Large Language Model-Driven Audio Codec is a Few-Shot Audio Task Learner

Anonymous Author(s)
Affiliation
Address
email

Abstract

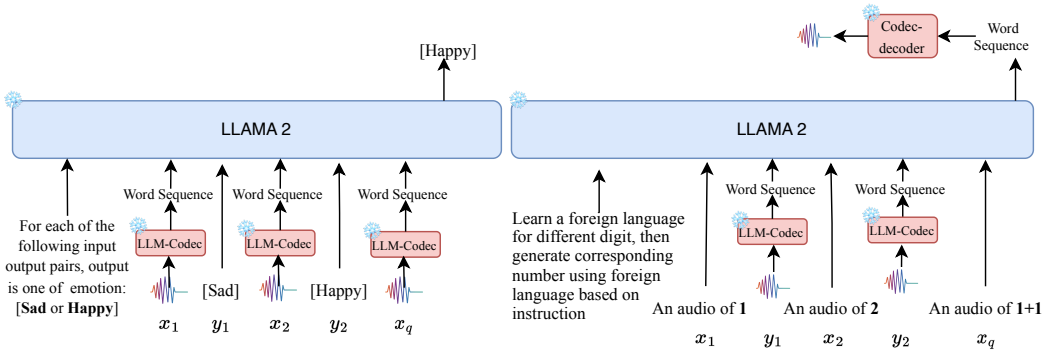
Large Language models (LLMs) have demonstrated supreme capabilities in textual understanding and generation, but cannot be directly applied to cross-modal tasks without fine-tuning. This paper proposes a cross-modal in-context learning approach, empowering the frozen LLMs to achieve multiple audio tasks in a few-shot style without any parameter update. Specifically, we propose a novel LLM-driven audio codec model, LLM-Codec, which transfers the audio modality into textual space by representing audio tokens with words or sub-words from the LLM vocabulary, while maintaining high audio reconstruction quality. The key idea is to reduce the modality heterogeneity between text and audio by compressing the audio modality into the well-trained textual space of LLMs. Thus, the audio representation can be viewed as a new *foreign language*, and LLMs can learn the new *foreign language* with several demonstrations. In experiments, we investigate the performance of the proposed approach across multiple audio understanding and generation tasks, *e.g.* speech emotion classification, audio classification, text-to-speech generation, speech enhancement, etc. Experimental results show that LLMs equipped with the LLM-Codec, prompted by only a few examples, can perform effectively in simple scenarios, validating our cross-modal in-context learning approach. To facilitate research on few-shot audio task learning and multi-modal LLMs, we have open-sourced the LLM-Codec model. [1]

1 Introduction

Large language models (LLMs) (*e.g.*, GPT-4 [2], LLAMA [35]) have become increasingly versatile and effective in handling diverse and complex Natural Language Processing (NLP) tasks as they scale in model size and training data. It is worth noting that the in-context learning ability of LLMs can be used to solve unseen tasks, *e.g.*, we can provide instructions along with a few demonstrations, enabling LLMs to learn and solve new tasks. The success of LLMs inspires the development of multi-modal LLMs, naturally leading to the idea of empowering their auditory capabilities to tackle audio-related tasks. There have been notable advancements in extending the capabilities of LLMs to tackle audio understanding tasks by combining the pre-trained audio encoder (*e.g.* Whisper encoder [30]) and LLMs. For instance, models like WavLLM [15], SALMONN [34], and Qwen-audio [8] propose training multi-modal LLMs by integrating a pre-trained audio encoder, a trainable adaptor, and pre-trained LLMs. They try to align the audio and text modalities by updating the adaptor or fine-tuning the LLMs with LORA [14]. However, previous works have limitations: (1) they primarily focus on expanding LLMs to solve specific audio tasks without leveraging in-context learning for unseen audio tasks; (2) they do not support audio generation tasks, which limits their applicability; (3) they require large-scale audio data for aligning audio and text modalities, increasing the burden of model training and data collection.

[1] https://github.com/LLM-Codec/LLM-Codec

Submitted to 38th Conference on Neural Information Processing Systems (NeurIPS 2024). Do not distribute.



(a) Speech emotion classification with in-context learning

(b) Text-to-speech generation with in-context learning

Figure 1: This figure illustrates the framework of the proposed approach for performing speech emotion classification and simple text-to-speech generation tasks. For each task, we prepare the instruction, demonstrations (e.g., $\{x_1, y_1, x_2, y_2\}$), and the query x_q. The LLAMA 2 model is then asked to predict the corresponding result y_q. Here, y_q can be either text or audio.

In this study, we propose a cross-modal in-context learning approach, empowering the frozen LLMs to solve any user-defined audio tasks based on a few demonstrations without any parameter update. To achieve this, we introduce a vector quantization audio codec model, named LLM-Codec, that maps the audio modality to the token space of frozen LLMs (*e.g.*, LLAMA 2 [35]). Our motivation is to reduce modality heterogeneity between audio and text by compressing audio data into the token space of LLMs. Given that the compressed audio and text modalities share a vocabulary, the compressed audio sequence can be treated as a new *foreign language*, which LLMs can learn from a few demonstration samples. Moreover, since LLMs are pre-trained on large-scale data and have discovered numerous token sequence patterns, they are well-positioned to generalize to this new *foreign language*. Figure 1 illustrates the integration of the proposed LLM-Codec with LLAMA 2 models for performing various audio tasks.

The proposed LLM-Codec aims to compress audio data into a lexical word sequence. A desired LLM-Codec should exhibit the following properties: (1) **Completeness** [12]: it should recover compressed audio with minimal loss. (2) **Compactness**: it should encode the audio into fewer-token sequences. (3) **Semantic richness**: it should encode audio into semantically rich token sequences, making them easier for pre-trained LLMs to recognize. Thus, we propose a semantic-guided multi-scale residual vector quantization (RVQ) based codec. Specifically, the codec model consists of three residual VQ layers: the first layer encodes semantic information, the second encodes coarse-grained acoustic information, and the third encodes residual acoustic information. Unlike previous works [9, 48], which encode audio data into the same granularity in each layer, we propose a multi-scale approach that encodes audio data at different granularities across layers. We are motivated by the observation that semantic-level information can be preserved with fewer tokens, while acoustic-level information requires more tokens This multi-scale approach not only shortens the token sequence length but also offers flexibility for various tasks; for instance, audio understanding tasks may only require the semantic-level VQ layer. Additionally, we design a novel semantic loss and consistency loss to enhance the training of the LLM-Codec model.
We conduct experiments to validate the effectiveness of LLM-Codec in an in-context learning setting. Using the pre-trained LLAMA 2 7B model without parameter updates, we evaluate LLM-Codec on various audio understanding and generation tasks, such as speech emotion classification, audio classification, simple text-to-speech, and speech denoising. The main contributions of this work are summarized as follows:

- We propose a novel LLMs-driven audio codec model, LLM-Codec, which effectively bridges the text and audio modalities. To the best of our knowledge, this is the first work to quantize the audio data into the representation space of LLMs.

- We demonstrate the feasibility and potential of using the in-context learning ability of LLMs to solve unseen audio tasks, including audio understanding and generation tasks. Extensive experiments and ablation studies further validate the effectiveness of our method.

2 Related works

Audio Codec Models Historical investigations into low-bitrate parametric audio codecs began with earlier studies [21, 4]. However, the quality of these codecs typically faced limitations. Recently, advancements in neural network-based audio codecs have led to several promising developments [48, 9, 45, 24, 20]. These systems typically involve an encoder that extracts deep features from a latent space, which are then quantized and transmitted to a decoder for reconstruction. Particularly relevant to our work are the FACodec [20] and SpeechTokenizer [50] models, which explicitly model different properties of audio in different vector quantization layers. In contrast, our proposed LLM-Codec encodes audio data into a lexical word sequence, differing from these approaches.

Multimodal Large Language Models Recently, there has been tremendous progress in the area of multimodal LLMs. These models use pre-trained LLMs as the base and incorporate additional input modalities, such as vision [51, 27, 47, 52, 26, 36] and audio [7, 23, 15, 49, 34, 19, 39]. In general, multimodal LLMs consist of a pre-trained LLM, a pre-trained vision/audio encoder, and a modality adaptor. These systems often require the construction of extensive multimodal datasets to fine-tune the models. In the audio modality, most previous works focus on solving speech understanding [15] or general audio understanding [34, 19], but these models are not applicable to audio generation tasks. SpeechGPT addresses some audio understanding and generation tasks by fine-tuning all parameters and expanding the speech token vocabulary based on LLAMA. However, the speech tokens in SpeechGPT contain only semantic-level information, limiting its application to broader audio tasks, such as text-to-audio generation. Additionally, SpeechGPT does not explore the in-context learning ability to handle unseen tasks.

In-context Learning In-context learning, a form of few-shot learning, allows a large language model (LLM) to quickly adapt to a specific task during inference by reviewing just a few examples provided in the prompt [6]. This approach has proven successful in both natural language [44] and visual-language tasks [3, 47, 52, 27]. In the audio domain, several advanced methods have been proposed to leverage in-context learning for solving unseen audio tasks. SALM [7] introduces speech-augmented language models using in-context learning to address speech recognition and translation tasks, demonstrating that the SALM model can also handle keyword boosting. ICL-GSLM [13] employs warmup training and prompt tuning strategies to enhance the in-context learning abilities of pre-trained speech language models [25] for unseen tasks. However, ICL-GSLM primarily focuses on audio understanding tasks and overlooks audio generation tasks. Dynamic-superb [16] employs instruction-tuning for audio understanding tasks. Similarly, [41] and [40] investigate in-context learning within the speech understanding domain. Building on the success of in-context learning in NLP [44] and vision-language tasks [47, 52, 27], our study focuses on harnessing the in-context capabilities of frozen LLMs to address a broad range of audio understanding and generation tasks.

3 LLM-Codec

3.1 Overview

Previous audio codec models [9, 48, 45] use a VQ-VAE [37] framework, where the audio signal is first encoded into a discrete latent space and then decoded back into audio. Because audio codec models map the audio signal into discrete token sequences, many studies [17, 46, 5] have proposed training auto-regressive (AR) language models to generate these sequences, inspired by the success of LLMs in natural language processing. However, there is modal heterogeneity between the discrete audio tokens produced by the codec models and the text tokens used in LLMs. For example, the codebooks in audio codecs and the vocabularies of LLMs are typically unconnected, making it difficult to extend well-trained LLMs to audio modalities. Although previous works [49, 33] have demonstrated the effectiveness of expanding the vocabulary of LLMs to audio tokens and updating all of the parameters of LLMs, it will cost a lot of computing resources and forget the knowledge of the text. In this part, we present a large language models-driven audio codec model (LLM-Codec), which effectively bridges the gap between audio and text modalities. LLM-Codec is built on the VQ-VAE framework, but it differs from previous work in several key ways: (1) LLM-Codec is forced to quantize the audio signal into the token space of LLMs; (2) LLM-Codec adopts a multi-scale residual vector quantization strategy to balance the completeness and compactness of codec model; (3) LLM-Codec explicitly encodes different level information in different VQ layers. The following sections provide detailed insights into LLM-Codec, with Figure 2 offering a visual depiction.

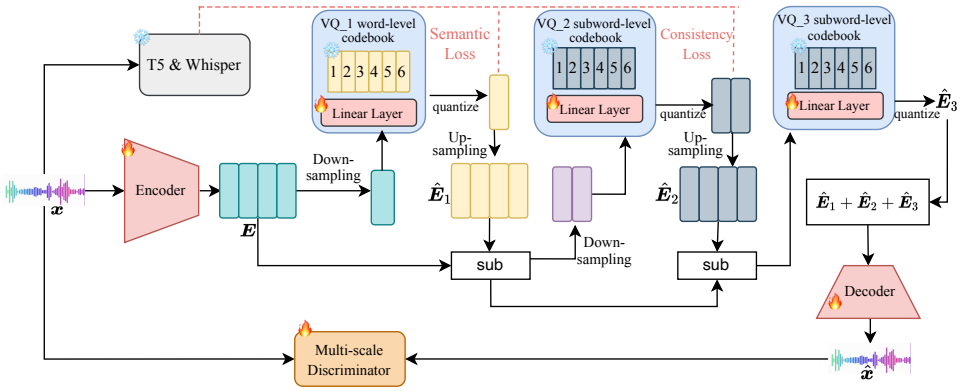


Figure 2: This figure provides a high-level overview of LLM-Codec, including an encoder, a decoder, a multi-scale discriminator, and a multi-scale residual VQ layers. Here, 'sub' denotes feature subtraction. Note that the modules marked with a snowflake are frozen during training.

3.2 Encoder and Decoder

For any audio $\boldsymbol{x}$, the encoder first encodes it into latent presentations $\boldsymbol{E}^{T,d}$, where T denotes the number of frames, d denotes the dimension of each vector. We set 4 down-sampling layers with $S = [3, 4, 5, 8]$ in the encoder, which results in 480 times down-sampling for audio. Then, each frame $\boldsymbol{e} \in \boldsymbol{E}$ is passed through the quantizer, which assigns it to the closest entry in a codebook, resulting in the quantized embedding $\hat{\boldsymbol{e}}$. Finally, the quantized feature $\hat{\boldsymbol{E}}$ inputs into the decoder to reconstruct $\hat{\boldsymbol{x}}$. Refer to Appendix B to find more model structure details.

3.3 Multi-scale residual vector quantization with the vocabulary of frozen LLM

We use three residual VQ layers to maintain the balance between completeness and compactness. Furthermore, we propose to set different quantization granularity in different VQ layers: we expect the first VQ layer can encode the semantic information, and such information can be saved with fewer tokens, thus an interpolation function is used to down-sample the encoder features $\boldsymbol{E}^{T,d}$ into $\boldsymbol{E}_1^{T/k_1,d}$, then $\boldsymbol{E}_1^{T/k_1,d}$ is passed through the first VQ layer to obtain $\hat{\boldsymbol{E}}_1^{T/k_1,d}$. For the second VQ layer, we expect it can encode coarse-grained acoustic information, thus we pass the residual of the first VQ layer into the next VQ layer. Before that, we first up-sampling $\hat{\boldsymbol{E}}_1^{T/k_1,d}$ into $\hat{\boldsymbol{E}}_1^{T,d}$, then obtain the residual features by

$$\boldsymbol{E}_2^{T,d} = \boldsymbol{E}^{T,d} - \hat{\boldsymbol{E}}_1^{T,d}. \tag{1}$$

Similarly, we also apply a down-sampling operation to $\boldsymbol{E}_2^{T,d}$, we set the down-sampling step as k_2. The features become as $\boldsymbol{E}_2^{T/k_2,d}$. Then we pass it into the second VQ layer and obtain $\hat{\boldsymbol{E}}_2^{T/k_2,d}$. Lastly, we expect the last VQ layer can preserve all of the residual acoustic information. We first obtain the residual features based on the quantized features of the first two VQ layers

$$\boldsymbol{E}_3^{T,d} = \boldsymbol{E}^{T,d} - \hat{\boldsymbol{E}}_1^{T,d} - \hat{\boldsymbol{E}}_2^{T,d}. \tag{2}$$

Considering the residual acoustic information is more complex and diverse, we directly apply the VQ operation to each frame without any down-sampling. By using a large down-sampling step in the encoder of codec, and applying a multi-scale VQ strategy, we can effectively reduce the number of quantized audio token sequences. In our setting, 1-second audio with a 16k sampling rate will be quantized into 57 tokens. To ensure that the first VQ layers encode semantic information, we propose incorporating a semantic loss during the training process. Furthermore, to maintain the training stability, we propose a consistency loss. The details will be introduced in Section 3.4.

The initialization of VQ layers To generate lexical tokens, we utilize a pre-trained LLAMA 2 codebook to initialize the VQ layers. Considering that the first layer, the VQ layer, is designed to encode the semantic information, we do not directly use the full LLAMA codebook. Instead, we

Table 1: Performance comparison between open-sourced audio codec models, baselines, and the proposed LLM-Codec. * means the reproduced results by ourselves.

Model	Down-sampling steps	Tokens per second	PESQ	STOI
Encodec_24k (3 Vanilla RVQ) [9]	320	225	2.18	0.79
DAC_16k (3 Vanilla RVQ) [24]	320	150	1.76	0.78
Baseline* (3 Vanilla RVQ)	480	99	2.64	0.83
Baseline* (2 Multi-scale RVQ)	480	41	2.22	0.79
Baseline* (1 Vanilla VQ)	480	33	2.01	0.76
LLM-Codec (Ours)	480	57	2.55	0.82

define a new codebook based on Oxford 5000 Words, these words are commonly used to make up any meaningful sentence. We choose these words that only consist of one or two sub-words in the LLAMA codebook. If a word includes two sub-words, we use the mean representation of two sub-words in the LLAMA codebook as the final representation. Lastly, the codebook size of the first VQ layer is 3248. We directly use the LLAMA codebook to initialize the second and third VQ layers. The codebook size is 32000. Furthermore, the LLAMA codebook embedding dimension is 4096, which is too large for codec training. Thus, we apply a linear mapping to 512. In the training process, the parameters of codebooks are fixed.

3.4 Training loss

Our approach is based on a GAN objective, in which we optimize both the generator(it consists of encoder, quantizer, and decoder) and the discriminators. For the generator, its training loss consists of three parts: (1) reconstruction loss term; (2) adversarial loss term (via discriminators); and (3) semantic and consistency losses. In the following, we give the details of proposed semantic loss and consistency loss. Refer to Appendix B.2 to find the details of reconstruction loss and adversarial loss.

Semantic loss To enhance the semantic representation ability in the first layer, we introduce a semantic loss for the first VQ layer. We expect it can encode semantic information, for example, if the input audio includes a sound event, the first layer should encode which semantic information of the sound event. Similarly, if the input audio is speech, the first layer should encode the content of the speech. To realize this target, we use a pre-trained T5-base model [31] to extract a global representation vector $\boldsymbol{g}$ for the input audio content. We use Whisper to obtain its transcriptions if the input audio is speech. If the input audio is sound, we use its audio caption label:

$$\mathcal{L}_s = L_1(mean(\hat{\boldsymbol{E}}_1^{T,d}), \boldsymbol{g}) \tag{3}$$

Consistency loss In our early experiments, we found the training of LLM-Codec is not stable, and the model is easy to collapse. One of the reasons is that we designed a significant down-sampling rate and the codebooks are fixed in the training, which increases the training difficulty. To solve this issue, we propose a consistency loss to maintain the training stability. Specifically, we propose using a pre-trained Whisper encoder [30] to extract frame-level features $\boldsymbol{w}$, then using these features as prior knowledge to guide the second VQ layer.

$$\mathcal{L}_c = L_1(\hat{\boldsymbol{E}}_2^{T/2,d}, inp(\boldsymbol{w})) \tag{4}$$

where inp denotes the interpolation function to align the feature dimension between the quantized features and whisper features. We chose the Whisper encoder because it is trained not only on speech data but also on non-speech data. Furthermore, we do not apply this loss on the third VQ layer, because we expect the third VQ layer to encode the residual information.

4 Experimental Results

4.1 Experimental Settings

4.1.1 LLM-Codec training and reconstruction performance evaluation

Training data LLM-Codec is a universal audio codec model, trained on both speech and sound datasets. For speech data, we use a portion of the MLS dataset [29], and for sound data, we use the

Table 2: Audio understanding task evaluation results. Task induction denotes the explanatory text that precedes the sequence of audio and text. It is intended to describe the task to the model in natural language, for example: Please answer the question. Accuracy (%) is used as the metric. For the Random guess, we calculate the average based 5 times evaluation. K shots refers to the number of distinct samples for each category, and Repeats refer to how many times we copy the prompt samples.

Method	# Layers	Task Induction	✗	✓	✓	✓	✓
		K Shots	1	1	3	1	1
		Repeats	0	0	0	2	3
2-way speech emotion classification							
Random	None				44		
BLSP [39]	Whisper encoder		9	29	50	33	19
LLM-Codec	semantic layer		25	**53**	**59**	53	54
LLM-Codec	semantic + acoustic layers		**45**	49	53	**55**	**54**
2-way sound event classification.							
Random	None				45		
BLSP [39]	Whisper encoder		44	47	54	15	17
LLM-Codec	semantic layer		**48**	**60**	**57**	**57**	**73**
LLM-Codec	semantic+acoustic layers		41	48	55	54	62
3-way sound event classification.							
Random	None				30		
BLSP [39]	Whisper encoder		23	26	36	24	16
LLM-Codec	semantic layer		**38**	**41**	**39**	43	42
LLM-Codec	semantic+acoustic layers		25	37	35	**44**	**50**

AudioCaps dataset [22]. In total, we utilized 2k hours of audio data to train the LLM-Codec model.

Model setting The details of the LLM-Codec model configuration can be found in Appendix B. For the proposed multi-scale RVQ, we set three scales, with down-sampling rates of $k_1 = 4$, $k_2 = 2$, and $k_3 = 1$ for each layer. We initialize the VQ layers using the vocabulary of the LLAMA 2 7B model. The VQ layers are fixed during the training stage.

Evaluation metrics To verify the reconstruction performance, we use Perceptual Evaluation of Speech Quality (PESQ), Short-Time Objective Intelligibility (STOI), and Mel reconstruction loss.

Evaluation data We conduct evaluation on speech and sound datasets, for speech data, we choose 200 utterances from VCTK [38], for sound data, we choose 200 utterances from ESC 50 dataset.

Baselines We compare our model with publicly available audio codec models, including Encodec [9] and DAC [24]. Additionally, we compare it with our reproduced audio codec models, which were trained on the same dataset as LLM-Codec.

4.1.2 LLMs equipped with the proposed LLM-Codec for downstream audio tasks

Evaluation dataset We choose commonly used test datasets for each task and construct N-way-K-shot test pairs. More details about constructing evaluation samples can be found in Appendix C.1.

Baselines Given the limited number of works focusing on few-shot learning for unseen audio tasks, we choose BLSP [39] as one of the baselines for audio understanding tasks. Since BLSP is fine-tuned on a continuation writing task and does not explicitly address audio classification, these tasks are considered unseen for the BLSP model. Additionally, we compare our approach with instruction-tuning-based models as described in dynamic-superb [16]. For audio generation tasks, we did not find directly related works, so we report the performance of state-of-the-art specialized models.

4.2 Main results

We first present the reconstruction performance comparison. Then we apply the LLM-Codec and LLAMA 2 7B model for audio understanding and audio generation tasks, to verify the ability of the proposed method. Lastly, we give the visualization of LLM-Codec to explain why it can work. We leave more experiments on Appendix D.

Reconstruction performance We compare the audio reconstruction quality with previous works

Table 3: Evaluation on dynamic-superb benchmark tasks. Accuracy (%) is used as the metric.

Task	ImageBind-LLM [16]	Whisper-LLM [16]	ASR-ChatGPT [16]	Ours
Accent Classification	19	4	7	**24**
Bird Sound Detection	28	14	15	**50**
Chord Classification	44	**58**	3	55
Language Identification	26	13	**96**	25

Encodec [9], DAC-Codec [24], and our baseline model. We report Perceptual Evaluation of Speech Quality (PESQ) and Short-Time Objective Intelligibility (STOI). Table 1 shows the results. Compared to previous methods, the LLM-Codec achieves better reconstruction performance while utilizing fewer tokens. More specifically, the LLM-Codec model can compress 1-second audio data into a sequence that only includes 57 tokens, which significantly reduces the sequence length. Compared to the RVQ baseline model, the LLM-Codec significantly reduces the compressed tokens, and its reconstruction performance does not significantly decline. In Section 4.3, we will show the importance of compressing audio into fewer tokens. We also conduct experiments to validate whether we can use a few VQ layers, such as 1 VQ layer or 2 VQ layer, we can see that the reconstruction performance will significantly drop. To maintain the balance between completeness and compactness, we choose a multi-scale 3 VQ layer as the default setting.

Speech Emotion Classification The speech emotion classification task [10] aims to predict the emotion label of the speech. We conduct 2-way K-shot experiments on the ESD [1] dataset. Experimental results are shown in Table 2. We have the following findings: (1) Task induction is important to maintain the stability of performance, we can see that without task induction, the classification accuracy will dramatically decline. (2) The semantic layer effectively extracts the global semantics of audio, which can be easily understood by the LLAMA model. (3) Using more demonstration samples (*e.g.* 3 shots), the performance will be better. (4) Repeating the demonstration samples can also bring improvement. (5) Compared to the BLSP, our method performs better in any setting. Furthermore, we also note that the performance of BLSP will drop when repeat operation is used. One possible reason is that BLSP only learns the translation relationship between text and speech, repeating samples cannot bring new cues for LLMs to solve the new task. Instead, our LLM-Codec learns to map the audio data into the latent space of LLMs, increasing the number of demonstration samples can help LLMs to find special patterns to solve this new task.

Table 4: Text-to-speech generation performance.

Model	ACC	DNSMOS
GT	-	2.91
FastSpeech 2	-	3.42
LLM-Codec (Ours)	70	2.92

Sound Event Classification Sound event classification aims to recognize the sound event in the audio. In general, an audio may include multiple events. To simplify the recognition difficulty, we assume each audio only includes one event. We conduct experiments on the ESC50 dataset [28], which includes 50 different types of events. We construct 2-way-K-shot and 3-way-K-shot evaluations based on the ESC50 test set. Compared with the BLSP model, our proposed method gets better performance. Based on the experimental results from two audio understanding tasks, we can see that the semantic VQ layer is very important for understanding tasks.

Dynamic-SUPERB Benchmark We also conduct experiments on Dynamic-SUPERB Benchmark tasks [16]. In [16], authors propose an instruction-tuning strategy for multi-modal LLMs. They first construct a lot of audio tasks as training data, then validate some unseen audio tasks in a zero-shot way. To make a fair comparison, we use the same test set with them, and choose the first N samples as the demonstration to construct a N-way-1-shot evaluation. As Table 3 shows 4 selected audio understanding tasks, our proposed method obtains better or compared performance than these baselines in [16]. Especially, for the bird sound detection task, our proposed method obtained great improvement over previous methods. We also note that our method performs worse on language identification, the possible reason is that our codec model is only trained on English speech data. In the following, we will show that LLM-Codec also can be used to conduct audio generation tasks.

Simple text-to-speech generation We conduct text-to-speech generation on the Free Spoken Digit Dataset (FSDD) dataset [11], which includes 3 speakers and 1,500 recordings. Unlike the traditional TTS model, which generates any speech content, this task generates digit speech. Our current model to generate complex speech content is still challenging. We use accuracy (ACC) to assess the

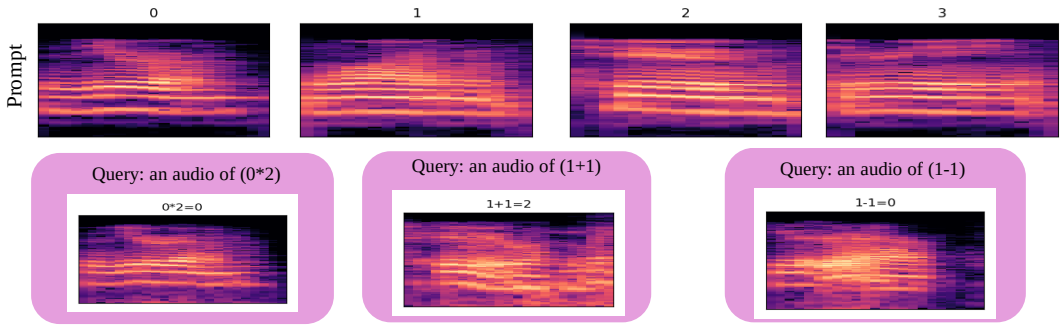


Figure 3: Examples of simple text-to-speech generation using LLM-Codec and LLAMA2 model.

content of the generated sample whether following the instructions. DNSMOS is used to assess the speech quality of generated samples. We construct 20 different query questions, including addition, subtraction, multiplication, division, and reasoning (finding more details from Appendix C.2). From Table 4, we can see that our proposed model can accurately understand the query in most cases (the accuracy is 70 %) and generate good-quality speech samples. Figure 3 gives a visualization of generating speech based on the query. The frozen LLAMA model learns about 4 digits (0-3), each audio digit includes 5 samples. We add the context for each audio: "an audio of k" before inputting the audio's discrete representations into LLAMA, as Figure 1 (b) shows. After that, we let the LLAMA 2 model generate corresponding speech digits based on the instruction. We also note that the generated audio appears different from all context audio samples, demonstrating the cross-modal reasoning capability of LLMs when using the LLM-Codec as the connector for text and audio.

Simple Speech Denoising To verify whether the proposed method can conduct speech-denoising tasks, we simulate noisy speech based on the VCTK dataset and NoiseX-92 dataset, we set the SNR ranges from -20 to 20. For each clean speech, we choose 5 different noises to simulate noisy speech. The first 4 noisy and clean audio pairs are used as demonstrations, and the model learns to denoise the last noisy one. To improve in-context learning ability, we repeat the demonstration samples 4 times. The experimental results as Table 5 shows, we can see that the proposed method can also learn to denoise without any training. Furthermore, we also note that the performance has a large room to improve compared to special models.

Table 5: Speech denosing evaluation.

Model	PESQ	STOI
SGMSE+ [32]	3.53	0.79
LLM-Codec (Ours)	2.17	0.57

Token Visualization We visualize the tokens produced by the first VQ layer of LLM-Codec for different types of sound in Figure 4. We have the following findings: (1) Although the two audios include the same sound event, the quantized sequence is not exactly the same. (2) The quantized sequence of two same types of audio has a similar pattern, *e.g.* their token sequences have similar repeating patterns or the same word. Such patterns may help the LLMs recognize the type of audio.

4.3 Ablation study

The influence of multi-scale RVQ We first conduct experiments to see the effectiveness of multi-scale RVQ. As Table 6 shows, compared with vanilla RVQ, the proposed multi-scale RVQ does not bring a significant reconstruction performance drop, which validates our assumption that semantic information does not need too much token to encode. Secondly, we find the multi-scale RVQ significantly reduces the length of the token sequence and brings benefits for downstream tasks (the audio classification accuracy is better than the baseline). One potential explanation is that LLMs can better identify the unique pattern in a brief sequence. Intuitively, the semantic information included in a 1-second audio is limited. It is unnecessary to use very long sequences to represent limited information.

The influence of down-sampling times We can see that using a smaller down-sampling rate (320) can improve the reconstruction performance, but it also increases the length of the token sequence. We can see that the classification accuracy will decrease when the sequence length increases.

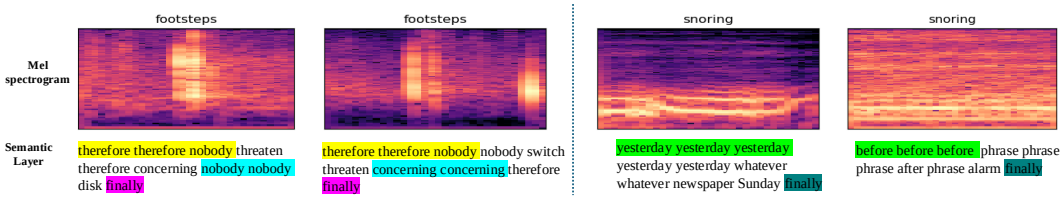


Figure 4: The token visualization of the semantic layer of LLM-Codec is shown. We present two groups of samples, each containing two audio recordings with the same sound event label. In each group, we use the same color to highlight potentially similar patterns in the two audio recordings, such as identical token sub-sequences or token repeating frequencies. We speculate that these patterns can be easily recognized by LLMs, allowing them to learn new sound events quickly with just a few demonstrations.

The influence of semantic loss Without semantic loss, the performance of the audio understanding task will drop. Furthermore, we also find that adding semantic loss does not influence the reconstruction performance. In summary, the proposed semantic loss is very useful.

The influence of consistency loss We find that consistency loss is important to maintain training stability. Without it, we can see the model fails to reconstruct the audio. We conjecture that frozen codebooks and large compression rates significantly improve the difficulty of training. The consistency loss forces the second VQ layer to produce features similar to those of the Whisper encoder, which provides guidance for vector quantization and prevents the model from collapsing in the early stage.

The influence of word-level codebooks We also conduct experiments to show the effectiveness of using word-level codebooks to initialize the first VQ layer. Compared with using sub-word vocabulary for the first VQ layer, we can see that using the proposed word-level codebook can improve the reconstruction performance and classification accuracy.

The importance of frozen codebooks LLM-Codec compresses the audio data into the token space of LLMs by initializing the codebooks with the LLMs' vocabulary and fixing it during the training stage. Table 6 also presents the results of updating codebooks: it can improve the reconstruction performance, but the accuracy is a significant drop. The result is consistent with our hypothesis: updating the codebooks parameter will decrease the codec training difficulty, but the learned codebook space is different from the LLM's token space, resulting in the downstream task performance declines.

Different setting of k_1 and k_2 in multi-scale RVQ We validate a new setting for multi-scale RVQ with $k_1 = 3$ and $k_2 = 5$. We can see that the reconstruction performance will decline. We think one of the reasons is that the second VQ layer should not apply a large down-sampling step, which significantly influences the reconstruction.

Codebook usage Previous works [18, 48, 24] suggest that using a large-scale codebook may result in codebook collapse ((where a fraction of the codes are unused). We calculate the codebook usage for each VQ layer in LLM-Codec. The used codes are 3246 (3248), 31911 (32000), 31941 (32000) for each VQ layer, which shows that most of codes are used.

Table 6: Ablation studies on training loss, multi-scale RVQ setting, initialization of VQ layer. The classification accuracy (%) is evaluated under the sound event classification task 2-way 1-shot setup.

Model	Down-sampling	Tokens per second	PESQ	STOI	ACC
Baseline (3 Vanilla RVQ)	480	99	2.64	0.83	55
LLM-Codec (3 Multi-scale RVQ)	480	57	2.55	0.82	60
LLM-Codec (3 Multi-scale RVQ)	320	87	2.60	0.83	57
w/o semantic loss	480	57	2.54	0.82	58
w/o consistency loss	480	57	1.19	0.53	48
w/o word-level codebook	480	57	2.46	0.81	59
updating codebooks	480	57	2.63	0.83	55
seting $k_1 = 3$ and $k_2 = 5$	480	50	2.35	0.79	58

5 Conclusion

In this study, we explore a cross-modal in-context learning approach to solve unseen audio tasks in a few-shot style. Specifically, we propose to train a LLMs-driven audio codec (LLM-Codec) that compresses the audio signal into the token space of LLMs. The LLM-Codec effectively reduces the modal heterogeneity between text and audio. With the help of LLM-Codec, pre-trained LLMs can be applied to audio understanding and generation tasks. We demonstrate that LLM-Codec has good reconstruction performance, and the compressed token sequence is suitable for LLMs to understand and generate. Experiments show that the LLMs equipped with the proposed LLM-Codec, prompted by only a few examples, are capable of achieving the expected functions in many scenarios. The discussions on limitations are included in the Appendix.

References

[1] Emotional voice conversion: Theory, databases and esd. *Speech Communication*, 137:1–18, 2022.

[2] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[3] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. *Advances in neural information processing systems*, 35:23716–23736, 2022.

[4] Bishnu S Atal and Suzanne L Hanauer. Speech analysis and synthesis by linear prediction of the speech wave. *The journal of the acoustical society of America*, 50(2B):637–655, 1971.

[5] Zalán Borsos, Raphaël Marinier, Damien Vincent, Eugene Kharitonov, Olivier Pietquin, Matt Sharifi, Dominik Roblek, Olivier Teboul, David Grangier, Marco Tagliasacchi, et al. Audiolm: a language modeling approach to audio generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2023.

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[7] Zhehuai Chen, He Huang, Andrei Andrusenko, Oleksii Hrinchuk, Krishna C Puvvada, Jason Li, Subhankar Ghosh, Jagadeesh Balam, and Boris Ginsburg. Salm: Speech-augmented language model with in-context learning for speech recognition and translation. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 13521–13525. IEEE, 2024.

[8] Yunfei Chu, Jin Xu, Xiaohuan Zhou, Qian Yang, Shiliang Zhang, Zhijie Yan, Chang Zhou, and Jingren Zhou. Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models. *arXiv preprint arXiv:2311.07919*, 2023.

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

[10] Moataz El Ayadi, Mohamed S Kamel, and Fakhri Karray. Survey on speech emotion recognition: Features, classification schemes, and databases. *Pattern recognition*, 44(3):572–587, 2011.

[11] Zohar JacksonCésar. et.all. free-spoken-digit-dataset: v1.0.8 (v1.0.8). zenodo. *https://doi.org/10.5281/zenodo.1342401*, 2018.

[12] Haohan Guo, Fenglong Xie, Xixin Wu, Frank K. Soong, and Helen Meng. Msmc-tts: Multi-stage multi-codebook vq-vae based neural tts. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1811–1824, 2023.

[13] Ming-Hao Hsu, Kai-Wei Chang, Shang-Wen Li, and Hung-yi Lee. An exploration of in-context learning for speech language model. *arXiv preprint arXiv:2310.12477*, 2023.

[14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[15] Shujie Hu, Long Zhou, Shujie Liu, Sanyuan Chen, Hongkun Hao, Jing Pan, Xunying Liu, Jinyu Li, Sunit Sivasankaran, Linquan Liu, et al. Wavllm: Towards robust and adaptive speech large language model. *arXiv preprint arXiv:2404.00656*, 2024.

[16] Chien-yu Huang, Ke-Han Lu, Shih-Heng Wang, Chi-Yuan Hsiao, Chun-Yi Kuan, Haibin Wu, Siddhant Arora, Kai-Wei Chang, Jiatong Shi, Yifan Peng, et al. Dynamic-superb: Towards a dynamic, collaborative, and comprehensive instruction-tuning benchmark for speech. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12136–12140. IEEE, 2024.

[17] Rongjie Huang, Chunlei Zhang, Yongqi Wang, Dongchao Yang, Luping Liu, Zhenhui Ye, Ziyue Jiang, Chao Weng, Zhou Zhao, and Dong Yu. Make-a-voice: Unified voice synthesis with discrete representation. *arXiv preprint arXiv:2305.19269*, 2023.

[18] Minyoung Huh, Brian Cheung, Pulkit Agrawal, and Phillip Isola. Straightening out the straight-through estimator: Overcoming optimization challenges in vector quantized networks. In *International Conference on Machine Learning*. PMLR, 2023.

[19] Atin Sakkeer Hussain, Shansong Liu, Chenshuo Sun, and Ying Shan. Mugen: Multi-modal music understanding and generation with the power of large language models. *arXiv preprint arXiv:2311.11255*, 2023.

[20] Zeqian Ju, Yuancheng Wang, Kai Shen, Xu Tan, Detai Xin, Dongchao Yang, Yanqing Liu, Yichong Leng, Kaitao Song, Siliang Tang, et al. Naturalspeech 3: Zero-shot speech synthesis with factorized codec and diffusion models. *arXiv preprint arXiv:2403.03100*, 2024.

[21] 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*, volume 7, pages 597–600. IEEE, 1982.

[22] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 119–132, 2019.

[23] Zhifeng Kong, Arushi Goel, Rohan Badlani, Wei Ping, Rafael Valle, and Bryan Catanzaro. Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities. *arXiv preprint arXiv:2402.01831*, 2024.

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

[25] Kushal Lakhotia, Eugene Kharitonov, Wei-Ning Hsu, Yossi Adi, Adam Polyak, Benjamin Bolte, Tu-Anh Nguyen, Jade Copet, Alexei Baevski, Abdelrahman Mohamed, et al. On generative spoken language modeling from raw audio. *Transactions of the Association for Computational Linguistics*, 9:1336–1354, 2021.

[26] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[27] Hao Liu, Wilson Yan, and Pieter Abbeel. Language quantized autoencoders: Towards unsupervised text-image alignment. *Advances in Neural Information Processing Systems*, 36, 2024.

[28] Karol J. Piczak. ESC: Dataset for Environmental Sound Classification. In *Proceedings of the 23rd Annual ACM Conference on Multimedia*, pages 1015–1018. ACM Press.

[29] 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.

[30] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518. PMLR, 2023.

[31] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[32] Julius Richter, Simon Welker, Jean-Marie Lemercier, Bunlong Lay, and Timo Gerkmann. Speech enhancement and dereverberation with diffusion-based generative models. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:2351–2364, 2023.

[33] Paul K Rubenstein, Chulayuth Asawaroengchai, Duc Dung Nguyen, Ankur Bapna, Zalán Borsos, Félix de Chaumont Quitry, Peter Chen, Dalia El Badawy, Wei Han, Eugene Kharitonov, et al. Audiopalm: A large language model that can speak and listen. *arXiv preprint arXiv:2306.12925*, 2023.

[34] Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang. Salmonn: Towards generic hearing abilities for large language models. *arXiv preprint arXiv:2310.13289*, 2023.

[35] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[36] Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

[37] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[38] Christophe Veaux, Junichi Yamagishi, and Kirsten MacDonald. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. 2017.

[39] Chen Wang, Minpeng Liao, Zhongqiang Huang, Jinliang Lu, Junhong Wu, Yuchen Liu, Chengqing Zong, and Jiajun Zhang. Blsp: Bootstrapping language-speech pre-training via behavior alignment of continuation writing. *arXiv preprint arXiv:2309.00916*, 2023.

[40] Siyin Wang, Chao-Han Yang, Ji Wu, and Chao Zhang. Can whisper perform speech-based in-context learning? In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 13421–13425. IEEE, 2024.

[41] Siyin Wang, Chao-Han Huck Yang, Ji Wu, and Chao Zhang. Bayesian example selection improves in-context learning for speech, text, and visual modalities. *arXiv preprint arXiv:2404.14716*, 2024.

[42] Yuanyuan Wang, Hangting Chen, Dongchao Yang, Jianwei Yu, Chao Weng, Zhiyong Wu, and Helen Meng. Consistent and relevant: Rethink the query embedding in general sound separation. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 961–965. IEEE, 2024.

[43] Pete Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *arXiv preprint arXiv:1804.03209*, 2018.

[44] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[45] Dongchao Yang, Songxiang Liu, Rongjie Huang, Jinchuan Tian, Chao Weng, and Yuexian Zou. Hifi-codec: Group-residual vector quantization for high fidelity audio codec. *arXiv preprint arXiv:2305.02765*, 2023.

[46] Dongchao Yang, Jinchuan Tian, Xu Tan, Rongjie Huang, Songxiang Liu, Xuankai Chang, Jiatong Shi, Sheng Zhao, Jiang Bian, Xixin Wu, et al. Uniaudio: An audio foundation model toward universal audio generation. *arXiv preprint arXiv:2310.00704*, 2023.

[47] Lijun Yu, Yong Cheng, Zhiruo Wang, Vivek Kumar, Wolfgang Macherey, Yanping Huang, David Ross, Irfan Essa, Yonatan Bisk, Ming-Hsuan Yang, et al. Spae: Semantic pyramid autoencoder for multimodal generation with frozen llms. *Advances in Neural Information Processing Systems*, 36, 2024.

[48] 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, 2021.

[49] Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. *arXiv preprint arXiv:2305.11000*, 2023.

[50] Xin Zhang, Dong Zhang, Shimin Li, Yaqian Zhou, and Xipeng Qiu. Speechtokenizer: Unified speech tokenizer for speech large language models. *arXiv preprint arXiv:2308.16692*, 2023.

[51] Kaizhi Zheng, Xuehai He, and Xin Eric Wang. Minigpt-5: Interleaved vision-and-language generation via generative vokens. *arXiv preprint arXiv:2310.02239*, 2023.

[52] Lei Zhu, Fangyun Wei, and Yanye Lu. Beyond text: Frozen large language models in visual signal comprehension. *arXiv preprint arXiv:2403.07874*, 2024.

Appendices

A Appendix Overview

These Appendices provide additional details to support our main manuscript, including (1) the training detail and model structure of LLM-Codec. (2) The details of the evaluation dataset. (3) More audio task evaluation results. (4) Limitations.

B More details of LLM-Codec

B.1 Model structure

Table 7 gives the details of LLM-Codec configuration, which results in 160M parameters. To facilitate research on cross-modal in-context learning and multi-modal LLMs, we have open-sourced the LLM-Codec models.

	LLM-Codec
Input shape	(1, 1, T)
Encoder (input dimension)	32
Down-sampling rate	[3, 4, 5, 8]
latent dimension	512
Codebook dimension	4096
Transformer layer dimension	512
Number of Transformer heads	8
Decoder dimension	1536
Up-sampling rate	[8, 5, 4, 3]
VQ strides	[5, 3, 1]

Table 7: LLM-Codec model backbone configurations

Encoder and Decoder Considering a single-channel audio signal $\boldsymbol{x} \in \mathcal{R}^{t \times sr}$, where t and sr denote the audio duration and the sample rate. The overall architecture is similar to previous audio codec models, such as Encodec [9], DAC [24], and HiFi-Codec [45], which includes four main parts: encoder, quantizer, decoder, and discriminators. Figure 2 provides a visual depiction of the proposed method. For any input $\boldsymbol{x}$, the encoder first encodes it into latent presentations $\boldsymbol{E}^{T,d}$, where T denotes the number of frames, d denotes the dimension of each vector. Due to the encoder includes some down-sampling layers, resulting in $T << t \times sr$. Then each frame $\boldsymbol{e} \in \boldsymbol{E}$ is passed through the quantizer, which assigns it to the closest entry in a codebook, resulting in the quantized embedding $\hat{e}$. Finally, the quantized feature $\hat{\boldsymbol{E}}$ inputs into the decoder to reconstruct $\hat{\boldsymbol{x}}$. The encoder and decoder architecture follows previous works Encodec [9] and DAC-Codec [24], which includes several convolution layers and transformer layers. Specifically, the encoder model comprises a 1D convolution with C channels and a kernel size of 7, leading into B convolution blocks. Each block contains a residual unit followed by a down-sampling layer, which employs a convolution with a kernel size K that is twice the stride S. The residual unit itself comprises two convolutions, each with a kernel size of 3, linked by a skip connection. The transformer block is used for sequence modeling, and concludes with a final 1D convolution layer featuring a kernel size of 7. In this study, we set $S = [3, 4, 5, 8]$, which results in 480 times down-sampling for audio. The decoder mirrors the encoder's architecture, substituting stride convolutions with transposed convolutions and reversing the stride order.

Discriminators For the discriminators, we follow previous work [46], which combines the mel-spectrogram and log-mel-spectrogram features and then input them into a network consisting of several convolutional layers. In our experiments, we use 6 different discriminators with different configurations. Specifically, we set the hidden dimension as {64, 128, 256, 512, 512, 512} and the hop length as {32, 64, 128, 256, 512, 1024}.

B.2 Reconstruction loss and adversarial loss for LLM-Codec

The reconstruction loss is calculated between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$. We design the loss from two aspects: the time domain and the frequency domain. For the time domain, we directly calculate the L_1 loss between $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$. For the frequency domain, we calculate the L_1 loss between the STFT spectrogram of $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$. Note that a sub-band split strategy [42] is used to split the spectrogram into several parts, and then we calculate the loss between these sub-bands. The adversarial loss is used to improve the perceptual quality of generated audio. A multi-scale Mel-spectrogram discriminators [46] is used. To train the discriminator, we can optimize the following objective function:

$$\mathcal{L}_d = \frac{1}{K}\sum_{i=1}^{K} max(0, 1 - D_k(\boldsymbol{x})) + max(0, 1 + D_k(\hat{\boldsymbol{x}})) \tag{5}$$

where K denotes the number of discriminators. In the training stage, the adversarial loss for the generator is calculated as a hinge loss over the logits of these discriminators:

$$\mathcal{L}_{adv} = \frac{1}{K}\sum_{i=1}^{K} max(0, 1 - D_k(\hat{\boldsymbol{x}})) \tag{6}$$

We also compute the feature loss by taking the average absolute difference between the discriminator's internal layer outputs for the generated audio and those for the corresponding real audio.

B.3 Training details

The AdamW optimizer is used in the training. We set the learn rate as $1e-4$. We train the model with 100k steps. For the training loss, we combine all of the loss terms without a special loss design. In the training stage, we use the pre-trained T5-base model and Whisper-base model for the reason that their latent dimension is both 512. We conduct all of the experiments with 2 NVIDIA A100-80G GPUs.

C Evaluation dataset

In this part, we show how to construct an evaluation dataset for the N-way-k-shot test.

C.1 N-way-k-shot test samples

Speech emotion classification with LLAMA 2. We give an example of 2-way 1-shot classification tasks. Firstly, we get the emotion class set from the ESD dataset: ['Angry', 'Happy', 'Neutral', 'Sad', 'Surprise']. Then we randomly choose two emotions as targets, and get the corresponding audios. For example, assuming that we get *Happy* and *Sad* the prompt can be

```
For each of the following input-output pairs, the output is
one of ['Happy' or 'Sad']
###
Input: <token sequence from a happy emotion of audio>
Output: happy
###
Input: <token sequence from a sad emotion of audio>
Output: sad
###
Input: <token sequence from the query audio>
Output:
```

We use greedy decoding to get a maximum of 16 tokens from LLAMA 2 7B.

Sound event classification with LLAMA 2. We give an example of 3-way 1-shot classification tasks. Firstly, we get the sound event class set from the ESC50 dataset. Then we randomly choose three sound events as targets, and get the corresponding audio. For example, assuming that we get *dog*, *speaking*, and *mouse click* we set the prompt as

```
For each of the following input output pairs,
output is one of [‘dog’ or ‘speaking’ or ’mouse_click’]
###
Input: <token sequence from a dog event of audio>
Output: dog
###
Input: <token sequence from a speaking event of audio>
Output: speaking
###
Input: <token sequence from a mouse click event of audio>
Output: mouse click
###
Input: <token sequence from the query audio>
Output:
```

C.2 Audio generation

Text-to-speech generation with LLAMA 2

```
Instruction: Learn a foreign language for different digits,
then generate the corresponding number using
foreign language based on instruction
###
Input: <an audio of 1>
Output: <token sequence of audio 1>
###
Input: <an audio of 2>
Output: <token sequence of audio 2>
###
Input: <an audio of 3>
Output: <token sequence of audio 3>
###
Input: <an audio of 1+1>
Output:
```

To simplify to generation process, we set each audio has the same duration.

Text-to-speech question design we designed 20 different questions for text-to-speech, which include addition, subtraction, multiplication, division, and reasoning.

```
###
Input: <an audio of (1+1)>
###
Input: <an audio of (1+2)>
###
Input: <an audio of (2+2)>
###
Input: <an audio of (5-1)>
###
Input: <an audio of (5-2)>
###
Input: <an audio of (1-1)>
###
Input: <an audio of (0*2)>
###
Input: <an audio of (2*2)>
###
Input: <an audio of (1/1)>
###
Input: <an audio of (2/1)>
```

```
###
Input: <an audio of (4/2)>
###
Input: <an audio of (the square root of 4)>
###
Input: <an audio of (the square root of 1)>
###
Input: <an audio of (the last digit of 110)>
###
Input: <an audio of (the first digit of 110)>
###
Input: <an audio of (the sum of 1+1+1)>
###
Input: <an audio of (the next digit of 4)>
###
Input: <an audio of (sequence 0,1,2,3 what is next?)>
###
Input: <an audio of (sequence 4,3,2,1 what is next?)>
###
Input: <an audio of (how many days in a week)>
```

D More audio tasks evaluation experiments with the proposed method

In the following, we show the results of speech command recognition and text-to-sound generation.

D.1 Speech Command Recognition

Table 8: Speech command recognition evaluation results on Speech Command dataset. Accuracy (%) is used as the metric. For the Random guess, we run 5 times then calculate the average.

Method	# Layers	Task Induction	✗	✓	✓	✓	✓
		K Shots	1	1	3	1	1
		Repeats	0	0	0	1	3
LLM-Codec	semantic layer		50	53	59	54	56
LLM-Codec	semantic+acoustic layers		25	53	58	49	52
BLSP [39]	Whisper encoder		29	65	84	69	59
Random	None				44		

Speech command recognition refers to the recognition and interpretation of short phrases or keywords that are typically used to control devices or applications. In this part, we choose audio samples from the Speech Command dataset [43]. We choose four types of commands, including down, go, left, and right. For each command, we randomly choose 20 utterances, then we use these data to construct a 2-way-K-shot evaluation. Experimental results are shown in Table 8, we can see that only using the semantic VQ layer brings the best performance. Instead, if the tokens from the acoustic layer are used, the performance will decline. One possible reason is that for the speech command recognition task, it only needs to understand the content, and the content information has been saved in the semantic layer, the additional acoustic information may disturb the LLMs's prediction.

D.2 Simple text-to-sound generation

Similarly, we can also use the same setting as text-to-speech to conduct text-to-sound generation tasks. We choose a test set from the ESC50 dataset [28], and let the model learn to generate sound events based on the text label. For example, we can set several different sound types in the prompt, and then ask the LLAMA model to generate a new audio. However, we also find that it is hard to ask LLM to generate new types of sound.

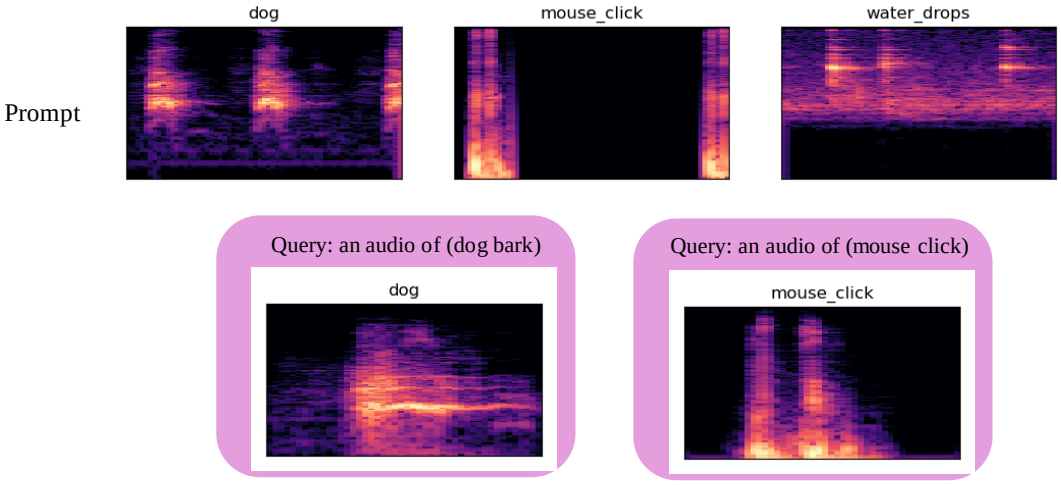


Figure 5: Examples of simple text-to-sound generation on FSDD dataset using LLM-Codec with a frozen LLAMA2 7B model.

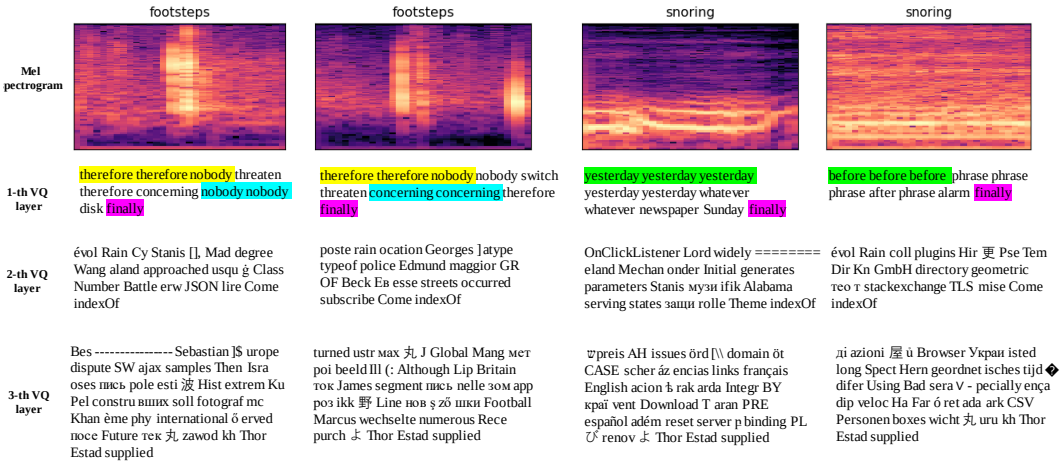


Figure 6: The token visualization of three VQ layers with LLM-Codec. The audio samples are from the ESC50 dataset.

D.3 Token visualization

Figure 6 shows the details of three VQ layers token visualization. We have the following findings: (1) The few tokens in the first layers seem to more easy to understand audio's pattern. For example, we can easily find two audios that have the same sound event can be quantized into a very similar sequence. Because we force the first VQ layer to encode the semantic level information. Instead, the second and third VQ layers aims to encode the acoustic information, but these audios have obvious difference in acoustic condition (we can observe it from its mel-spectrogram). Second, it is worth noting that all of the training data is English-related, but we can see that the encoded sequence also includes other language, such as Chinese.

E Limitations

Although we show the possibility of using the in-context learning ability of LLMs for unseen audio tasks without any parameter update, the performance of these tasks is still poorer than these special models in the audio domain. The capability to learn within an in-context framework is significantly

limited for a modality that was not exposed during the training process. Due to the LLM's context length limitation, we cannot add more demonstration samples to help improve the performance. We think it is worth exploring using more demonstrations to improve its in-context learning ability. Moreover, we only explore to use LLAMA 2 7B model as the backbone, more advanced open-sourced LLMs are worth exploring. Considering we have open-sourced the LLM-Codec, readers can conduct experiments on their favorite LLMs. In the future, we will explore to train multi-modal LLMs by fine-tuning LLMs on text-audio datasets with the help of LLM-Codec.

F Potential Negative Societal Impacts

This paper aims to build a multi-modal LLM for audio understanding and generation tasks in a few-shot style without any parameter update for LLMs, which will ease the effort to develop different special models for different tasks. A negative impact is the risk of misinformation. To alleviate it, we can train an additional classifier to discriminate the fakes. We believe the benefits outweigh the downsides. The proposed method lowers the requirements for designing many special models, which may cause unemployment for people with related occupations, such as audio engineers.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We state the claims made, including the contributions made in the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section E in Appendix.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA] .

Justification: **[TODO]**

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section B in Appendix with data, and model configurations information.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [NA]

Justification: **[TODO]**

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: See Section B in Appendix with data and model configurations information. We also report hyper-parameters of LLM-Codec.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report the mean value for experiments to run 5 times.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Section B in Appendix for information on compute resources.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: **[TODO]**

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Section F in Appendix.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [No]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: **[TODO]**

    Justification: **[TODO]**

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [No]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: **[TODO]**

    Justification: **[TODO]**

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.