outperformed Mimi, the current state-of-the-art RVQ-based streaming codec model, on the majority of evaluation metrics.

Training configurations All models were trained on 16 NVIDIA V100 32G GPUs. For TS3-Codec (X1), (X3) and (X5), the codebook size is 65536, and (X2) and (X4) are with codebook size of 131,072. The batch size was adjusted to maximize GPU memory usage for training BigCodec-S and TS3-Codec. We employed AdamW [37] as the optimizer, with the moving average coefficients β_1 and β_2 set to 0.8 and 0.9, respectively. A scheduled learning rate was used, linearly declining from 1e-4 to 1e-5 for BigCodec-S and from 2e-4 to 2e-5 for TS3-Codec. We performed 1k learning rate warmup steps, and all models were trained for 500k steps.

⁸ <https://github.com/facebookresearch/encodec>

⁹ <https://github.com/descriptinc/descript-audio-codec>

¹⁰ <https://github.com/ZhangXInFD/SpeechTokenizer>

¹¹ <https://huggingface.co/kyutai/mimi>

¹² <https://github.com/Aria-K-Alethia/BigCodec>

¹³ <https://github.com/jishengpeng/WavTokenizer>

¹⁴ Removing the upsampling/downsampling operations yielded slightly better results compared to using causal upsampling/downsampling operations in the streaming setting.

Table 2: Comparison between codecs under 1000 bps. BigCodec and BigCodec-S denote the non-streaming and streaming versions, respectively.

Model Tag	Streaming	Bitrate	Codebook Layer	Frame Rate	Token Rate	MACs	Paras	WER↓	STOI↑	PESQ↑	MCD↓	SPK-SIM↑	UTMOS↑
Ground Truth	-	-	-	-	-	-	-	2.0	1.000	4.64	0.00	1.00	4.09
DAC (A)	✗	1500	2	75	150	55.6G	74.1M	7.2	0.829	1.48	4.83	0.47	1.68
SpeechTokenizer (B1)	✗	1000	2	50	100	17.1G	103.7M	3.9	0.768	1.21	6.30	0.33	2.32
BigCodec (C)	✗	1040	1	80	80	67.1G	159.4M	2.8	0.935	2.68	3.01	0.84	4.11
WavTokenizer (D1)	✗	975	1	75	75	6.3G	80.6M	6.8	0.886	2.05	4.00	0.59	3.89
Encodec (E1)	✓	1500	2	75	150	5.6G	14.9M	4.9	0.845	1.56	4.32	0.60	1.58
Mimi (F1)	✓	1100	8	12.5	100	8.1G	79.3M	3.0	0.905	2.22	3.81	0.73	3.60
BigCodec-S (G1)	✓	1040	1	80	80	7.1G	21.8M	4.6	0.888	1.96	3.80	0.56	3.41
BigCodec-S (G2)	✓	1040	1	80	80	61.1G	159.9M	3.8	0.906	2.17	3.52	0.65	3.73
TS3-Codec (X1)	✓	800	1	50	50	7.6G	203.6M	3.6	0.909	2.22	3.52	0.68	3.85
TS3-Codec (X2)	✓	850	1	50	50	7.6G	203.6M	3.6	0.910	2.23	3.50	0.68	3.84

Table 3: Comparison between different codec models under around 600 bps.

Model Tag	Streaming	Bitrate	Codebook Layer	Frame Rate	Token Rate	MACs	Paras	WER↓	STOI↑	PESQ↑	MCD↓	SPK-SIM↑	UTMOS↑
Ground Truth	-	-	-	-	-	-	-	2.0	1.000	4.64	0.00	1.00	4.09
SpeechTokenizer (B2)	✗	500	1	50	50	17.1G	103.7M	4.9	0.675	1.12	8.38	0.17	1.34
WavTokenizer (D2)	✗	520	1	40	40	3.4G	80.9M	8.0	0.868	1.88	4.32	0.57	3.77
Encodec (E2)	✓	750	1	75	75	5.6G	14.9M	29.0	0.770	1.23	5.66	0.25	1.25
Mimi (F2)	✓	687.5	5	12.5	62.5	8.1G	79.3M	4.0	0.872	1.82	4.40	0.58	3.27
BigCodec-S (G3)	✓	640	1	40	40	4.6G	21.8M	5.9	0.870	1.78	4.16	0.50	3.20
BigCodec-S (G4)	✓	640	1	40	40	39.6G	160.5M	5.4	0.889	1.96	3.97	0.58	3.68
TS3-Codec (X3)	✓	640	1	40	40	6.2G	204.4M	4.9	0.893	2.01	3.81	0.61	3.69
TS3-Codec (X4)	✓	680	1	40	40	6.2G	204.4M	4.5	0.897	2.06	3.75	0.63	3.73

5. Experimental results

5.1. Results at approximately 1000 bps

In Table 2, four non-streaming baselines, (A)–(D1), are evaluated using results derived from their official checkpoints. We observe that among the four non-streaming baselines, BigCodec (C) demonstrates the best performance at approximately 1000 bps, surpassing other codec models by a significant margin across all metrics. This is the reason we selected BigCodec as the basis for developing a streaming version as our baseline.

Next, by comparing the six streaming models, we observe the followings: (1) TS3-Codec models (X1 and X2 with sliding window size of 32) perform the best for STOI, PESQ, MCD, UTMOS, and the second best for WER and SPK-SIM. (2). Mimi performs the best for WER, probably because of their inclusion of semantic distillation. We listened to utterances generated by Mimi, the audio quality is worse than the TS3-Codec.

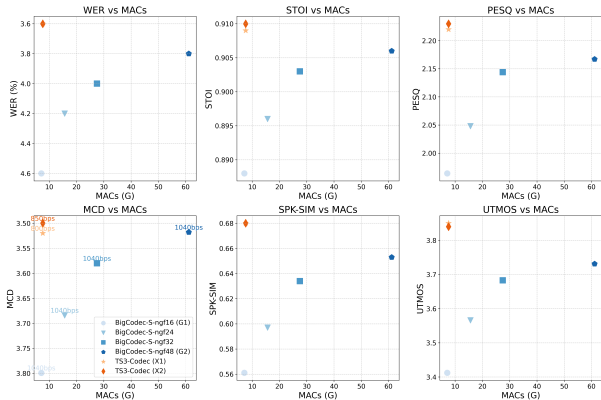


Figure 2: Comparison between BigCodec-S and TS3-Codec (Bitrate \approx 1000 bps). To enhance visualization, the y-axes for WER and MCD are inverted, so that model points in the upper-left corner exhibit the best performance with the least computational cost. ngf is a factor related to the model size of BigCodec-S.

Figure 2 compare TS3-Codec and BigCodec under the streaming settings, at a bitrate of approximately 1000 bps (for

600 bps, we have a similar observation.). Note that at the 1000 bps bitrate setting, TS3-Codec operates at 800 or 850 bps, significantly lower than its counterpart, BigCodec-S, which operates at 1040 bps. Comparing TS3-Codec and BigCodec-S, we can observe that: (1) Under similar computational budgets, the proposed TS3-Codec always outperforms BigCodec-S significantly across all metrics (e.g. in 640bps, UTMOS scores for TS3-Codec and BigCodec-S are 3.85 and 3.41, WERs are 3.6 and 4.6). (2) TS3-Codec achieves comparable or superior performance to BigCodec with significantly less computation. For example, TS3-Codec, with just 12% of the computation and 77% of the bitrate of its BigCodec counterparts, similarly delivers comparable or superior results.

5.2. Results at approximately 600 bps

Table 3 presents the results for bitrates around 600 bps. We observe the following: (1). Under the streaming setup, TS3-Codec models (X3 and X4 with sliding window size of 16) achieve the best performance in STOI, PESQ, MCD, SPK-SIM, and UTMOS, while securing the second-best WER. TS3-Codec also outperforms the two non-causal baselines across all metrics. (2). Mimi achieves the best WER, likely due to its incorporation of semantic distillation during training. (3). SpeechTokenizer performs poorly in PESQ, MCD, SPK-SIM, and UTMOS metrics, though its WER is relatively decent. Upon listening, some male voices are distorted to sound like female robotic speech, yet the content remains intelligible. This may be due to SpeechTokenizer’s use of semantic distillation during training, allowing the first-stream tokens to encode sufficient information for content recognition.

6. Conclusion

This paper introduces TS3-Codec, the first transformer-only NAC model designed for streaming processing with low computational complexity and a single codebook. Compared to BigCodec, the state-of-the-art convolutional codec, TS3-Codec achieved similar or better performance with just 12% of the computation and 77% of the bitrate, and performed significantly better under similar computational settings. TS3-Codec sets a new direction for simple and efficient streaming NACs.

7. References

- [1] M. Kim and J. Skoglund, "Neural speech and audio coding," *arXiv preprint arXiv:2408.06954*, 2024.
- [2] H. Wu, H.-L. Chung *et al.*, "Codec-SUPERB: An in-depth analysis of sound codec models," in *Findings of ACL 2024*. ACL, Aug. 2024, pp. 10 330–10 348.
- [3] J. Shi *et al.*, "Espnet-codec: Comprehensive training and evaluation of neural codecs for audio, music, and speech," *arXiv preprint arXiv:2409.15897*, 2024.
- [4] H. Wu *et al.*, "Codec-superb@ slt 2024: A lightweight benchmark for neural audio codec models," *arXiv preprint arXiv:2409.14085*, 2024.
- [5] P. Mousavi *et al.*, "Dasb—discrete audio and speech benchmark," *arXiv preprint arXiv:2406.14294*, 2024.
- [6] Z. Borsos *et al.*, "Audiolm: a language modeling approach to audio generation," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 31, pp. 2523–2533, 2023.
- [7] C. Wang, S. Chen, Y. Wu, Z. Zhang, L. Zhou, S. Liu, Z. Chen, Y. Liu, H. Wang, J. Li, L. He, S. Zhao, and F. Wei, "Neural codec language models are zero-shot text to speech synthesizers," *arXiv preprint arXiv:2301.02111*, 2023.
- [8] X. Wang *et al.*, "Speechx: Neural codec language model as a versatile speech transformer," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024.
- [9] S. Arora, K.-W. Chang, C.-M. Chien, Y. Peng, H. Wu, Y. Adi, E. Dupoux, H.-Y. Lee, K. Livescu, and S. Watanabe, "On the landscape of spoken language models: A comprehensive survey," *arXiv preprint arXiv:2504.08528*, 2025.
- [10] H. Wu *et al.*, "Towards audio language modeling—an overview," *arXiv preprint arXiv:2402.13236*, 2024.
- [11] S. Ji, Z. Jiang, X. Cheng, Y. Chen, M. Fang, J. Zuo, Q. Yang, R. Li, Z. Zhang, X. Yang *et al.*, "Wavtokenizer: an efficient acoustic discrete codec tokenizer for audio language modeling," *arXiv preprint arXiv:2408.16532*, 2024.
- [12] A. Défossez, L. Mazaré, M. Orsini, A. Royer, P. Pérez, H. Jégou, E. Grave, and N. Zeghidour, "Moshi: a speech-text foundation model for real-time dialogue," *arXiv preprint arXiv:2410.00037*.
- [13] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.
- [14] T. A. Nguyen, E. Kharitonov, J. Copet, Y. Adi, W.-N. Hsu, A. Elkahky, P. Tomasello, R. Algayres, B. Sagot, A. Mohamed, and E. Dupoux, "Generative spoken dialogue language modeling," *Transactions of the Association for Computational Linguistics*, vol. 11, pp. 250–266, 2023.
- [15] 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.
- [16] A. Défossez, J. Copet, G. Synnaeve, and Y. Adi, "High fidelity neural audio compression," *arXiv preprint arXiv:2210.13438*, 2022.
- [17] L. Yu, D. Simig, C. Flaherty, A. Aghajanyan, L. Zettlemoyer, and M. Lewis, "Megabyte: Predicting million-byte sequences with multiscale transformers," *Advances in Neural Information Processing Systems*, vol. 36, pp. 78 808–78 823, 2023.
- [18] D. Yang, J. Tian, X. Tan, R. Huang, S. Liu, X. Chang, J. Shi, S. Zhao, J. Bian, X. Wu *et al.*, "Uniaudio: An audio foundation model toward universal audio generation," *arXiv preprint arXiv:2310.00704*, 2023.
- [19] Y. Ren, T. Wang, J. Yi, L. Xu, J. Tao, C. Y. Zhang, and J. Zhou, "Fewer-token neural speech codec with time-invariant codes," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 12 737–12 741.
- [20] H. Li, L. Xue, H. Guo, X. Zhu, Y. Lv, L. Xie, Y. Chen, H. Yin, and Z. Li, "Single-codec: Single-codebook speech codec towards high-performance speech generation," *arXiv preprint arXiv:2406.07422*, 2024.
- [21] H. Siuzdak, "Vocos: Closing the gap between time-domain and fourier-based neural vocoders for high-quality audio synthesis," *arXiv preprint arXiv:2306.00814*, 2023.
- [22] D. Xin, X. Tan, S. Takamichi, and H. Saruwatari, "Bigcodec: Pushing the limits of low-bitrate neural speech codec," *arXiv preprint arXiv:2409.05377*, 2024.
- [23] Y.-C. Wu, I. D. Gebru, D. Marković, and A. Richard, "Audiodec: An open-source streaming high-fidelity neural audio codec," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.
- [24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.
- [25] A. Van Den Oord, O. Vinyals *et al.*, "Neural discrete representation learning," *Advances in neural information processing systems*, vol. 30, 2017.
- [26] T. Okamoto, H. Yamashita, Y. Ohtani, T. Toda, and H. Kawai, "Wavenext: Convnext-based fast neural vocoder without istft layer," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2023, pp. 1–8.
- [27] Y. Zhang, "Hearing the agi from gmm hmm to gpt 4o," 2024, ITI at CMU. [Online]. Available: <https://www.youtube.com/watch?v=pRUrO0x637A>
- [28] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2794–2802.
- [29] J. Kong, J. Kim, and J. Bae, "Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis," *Advances in neural information processing systems*, vol. 33, pp. 17 022–17 033, 2020.
- [30] J. Kahn, M. Riviere, W. Zheng, E. Kharitonov, Q. Xu, P.-E. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen *et al.*, "Libri-light: A benchmark for asr with limited or no supervision," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7669–7673.
- [31] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.
- [32] W.-N. Hsu, B. Bolte, Y.-H. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "Hubert: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 29, pp. 3451–3460, 2021.
- [33] S. Chen, C. Wang, Z. Chen, Y. Wu, S. Liu, Z. Chen, J. Li, N. Kanda, T. Yoshioka, X. Xiao *et al.*, "Wavlm: Large-scale self-supervised pre-training for full stack speech processing," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1505–1518, 2022.
- [34] T. Saeki, D. Xin, W. Nakata, T. Koriyama, S. Takamichi, and H. Saruwatari, "Utmos: Utokyo-sarulab system for voicemos challenge 2022," *arXiv preprint arXiv:2204.02152*, 2022.
- [35] 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.
- [36] X. Zhang, D. Zhang, S. Li, Y. Zhou, and X. Qiu, "Spechtoktokenizer: Unified speech tokenizer for speech large language models," *arXiv preprint arXiv:2308.16692*, 2023.
- [37] I. Loshchilov, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.