

Kimi-Audio Technical Report

Kimi Team

Abstract

We present Kimi-Audio, an open-source audio foundation model that excels in audio understanding, generation, and conversation. We detail the practices in building Kimi-Audio, including model architecture, data curation, training recipe, inference deployment, and evaluation. Specifically, we leverage a 12.5hz audio tokenizer, design a novel LLM-based architecture with continuous features as input and discrete tokens as output, and develop a chunk-wise streaming detokenizer based on flow matching. We curate a pre-training dataset that consists of more than 13 million hours of audio data covering a wide range of modalities including speech, sound, and music, and build a pipeline to construct high-quality and diverse post-training data. Initialized from a pre-trained LLM, Kimi-Audio is continual pre-trained on both audio and text data with several carefully designed tasks, and then fine-tuned to support a diverse of audio-related tasks. Extensive evaluation shows that Kimi-Audio achieves state-of-the-art performance on a range of audio benchmarks including speech recognition, audio understanding, audio question answering, and speech conversation. We release the codes, model checkpoints, as well as the evaluation toolkits in <https://github.com/MoonshotAI/Kimi-Audio>.

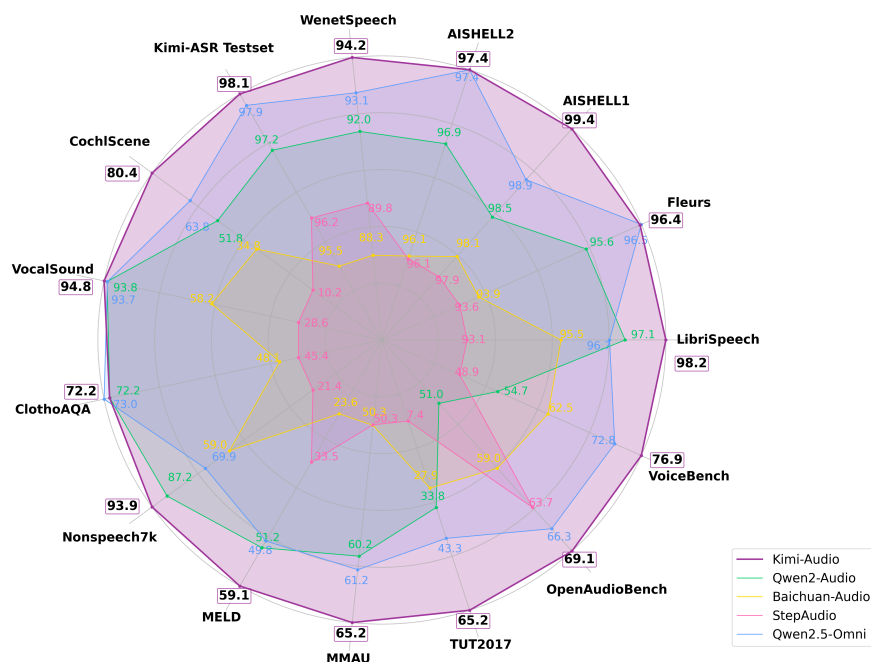


Figure 1: Performance of Kimi-Audio and previous audio language models including Qwen2-Audio [11], Baichuan-Audio [41], Step-Audio [28], and Qwen2.5-Omni [73] on various benchmarks.

1 Introduction

Audio plays an indispensable role in human daily life, such as environment perception, speech conversation, emotion expression, and music appreciation, and is an important topic in artificial general intelligence. Traditional audio modeling, constrained by the development of artificial intelligence, handles each audio processing task (e.g., speech recognition, emotion recognition, sound event detection, and speech conversation) separately. However, audio is naturally sequential and speech has strict correspondence with text, which makes it suitable to take advantage of the rapid progress in large language models (LLMs) in audio modeling. Just as natural language processing has experienced, audio processing evolves quickly from separate models for separate tasks to a universal model handling a variety of tasks.

For example, pioneer works introduce language models in audio generation [3, 77], audio understanding [10, 63, 9], speech recognition [58, 87], speech synthesis [70, 80], and end-to-end speech conversation [29, 14]. However, previous works fall short of building a universal audio foundation model for a variety of audio processing tasks in several aspects: 1) not universal but only focus on a specific type of tasks, such as audio understanding [10, 11, 21, 63, 88, 35], audio generation [45, 77], or speech conversation [14, 84]; 2) not much emphasis on audio pre-training but only fine-tuning an LLM on downstream audio tasks [10, 63, 9]; 3) no access to source codes and checkpoints, with limited value to the community [29, 7].

In this report, we present Kimi-Audio, an open-source audio foundation model that handles a variety of audio processing tasks. We detail our effort in building a state-of-the-art (SOTA) audio foundation model in three aspects: architecture, data, and training.

- **Architecture.** Our model consists of three components: an audio tokenizer and detokenizer as audio I/O, and an audio LLM as the core processing part (see Section 2.1). We use discrete semantic audio tokens as the basic representation for both the input and output of the audio LLM. Meanwhile, we concatenate the semantic audio token with continuous acoustic vectors in the input to enhance perception capability, and concatenate with discrete text tokens in the output to enhance the generation capability. In this way, we can achieve good audio perception and generation capabilities at the same time, facilitate universal audio modeling. We reduce the number of token per second in audio to bridge the gap between text and audio sequence and set the compression rate of both the semantic and acoustic audio tokens as 12.5Hz. The detailed design of the audio tokenizer for both discrete semantic tokens and continuous acoustic vectors, as well as the generation of both discrete semantic tokens and text tokens are introduced in Section 2.2 and 2.3 respectively.
- **Data.** To achieve SOTA universal audio modeling, we need to pre-train the model on a large amount of audio data to see diverse scenarios. To this end, we crawl and process a large-scale audio pre-training dataset. We develop a data processing pipeline consisting of speech enhancement, diarization, transcription, filtering, etc, to enable high data quality (see Section 3.1). To support diverse audio processing tasks, we curate a large amount of task-specific data for supervised fine-tuning (SFT). We demonstrate an economic way to construct most of SFT data with pure open and accessible data sources and processing tools to achieve SOTA performance, without relying on any data purchase (see Section 3.2).
- **Training.** To achieve good audio understanding/generation capability while maintaining high knowledge capacity and intelligence, we initialize the audio LLM with a pre-trained LLM, and carefully design a series of pre-training tasks to fully learn the audio data and bridge the gap

between text and audio. Specifically, the pre-training tasks can be divided into three categories: 1) text-only and audio-only pre-training, which aims to learn the knowledge from text and audio domains separately; 2) audio-to-text mapping, which encourages the conversion between audio and text; 3) audio-text interleaving, which further bridge the gap between text and audio (see Section 4.1). In the supervised fine-tuning stage, we develop a training recipe to improve fine-tuning efficiency and task generalization (see Section 4.2).

Furthermore, we introduce the practices for deploying and serving our audio foundation model for inference in Kimi APP, as described in Section 5. Evaluating and benchmarking an audio foundation model in various downstream tasks such as speech recognition, audio understanding, and speech conversation is challenging. We encounter tricky issues in fairly comparing different audio models, such as non-standardized metric, evaluation protocol, and inference hyper-parameters. Therefore, we develop an evaluation toolkit that can faithfully evaluate audio LLMs on comprehensive benchmarks (see Section 6.1). We open-source this toolkit to facilitate a fair comparison in the community.

Based on our evaluation toolkit, we conduct a comprehensive evaluation of Kimi-Audio and other audio LLMs on a variety of audio benchmarks (see Section 6.2). Evaluation results demonstrate that Kimi-Audio achieves SOTA performance in a series of audio tasks, including speech recognition, audio understanding, audio-to-text chat, and speech conversation. We open source the codes and checkpoints of Kimi-Audio, as well as the evaluation toolkit in <https://github.com/MoonshotAI/Kimi-Audio>, to boost the development of the community.

2 Architecture

2.1 Overview

Kimi-Audio is an audio foundation model designed to perform comprehensive audio understanding, generation, and conversation tasks within a unified architecture. As illustrated in Figure 2, our system comprises three primary components: (1) an audio tokenizer that converts input audio into discrete semantic tokens derived through vector quantization with a 12.5Hz frame rate. The audio tokenizer additionally extracts continuous acoustic vectors to enhance perception capability. (2) an audio LLM that generates semantic tokens together with text token to improve the generation capability, featuring shared transformer layers that process multimodal inputs before branching into specialized parallel heads for text and audio generation; and (3) an audio detokenizer that converts the discrete semantic tokens predicted by the audio LLM back into coherent audio waveforms using a flow matching approach. This integrated architecture enables Kimi-Audio to seamlessly handle diverse audio-language tasks from speech recognition and understanding to speech conversation within a single unified model framework.

2.2 Audio Tokenizer

Our audio foundation model employs a hybrid audio tokenization strategy, integrating discrete semantic tokens and complementary continuous vectors of acoustic information to effectively represent speech signals for downstream tasks. This tokenization allows the model to leverage the efficiency and semantic focus of discrete tokens while benefiting from the rich acoustic details captured by continuous representations.

We incorporate the discrete semantic tokens proposed by GLM-4-Voice [84]. This component utilizes a supervised speech tokenizer derived from an automatic speech recognition (ASR) model.

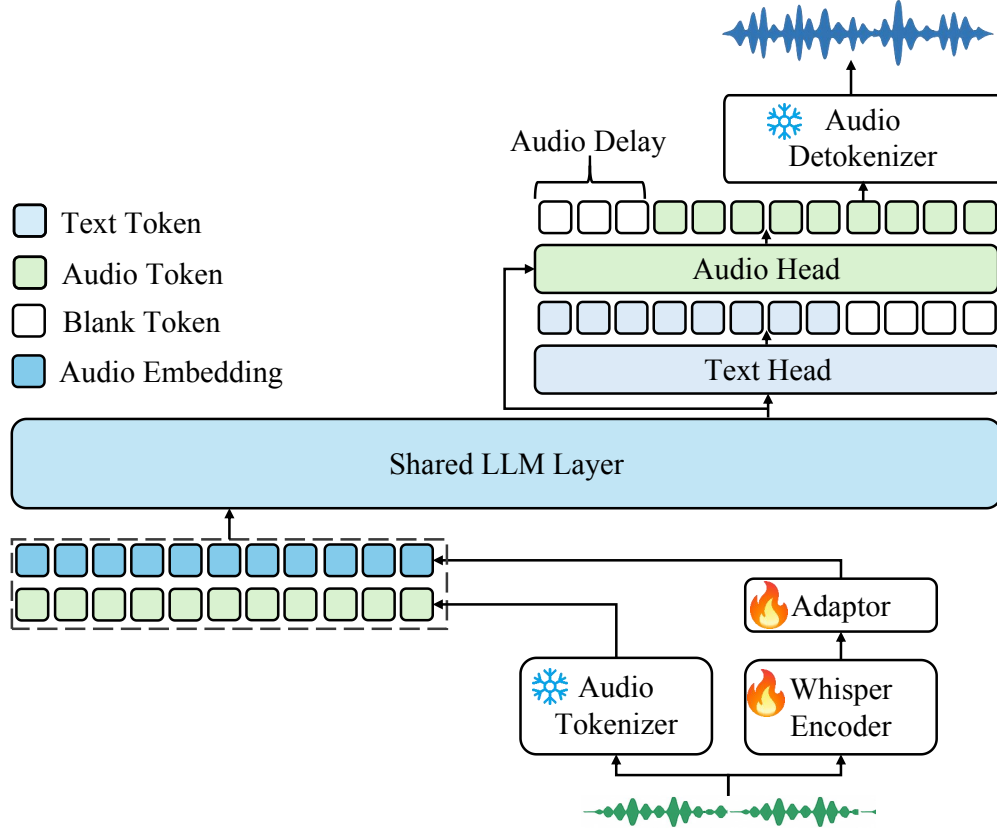


Figure 2: Overview of the Kimi-Audio model architecture: (1) an audio tokenizer that extracts discrete semantic tokens and a Whisper encoder that generates continuous acoustic features; (2) an audio LLM that processes audio inputs and generates text and/or audio outputs; (3) an audio detokenizer converts audio tokens into waveforms.

By introducing a vector quantization layer within the whisper encoder architecture [58], we can transform continuous speech representations into a sequence of discrete tokens at a low frame rate (i.e. 12.5Hz) using a single codebook.

Complementing the discrete semantic tokens, we incorporate a continuous feature representation derived from a pre-trained whisper model [58] to enhance the perception capability of our model. Since the whisper feature has a frame rate of 50Hz, we additionally introduce an adaptor upon the whisper feature extractor to downsample the feature from 50Hz to 12.5Hz. The downsampled features are added to the embeddings of discrete semantic tokens to serve as the input of the audio LLM.

By combining discrete semantic tokens with continuous whisper features, our model benefits from efficient, semantically grounded representation and detailed acoustic modeling, providing a comprehensive foundation for diverse audio processing tasks.

2.3 Audio LLM

The core of our system is an audio LLM designed to process the audio representations generated by the tokenization strategy described in Section 2.2 and produce multimodal outputs, which include the discrete semantic tokens of audio and the corresponding text tokens to improve the generation capability.

To enable the model to generate both audio semantic tokens and the corresponding textual responses, we adapt the standard LLM architecture by structuring it into components with shared and specialized functions. A significant portion of the original transformer bottom layers, i.e., the first several layers, are utilized as shared layers. These layers process the input sequence and learn cross-modal representations, integrating information from both text and audio modalities present in the input or context. Based on these shared layers, the architecture diverges into two parallel heads containing transformer layers. The first head is a text head that is specifically responsible for autoregressively predicting text tokens, forming the textual output of the model. The second head is an audio head to predict the discrete audio semantic tokens. These predicted audio tokens are subsequently passed to an audio detokenizer module to synthesize the final output audio waveform.

To take advantage of the strong language capabilities of the pre-trained text LLMs [76, 24, 13], the parameters of the shared transformer layers and the text head are initialized directly from the weights of the pre-trained text LLM. The audio head layers are initialized randomly. This initialization strategy ensures that the model retains robust text understanding and generation capabilities while learning to effectively process and generate audio information.

2.4 Audio Detokenizer

The audio detokenizer aims to generate high-quality and expressive speech conditioned on discrete semantic audio tokens. We employ the same detokenizer architecture as in MoonCast [32], which contains two parts: 1) a flow-matching module which converts 12.5Hz semantic tokens to 50Hz mel-spectrograms; 2) a vocoder which generates waveforms from mel-spectrograms. To reduce speech generation latency, we design a chunk-wise streaming detokenizer. Intuitively, we can split the semantic tokens into chunks and decode them separately, which, however, in our preliminary experiments, faces an intermittent issue in the chunk boundaries. Thus, we propose a chunk-wise autoregressive streaming framework with a look-ahead mechanism.

Chunk-wise Autoregressive Streaming Framework. We split the audio into chunks (e.g., 1 second per chunk): $\{c_1, c_2, \dots, c_i, \dots, c_N\}$, where N is the number of chunks. Firstly, to match the sequence length between semantic tokens (12.5Hz) and mel-spectrograms (50Hz), we upsample the semantic tokens by 4x rate. Secondly, we apply a chunk-wise causal mask during training and inference, i.e., for chunk c_i , all previous chunks c_j with $j < i$ are prompts. We denote chunk c_i 's mel-spectrograms as m_i and the corresponding discrete semantic audio tokens as a_i^d . The flow-matching model's forward step will mix m_i with Gaussian noise and the backward step will remove noise to obtain clean m_i with condition a_i^d and prompt c_j , where $j < i$, and c_j contains both m_j and a_j^d . With this design, during inference, when the LLM generates a chunk, we employ the flow-matching model to detokenize it to obtain the mel-spectrograms. Finally, we apply a BigVGAN [38] vocoder to generate waveforms for each chunk.

Look-Ahead Mechanism. With a preliminary study, we find that the generated audio in the boundaries of chunks still has an intermittent issue. Although a long range of history context has been seen during the diffusion denoising process, the future context of the boundary position cannot be seen due to the nature of block-wise causal attention, which causes the degradation of quality. Thus, we propose a look-ahead mechanism. In detail, for chunk c_i , we take the future n (e.g. 4) semantic tokens from chunk c_{i+1} and concatenate them to the end of c_i to form \hat{c}_i . Then we detokenize \hat{c}_i to generate the mel-spectrograms, but only retain the mel-spectrograms corresponding to c_i . This mechanism is training-free and will only delay the generation of the first chunk by n tokens.

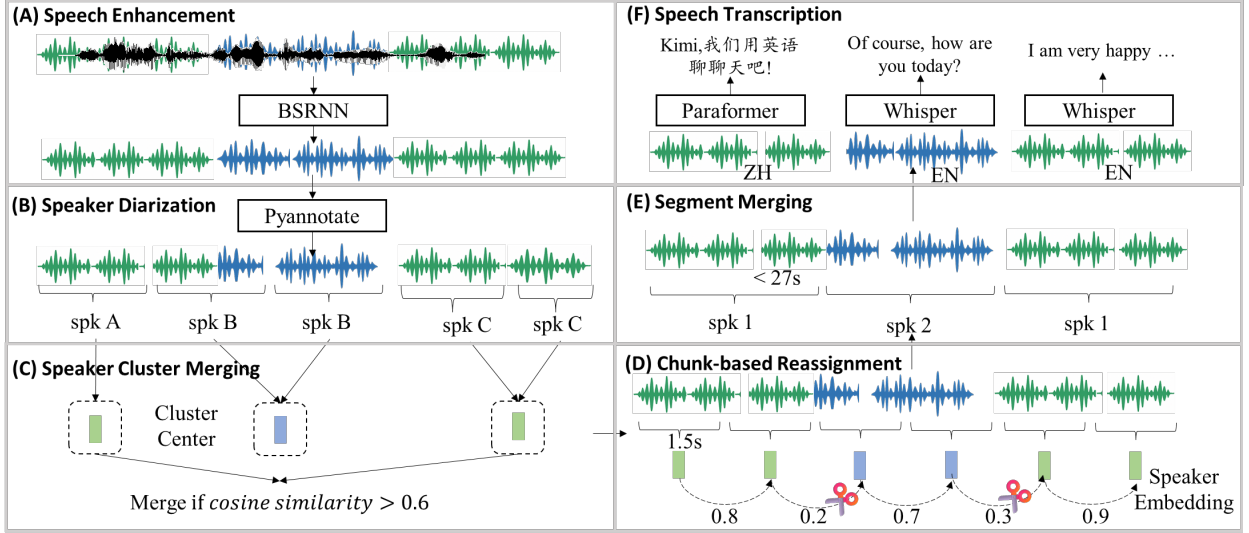


Figure 3: Processing pipeline for the audio pre-training data.

3 Data

3.1 Pre-Training Data

Our pre-training corpus comprises both unimodal (text-only, audio-only) and multimodal (text-audio) data. The audio-only pre-training data covers a wide range of real-world scenarios, including audiobooks, podcasts, and interviews, and consists of approximately 13 million hours of raw audio containing rich acoustic events, music, environmental sound, human vocalization, and multilingual information. The details of the text-only pre-training data can be found in [65].

Most audio corpus contains only raw audio without corresponding transcriptions, language types, speaker annotations, and segmentation boundaries. In addition, the raw audio often contains undesired artifacts such as background noise, reverberation, and speaker overlap.

Inspired by previous work [32, 81, 26], we develop an efficient automatic audio data processing pipeline to generate high-quality annotations, resulting in our multimodal (audio-text) data.

Compared to previous data processing pipelines that primarily focus on generating high-quality short audio segments without contextual information, our pipeline is designed to provide long-form audio annotations with consistent long-range context. The pipeline includes the following key components in a step-by-step manner, as shown in Figure 3 and described as follows.

Speech Enhancement. To suppress undesired background noise and reverberation, we develop a speech enhancement model based on the Band-Split RNN (BSRNN) architecture [49], as shown in Figure 3(A). Following the same hyper-parameter configuration as in [82], the model is applied to perform 48kHz speech enhancement. Empirically, we find that speech enhancement will remove the environmental sound and music, which can be harmful to audio understanding. Thus, we randomly choose original or enhanced audio with a ratio of 1 : 1 in the pre-training stage.

Segmentation by Diarization. We employ a diarization-driven approach to segment long-form audio. We utilize the PyAnnote toolkit¹ for speaker diarization (Figure 3(B)), which segments the

¹<https://github.com/pyannote/pyannote-audio>

audio and assigns speaker labels. However, the raw output is sub-optimal, and thus we develop a post-processing pipeline to address the issues in the previous segmentation results:

- **Speaker Cluster Merging.** We observe that PyAnnote sometimes assigns multiple speaker labels to the same actual speaker, which results in speaker fragmentation. We compute representative speaker embeddings for each initial cluster and merge pairs of clusters whose embeddings have a cosine similarity greater than 0.6, as shown in Figure 3(C).
- **Chunk-based Reassignment.** The initial diarization occasionally produced segments containing multiple speakers. To purify the segments, 1) we first divide all segments into 1.5-second chunks, and then 2) for each pair of adjacent chunks, if their cosine similarity is below 0.5, we treat them as belonging to different speakers and reassign each chunk to the speaker cluster with the highest similarity, as shown in Figure 3(D).
- **Segment Merging.** The initial diarization could result in segments of highly variable and sometimes impractical lengths (shorter than 1s or longer than 100s). So we iteratively merge adjacent segments labeled with the same speaker (after the reassignment step). The merging process terminates if the accumulated segment length exceeds 27 seconds or the silence gap between two segments is greater than 2 seconds, as shown in Figure 3(E).

The resulting segmentation from this refined diarization process provides more accurate and consistently sized speaker turns compared to the baseline diarization output.

Speech Transcription. To obtain the language type and text transcription for each speech segment, we first apply the Whisper-large-v3 model [58]² to detect the spoken language type. In this work, we retain only English and Mandarin segments for further transcription. For English segments, we directly use Whisper-large-v3 to generate both transcriptions and punctuation annotations. For Mandarin segments, we utilize the Paraformer-Zh [20] model from the FunASR toolkit³ to generate transcriptions along with character-level timestamps. Since Paraformer-Zh cannot output punctuation annotations, we add punctuation annotations with the following strategy: if the time gap between two consecutive characters is greater than 0.5 seconds but less than 1.0 second, we insert a “comma”; if the gap exceeds 1.0 second, we insert a “period”.

Implementation. The data processing pipeline is deployed on a cluster of 30 cloud instances. Each instance is equipped with 128 virtual CPUs (vCores), 1 TB of RAM, and 8 NVIDIA L20 GPUs, powered by Intel Xeon Platinum 8575C processors that support vectorized acceleration instructions, including Advanced Matrix Extensions (AMX). In total, the cluster provides 3,840 vCores, 30 TB of memory, and 240 NVIDIA L20 GPUs. Following extensive optimization, the pipeline achieves a daily processing throughput of approximately 200,000 hours of raw audio data.

3.2 SFT Data

After the pre-training stage, we perform supervised fine-tuning (SFT) to enhance the performance of Kimi-Audio on instruction following and audio processing. The SFT data can be mainly categorized into three parts: audio understanding, speech conversation, and audio-to-text chat.

3.2.1 Audio Understanding

²<https://huggingface.co/openai/whisper-large-v3>

³<https://github.com/modelscope/FunASR>

¹<https://github.com/fighting41love/zhvoice>

Table 1: List of datasets used for audio understanding and their training epoch in SFT stage.

Dataset	Audio Length (#hours)	Task Type	SFT Epochs
WenetSpeech [85]	10,518	ASR	2.0
WenetSpeech4TTS [50]	12,085	ASR	2.0
AISHELL-1 [4]	155	ASR	2.0
AISHELL-2 [17]	1,036	ASR	2.0
AISHELL-3 [62]	65	ASR	2.0
Emilla [25]	98,305	ASR	2.0
Fleurs [12]	17	ASR	2.0
CommonVoice [1]	43	ASR	2.0
KeSpeech [64]	1,428	ASR	2.0
Magicdata [79]	747	ASR	2.0
zhvoice ¹	901	ASR	2.0
Libriheavy [33]	51,448	ASR	2.0
MLS [57]	45,042	ASR	2.0
Gigaspeech [5]	10,288	ASR	2.0
LibriSpeech [54]	960	ASR	2.0
CommonVoice [1]	1,854	ASR	2.0
Voxpopuli [69]	529	ASR	2.0
LibriTTS [83]	568	ASR	2.0
CompA-R [22]	159	AQA	2.0
ClothoAQA [43]	7.4	AQA	4.0
AudioCaps [34]	137	AAC	2.0
Clotho-v2 [16]	24.0	AAC	2.0
MACS [51]	10.9	AAC	2.0
FSD50k [19]	80.8	SEC	2.0
CochlScene [31]	169.0	ASC	2.0
Nonspeech7k [59]	6.2	SEC	4.0
MusicAVQA _{audio-only} [39]	77.1	AQA	2.0
WavCaps [52]	3,793.3	AAC	2.0
AVQA _{audio-only} [78]	112	AQA	2.0
IEMOCAP [68]	10	SER	2.0
MELD [56]	9	SER	2.0
RAVDESS [47]	3	SER	2.0
SAVEE [30]	0.1	SER	2.0
ESD [89]	29	SER	2.0
TUT2016 [53]	10	ASC	2.0
TUT2017 [53]	13	ASC	4.0
TAU2022 [27]	67	ASC	2.0
ESC50 [55]	1	SEC	2.0
VocalSound [23]	19	SEC	4.0
VGGSound [6]	513	SEC	2.0
UrbanSound8K [61]	9	SEC	2.0
FSD50K [19]	74	SEC	2.0
Kimi Inhouse ASR Data	55,000	ASR	2.0
Kimi Inhouse Audio Data	5,200	AAC/AQA	2.0

We mainly leverage open-source datasets for audio understanding. The collected datasets include 6 tasks: Automatic Speech Recognition (ASR), Audio Question Answer (AQA), Automated Audio Caption (AAC), Speech Emotion Recognition (SER), Sound Event Classification (SEC), and Audio Scene Classification (ASC). The details of the datasets and the corresponding training epochs in the SFT stage are shown in Table 1. Besides the open-source datasets, we also utilize 55,000 hours in-house ASR data and 5,200 hours in-house audio data covering the AAC/AQA tasks.

3.2.2 Speech Conversation

To activate the Kimi-Audio model’s ability to generate speech with diverse styles and high expressiveness in different conversation scenarios, we construct a large volume of speech conversation data, which consists of multi-turn conversations made up with a series of user queries and assistant responses. For user queries, we instruct LLMs to write the text of user queries and then convert them into speech with our Kimi-TTS system, where the prompt speech is randomly selected from a large timbre set containing more than 125K timbres. For the assistant responses, we first select a voice actor as our Kimi-Audio speaker and synthesize the assistant responses with appropriate style and emotion with this single timbre. In the following, we introduce the data recording process for the Kimi-Audio speaker, as well as the Kimi-TTS and Kimi-VC systems used to synthesize assistant responses with diverse styles and expressiveness.

Data Recording for Kimi-Audio Speaker. To achieve diverse and highly expressive styles and emotions in the generated speech, we select a voice actor as the Kimi-Audio speaker and meticulously record a dataset of this speaker in a professional recording studio. We pre-define over 20 styles and emotions for recording, with each emotion further divided into 5 levels to represent varying emotional intensities. For each style and emotional level, we record an audio as the reference to maintain the consistency of the emotion and style among different text sentences. The whole recording process is guided by a professional recording director.

Kimi-TTS. We develop a zero-shot text-to-speech synthesis (TTS) system, called Kimi-TTS, to generate speech with only a 3-second prompt, while preserving the timbre, emotion, and style of the prompt speech. With the help of Kimi-TTS, we can synthesize speech for 1) the query text in diverse speakers/timbres with a large timbre set; 2) the response text with the styles and emotions recorded by the Kimi-Audio speaker, a voice actor selected by Kimi. Similar to the architecture of MoonCast [32], Kimi-TTS employs an LLM to generate speech tokens given the prompt speech and input text. Then a flow-matching-based speech detokenizer is used to generate high-quality speech waveforms. We train Kimi-TTS on about 1M hours generated by the automatic data pipeline (Section 3.1) and apply reinforcement learning to further enhance the robustness and quality of the generated speech.

Kimi-VC. Since it is difficult for the voice actor to record speech in any styles, emotions, and accents, we develop a voice conversion (VC) system, called Kimi-VC, to convert diverse and in-the-wild speech in different speakers/timbres into the timbre of Kimi-Audio speaker while preserving the styles, emotions, and accents. Built on the Seed-VC framework [46], Kimi-VC incorporates source timbre perturbation via a timbre-shifting model during training, which mitigates information leakage and ensures alignment between training and inference phases. To ensure high quality of voice conversion, we fine-tune the Kimi-VC model using speech data recorded by the Kimi-Audio speaker, a voice actor selected by Kimi.

3.2.3 Audio-to-Text Chat

To help Kimi-Audio with the basic ability on chat, we collect open-source supervised fine-tuning data from text domain, as listed in Table 2, and then convert the user queries to speech with a variety of timbres, resulting in the audio-to-text chat data whose user query is speech while the assistant response is text. Considering that some text cannot be easily converted to speech, we perform several preprocessing steps on text by 1) filtering out text containing complex math, code, table, complex multilingual content, or too long content, 2) making colloquial rewriting, and 3) convert

Table 2: List of text dataset used in audio-to-text chat with their training epochs in SFT stage.

Dataset	#Samples	SFT Epochs
Magpie-Pro [75]	300K	2.0
Magpie-MT [75]	300K	2.0
Evol-Instruct [15]	143K	2.0
Evol-Instruct-Code [15]	80K	2.0
Infinity-Instruct [2]	7M	2.0
Synthia [67]	119K	2.0
NuminaMath [40]	860K	2.0
Tulu3 [36]	900K	2.0
OpenHermes-2.5 [66]	1M	2.0
OpenOrca [42]	2M	2.0

a single-turn question-answer data with complex instruction into multi-turn data with easy and concise instructions.

4 Training

4.1 Pre-training

The pre-training stage of Kimi-Audio aims to learn the knowledge from both the real-world audio and text domains and align them in the model’s latent space, thereby facilitating complex tasks such as audio understanding, audio-to-text chat, and speech conversation. To this end, we design several pre-training tasks with the following aspects: 1) pre-training in the unimodality (i.e., audio and text) to learn the knowledge from each domain individually in Section 4.1.1; 2) learning audio-text mapping in Section 4.1.2; 3) three audio-text interleaving tasks to further bridge two modalities in Section 4.1.3.

Formally, given a raw audio A , the data pre-processing pipeline (described in Section 3.1) splits it into a series of segments $\{S_1, S_2, \dots, S_N\}$, and each segment $S_i, i \in [1, N]$ consists of an audio a_i and the corresponding transcription t_i . Furthermore, as illustrated in Section 2.2, for an audio segment a_i , we extract both the continuous acoustic vectors a_i^c and the discrete semantic tokens a_i^d . To comply with the design of our model architecture in Section 2 which takes discrete semantic audio tokens as the main representation of the input and output, while adding continuous acoustic audio tokens in the input and discrete text tokens in the output, we denote the training sequence as $\{a_1^c/a_1^d/t_1, a_2^c/a_2^d/t_2, \dots, a_N^c/a_N^d/t_N\}$, where $a_i^c/a_i^d/t_i$ denotes the semantic audio, acoustic audio, and text sequence for segment i . We make sure that the audio and text sequences have the same lengths by appending blank tokens to the shorter sequence. The actual pre-training segments can be either one or two of $a_i^c/a_i^d/t_i$, such as $a_i^d, t_i, a_i^c/a_i^d$, or a_i^d/t_i . For a_i^c/a_i^d , we add the continuous vectors a_i^c and the semantic tokens a_i^d (the semantic tokens will be converted into embedding using a lookup table) to obtain the final audio feature a_i . Thus, we use a_i to represent a_i^c/a_i^d for short. For a_i^d/t_i , we add the lookup embedding of the semantic tokens and text tokens as input and generate each token with its respective head, as described in Section 2.

With this notation, we formulate the following pre-training tasks in Table 3 and introduce them as follows.

Table 3: List of pre-training tasks. We design three categories pre-training tasks, including: 1) audio/text unimodal pre-training; 2) audio-text mapping pre-training; 3) audio-text interleaving pre-training. Notation: a_i^d and a_i^c denotes the discrete semantic tokens and continuous acoustic vectors for audio segment i respectively, a_i denotes the combination of a_i^d and a_i^c for audio segment i , underline means it will receive loss during training.

Category	Pre-training Task	Task Formulation	Task Weight
Audio/Text Unimodal	Text Only	$\underline{t_1}, \underline{t_2}, \dots, \underline{t_N}$	7
	Audio Only	$\underline{a_1^d}, \underline{a_2^d}, \dots, \underline{a_N^d}$	1
Audio-Text Mapping	Audio to Text	$a_1, \underline{t_1}, a_2, \underline{t_2}, \dots, a_N, \underline{t_N}$	1
	Text to Audio	$t_1, \underline{a_1^d}, t_2, \underline{a_2^d}, \dots, t_N, \underline{a_N^d}$	1
Audio-Text Interleaving	Audio to Semantic	$a_1, \underline{a_2^d}, a_3, \underline{a_4^d}, \dots, a_{N-1}, \underline{a_N^d}$	1
	Audio to Text	$a_1, \underline{t_2}, a_3, \underline{t_4}, \dots, a_{N-1}, \underline{t_N}$	1
	Audio to Semantic and Text	$a_1, \underline{a_2^d/t_2}, a_3, \underline{a_4^d/t_4}, \dots, a_{N-1}, \underline{a_N^d/t_N}$	2

4.1.1 Audio/Text Unimodal Pre-training

We first learn the knowledge of text and audio separately. For text pre-training, we directly utilize the text data in MoonLight [44], which is high-quality and comprehensive for training large language models. We apply next-token prediction on text tokens solely. For audio pre-training, for each segment S_i , we apply next-token prediction on its discrete semantic token sequence a_i^d .

4.1.2 Audio-Text Mapping Pre-training

Intuitively, in order to align audio and text in a unified space, it is helpful to learn a mapping between two modalities. Thus, we design the automatic speech recognition (ASR) and text-to-speech synthesis (TTS) pre-training tasks. For ASR, we formulate the training sequence as $\{a_1, t_1, a_2, t_2, \dots, a_N, t_N\}$. For TTS, we formulate the training sequence as $\{t_1, a_1^d, t_2, a_2^d, \dots, t_N, a_N^d\}$. We only calculate the loss on text tokens for ASR and on audio semantic tokens for TTS.

4.1.3 Audio-Text Interleaving Pre-training

To further bridge the gap between audio and text modalities, we design three audio-text interleaving pre-training tasks.

- **Audio to semantic token interleaving.** We formulate the training sequence as $\{a_1, a_2^d, a_3, a_4^d, \dots, a_{N-1}, a_N^d\}$ ⁴. Then we only calculate the loss on the semantic audio tokens a_i^d , but not on a_{i-1} .
- **Audio to text interleaving.** We formulate the training sequence as $\{a_1, t_2, a_3, t_4, \dots, a_{N-1}, t_N\}$. We only calculate the loss on text tokens t_i .
- **Audio to semantic token + text interleaving.** We formulate the training sequence as $\{a_1, a_2^d/t_2, a_3, a_4^d/t_4, \dots, a_{N-1}, a_N^d/t_N\}$. For a_i^d/t_i , as the semantic audio token sequence is always longer than the text token sequence, the prediction of the semantic token is like a streaming text-to-speech task as in Section 4.1.2. Empirically, we find that the prediction of the first few semantic tokens is hard because the model needs to concurrently predict the next text token and its semantic audio token. We address this issue by delaying the prediction of the first several semantic audio tokens by prepending 6 special blank tokens (6 is determined by trading off the generation quality and latency according to preliminary experiments) to the semantic audio tokens.

⁴It is also possible that the first segment is a_1^d , or the last segment is a_N .

4.1.4 Pre-training Recipe

We initialize the audio LLM of Kimi-Audio from the pre-trained Qwen2.5 7B model [76] and extend its vocabulary with semantic audio tokens and special tokens. We perform pre-training on the above pre-training tasks with the corresponding task weight 1 : 7 : 1 : 1 : 1 : 1 : 2, as shown in Table 3. We pre-train Kimi-Audio using 585B audio tokens and 585B text tokens with 1 epoch. We use AdamW [48] optimizer with a learning rate schedule from $2e^{-5}$ to $2e^{-6}$ in a cosine decay. We use 1% tokens for learning rate warmup.

The continuous acoustic feature extraction module in audio tokenizer is initialized from Whisper large-v3 [58], which can capture the fine-grained acoustic characteristics inherent in the input audio signal. During the initial phases of model pretraining (about 20% tokens in pretraining), the parameters of this whisper-based feature extractor are kept frozen. Subsequently, the feature extractor is unfrozen, enabling its parameters to be fine-tuned jointly with the rest of the model, allowing it to adapt more specifically to the nuances of the training data and the requirements of the target tasks.

4.2 Supervised Fine-tuning

4.2.1 Formulation

After pre-training Kimi-Audio with massive real-world audio and text data, we perform supervised fine-tuning to equip it with the ability of instruction-following. We have the following design choices: 1) considering the downstream tasks are diverse, we do not set special task switching operations, but use natural language as instructions for each task; 2) for instruction, we construct both the audio and text versions (i.e., the audio is generated by Kimi-TTS in a zero-shot way given the text) and randomly choose one during training; 3) to enhance the robustness of instruction-following capability, we construct 200 instructions for ASR task and 30 instructions for other tasks by LLM and randomly choose one for each training sample. As described in Section 3.2, we build about 300K hours of data for supervised fine-tuning.

4.2.2 Fine-tuning Recipe

As shown in Table 1 and Table 2, we fine-tune Kimi-Audio on each data source with 2-4 epochs based on comprehensive ablation experiments. We use AdamW [48] optimizer with a learning rate schedule from $1e^{-5}$ to $1e^{-6}$ in a cosine decay. We use 10% tokens for learning rate warmup.

4.3 Training of Audio Detokenizer

We train the audio detokenizer in three stages. Firstly, we use about 1M hours of audio from the pre-training data described in Section 3.1 and pre-train both the flow-matching model and the vocoder to learn audio with diverse timbre, prosody, and quality. Secondly, we adopt the chunk-wise fine-tuning strategy with a dynamic chunk size from 0.5 seconds to 3 seconds on the same pre-training data [32]. Finally, we fine-tune on the high-quality single-speaker recording data from the Kimi-Audio speaker.

5 Inference and Deployment

Kimi-Audio is designed to handle various audio-related tasks, such as speech recognition, audio understanding, audio-to-text chat, and speech-to-speech conversation. We take real-time speech-

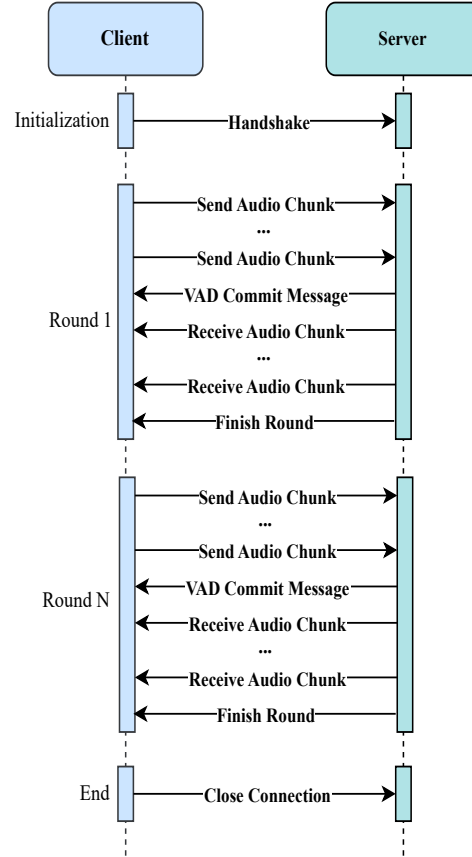


Figure 4: The client-server communication for real-time speech-to-speech conversation in Kimi-Audio.

to-speech conversation as an example to illustrate the practices in Kimi-Audio deployment, since this task is more complicated than the rest of audio tasks in terms of infrastructure and engineering efforts. We first introduce the workflow of real-time speech conversation between the client (e.g., Kimi APP or web browser) and the server (Kimi-Audio service) and then describe the practices of product deployment.

5.1 Workflow of Real-Time Speech Conversation

The workflow of a speech-to-speech conversation between the user client (e.g., Kimi APP) and the server (Kimi-Audio service) is illustrated in Figure 4. The workflow proceeds in the following manner for each conversation round:

- The user speaks to the client (e.g., Kimi APP or web browser), and the audio data is collected and streamed to the server.
- On the server side, a voice activity detection (VAD) module determines if the user has finished speaking.
- Once the user stops speaking, the server sends a commit signal and initiates the inference process of the Kimi-Audio model.
- During inference, the client receives audio chunks in real-time as they are generated and starts playing them for the user.

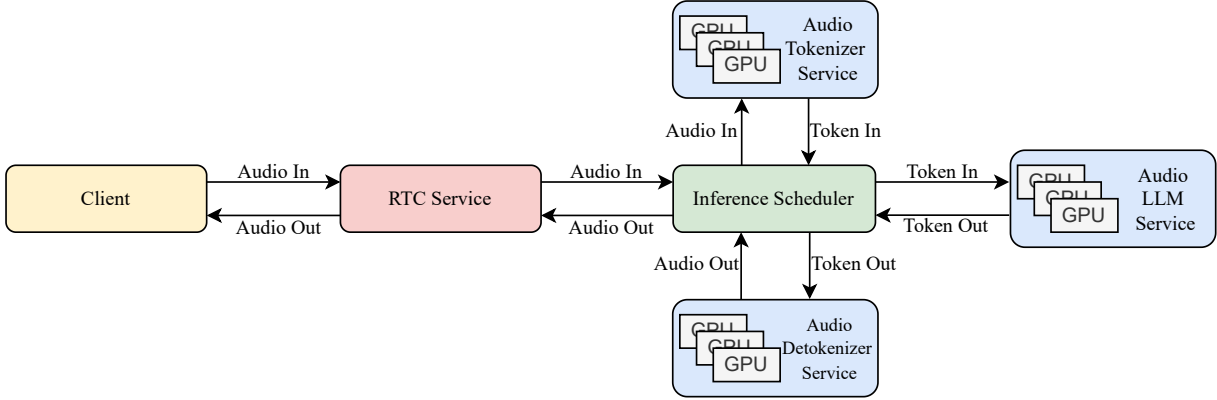


Figure 5: The workflow of production deployment for real-time speech-to-speech conversation in Kimi-Audio.

- The client (mobile phone or web browser) plays the received audio chunks back to the user.

The inference process of Kimi-Audio on the server side for each round follows these steps. First, the input audio is converted to discrete semantic tokens and continuous acoustic vectors using the audio tokenizer. Next, the input to the Audio LLM is assembled by concatenating the system prompt tokens, audio tokens, and conversation history tokens. The token sequence is then passed to the Audio LLM, which generates output tokens. Finally, the output tokens are converted back into an audio waveform using the detokenizer.

5.2 Production Deployment

As shown in Figure 5, in a production environment, all core components: Audio Tokenizer, Audio LLM, and Audio Detokenizer, are computationally intensive, requiring a scalable and efficient infrastructure. To address this, we designed the production deployment architecture as follows.

Kimi-Audio RTC Service. This service interfaces with the client, receiving audio from the user, forwarding it to the Inference Scheduler, and returning the generated audio chunks to the client. We use the WebRTC protocol to ensure a stable and low-latency connection.

Inference Scheduler. The Inference Scheduler manages the conversation flow by maintaining conversation history as tokens in a storage backend. For each interaction round, it performs the following steps:

- Call the Tokenizer Service to convert the user’s audio into tokens.
- Construct the model input by combining the new tokens with the conversation history.
- Send the input to the LLM Service to generate response tokens.
- Call the Detokenizer Service to convert the response tokens into audio output.

Additionally, it stores all output tokens as part of the ongoing conversation history to ensure continuity in the dialogue.

Tokenizer/Detokenizer/LLM Services: These services handle model inference and are equipped with a load balancer and multiple inference instances to handle requests in parallel, ensuring scalability.

This modular architecture ensures that Kimi-Audio can scale effectively to meet the performance demands of real-time speech interactions while maintaining low latency and high availability in production.

6 Evaluation

Evaluating audio foundation models and comparing with previous state-of-the-art systems are challenging, due to some inherent issues in the audio community. Thus, we first develop a fair, reproducible, and comprehensive evaluation toolkit for audio foundation models in Section 6.1, and then evaluate Kimi-Audio on a variety of audio processing tasks including speech recognition, general audio understanding, audio-to-text chat, and speech conversation, and compare Kimi-Audio with previous systems to demonstrate the advantages in Section 6.2.

6.1 Evaluation Toolkit

Even if an audio foundation model is fully open-source, it is still troublesome to reproduce the same results as reported in its paper or technical report, let alone those closed-source models. We analyze the challenges in evaluating and comparing audio foundation models in various audio processing tasks as follows:

- **Limitations in Metrics.** Current practices suffer from inconsistent metric implementations (e.g., variations in Word Error Rate calculation due to different text normalizations) and inadequate assessment methods (e.g., relying solely on exact string matching for tasks like audio question answering fails to capture the semantic correctness of complex LLM responses).
- **Diverse Configurations.** Reproducibility is severely hampered by the high sensitivity of model performance to inference parameters such as decoding temperature, system prompts, and task prompts.
- **Lack of Generation Evaluation** While progress has been made in understanding tasks, assessing the quality and coherence of the generated audio response still lacks benchmarks.

To address these critical limitations, we develop an open-source evaluation toolkit for audio foundation models on audio understanding, generation, and conversation tasks. It currently integrates and supports Kimi-Audio and a series of recent audio LLMs [11, 74, 84, 41, 28], and can be leveraged to evaluate any other audio foundation models. The toolkit provides the following features and benefits:

- We implement a standardized WER calculation (based on Qwen-2-Audio [11]) and integrate GPT-4o-mini as an intelligent judge (following [8]) for tasks like audio question answering. This approach overcomes the limitations of inconsistent metrics and simplistic string matching, enabling fair comparison.
- Our toolkit offers a single unified platform supporting diverse models and versions, simplifying side-by-side comparisons. It provides a crucial structure for defining and sharing standardized inference parameters and prompting strategies (“recipes”), thereby directly addressing inconsistencies in evaluation setups and fostering greater reproducibility across different research works.

Table 4: Performance of Kimi-Audio and baseline models on ASR task. Best results are in bold.

Datasets	Model	Performance (WER↓)
LibriSpeech [54] test-clean test-other	Qwen2-Audio-base	1.74 4.04
	Baichuan-Audio-base	3.02 6.04
	Step-Audio-chat	3.19 10.67
	Qwen2.5-Omni	2.37 4.21
	Kimi-Audio	1.28 2.42
Fleurs [12] zh en	Qwen2-Audio-base	3.63 5.20
	Baichuan-Audio-base	4.15 8.07
	Step-Audio-chat	4.26 8.56
	Qwen2.5-Omni	2.92 4.17
	Kimi-Audio	2.69 4.44
AISHELL-1 [4]	Qwen2-Audio-base	1.52
	Baichuan-Audio-base	1.93
	Step-Audio-chat	2.14
	Qwen2.5-Omni	1.13
	Kimi-Audio	0.60
AISHELL-2 [17] ios	Qwen2-Audio-base	3.08
	Baichuan-Audio-base	3.87
	Step-Audio-chat	3.89
	Qwen2.5-Omni	2.56
	Kimi-Audio	2.56
WenetSpeech [85] test-meeting test-net	Qwen2-Audio-base	8.40 7.64
	Baichuan-Audio-base	13.28 10.13
	Step-Audio-chat	10.83 9.47
	Qwen2.5-Omni	7.71 6.04
	Kimi-Audio	6.28 5.37
Kimi-ASR Internal Testset subset1 subset2	Qwen2-Audio-base	2.31 3.24
	Baichuan-Audio-base	3.41 5.60
	Step-Audio-chat	2.82 4.74
	Qwen2.5-Omni	1.53 2.68
	Kimi-Audio	1.42 2.44

- We record and release an evaluation benchmark to test the ability of audio LLMs on speech conversation from the perspective of 1) speech control on emotion, speed, and accent; 2) empathy conversation; and 3) diverse styles such as storytelling and tongue twister.

We open-source this toolkit to the community <https://github.com/MoonshotAI/Kimi-Audio-Evalkit>. We believe this toolkit can serve as a valuable asset to advance the field by promoting more reliable and comparable benchmarking. We actively encourage researchers and developers to utilize it, contribute by adding new models and datasets, and help refine standardized evaluation protocols and inference recipes. In this way, we can build a better ecosystem for the audio community.

6.2 Evaluation Results

In this section, based on our evaluation toolkit, we detail the evaluation of Kimi-Audio across a comprehensive suite of audio processing tasks, including Automatic Speech Recognition (ASR),

Table 5: Performance of Kimi-Audio and baseline models on audio understanding task. Best results are in bold.

Datasets	Model	Performance↑
MMAU [60] music sound speech	Qwen2-Audio-base	58.98 69.07 52.55
	Baichuan-chat	49.10 59.46 42.47
	GLM-4-Voice	38.92 43.54 32.43
	Step-Audio-chat	49.40 53.75 47.75
	Qwen2.5-Omni	62.16 67.57 53.92
	Kimi-Audio	61.68 73.27 60.66
ClothoQA [43] test dev	Qwen2-Audio-base	71.73 72.63
	Baichuan-chat	48.02 48.16
	Step-Audio-chat	45.84 44.98
	Qwen2.5-Omni	72.86 73.12
	Kimi-Audio	71.24 73.18
VocalSound [23]	Qwen2-Audio-base	93.82
	Baichuan-Audio-base	58.17
	Step-Audio-chat	28.58
	Qwen2.5-Omni	93.73
	Kimi-Audio	94.85
Nonspeech7k [59]	Qwen2-Audio-base	87.17
	Baichuan-chat	59.03
	Step-Audio-chat	21.38
	Qwen2.5-Omni	69.89
	Kimi-Audio	93.93
MELD [56]	Qwen2-Audio-base	51.23
	Baichuan-chat	23.59
	Step-Audio-chat	33.54
	Qwen2.5-Omni	49.83
	Kimi-Audio	59.13
TUT2017 [53]	Qwen2-Audio-base	33.83
	Baichuan-Audio-base	27.9
	Step-Audio-chat	7.41
	Qwen2.5-Omni	43.27
	Kimi-Audio	65.25
CochlScene [31] test dev	Qwen2-Audio-base	52.69 50.96
	Baichuan-Audio-base	34.93 34.56
	Step-Audio-chat	10.06 10.42
	Qwen2.5-Omni	63.82 63.82
	Kimi-Audio	79.84 80.99

audio understanding, audio-to-text chat, and speech conversation. We compare Kimi-Audio against other audio foundation models (Qwen2-Audio [11], Baichuan-Audio [41], Step-Audio [28], GLM-4-Voice [84], and Qwen2.5-Omini [73]) using established benchmarks and internal test sets.

6.2.1 Automatic Speech Recognition

The ASR capabilities of Kimi-Audio were evaluated on diverse datasets spanning multiple languages and acoustic conditions. As presented in Table 4, Kimi-Audio consistently demonstrates superior performance compared to previous models. We report Word Error Rate (WER) on these datasets, where lower values indicate better performance.

Table 6: Performance of Kimi-Audio and baseline models on the tasks of audio-to-text chat. Best results are in bold.

Datasets	Model	Performance↑
OpenAudioBench [41] AlpacaEval Llama Questions Reasoning QA TriviaQA Web Questions	Qwen2-Audio-chat	57.19 69.67 42.77 40.30 45.20
	Baichuan-chat	59.65 74.33 46.73 55.40 58.70
	GLM-4-Voice	57.89 76.00 47.43 51.80 55.40
	Step-Audio-chat	56.53 72.33 60.00 56.80 73.00
	Qwen2.5-Omni	72.76 75.33 63.76 57.06 62.80
	Kimi-Audio	75.73 79.33 58.02 62.10 70.20
VoiceBench [8] AlpacaEval CommonEval SD-QA MMSU	Qwen2-Audio-chat	3.69 3.40 35.35 35.43
	Baichuan-chat	4.00 3.39 49.64 48.80
	GLM-4-Voice	4.06 3.48 43.31 40.11
	Step-Audio-chat	3.99 2.99 46.84 28.72
	Qwen2.5-Omni	4.33 3.84 57.41 56.38
	Kimi-Audio	4.46 3.97 63.12 62.17
VoiceBench [8] OpenBookQA IFEval AdvBench Avg	Qwen2-Audio-chat	49.01 22.57 98.85 54.72
	Baichuan-chat	63.30 41.32 86.73 62.51
	GLM-4-Voice	52.97 24.91 88.08 57.17
	Step-Audio-chat	31.87 29.19 65.77 48.86
	Qwen2.5-Omni	79.12 53.88 99.62 72.83
	Kimi-Audio	83.52 61.10 100.00 76.93

Notably, Kimi-Audio achieves the best results on the widely-used LibriSpeech [54] benchmark, attaining error rates of 1.28 on test-clean and 2.42 on test-other, significantly outperforming models like Qwen2-Audio-base and Qwen2.5-Omni. For Mandarin ASR benchmarks, Kimi-Audio sets SOTA results on AISHELL-1 [4] (0.60) and AISHELL-2 ios [17] (2.56). Furthermore, it excels on the challenging WenetSpeech [85] dataset, achieving the lowest error rates on both test-meeting and test-net. Finally, evaluation on our internal Kimi-ASR test set confirms the model robustness. These results demonstrate the strong ASR capabilities of Kimi-Audio across various domains and languages.

6.2.2 Audio Understanding

Beyond speech recognition, we assess Kimi-Audio’s ability to comprehend diverse audio signals, including music, sound events, and speech. Table 5 summarizes the performance on various audio understanding benchmarks, where higher scores generally indicate better performance.

On the MMAU benchmark [60], Kimi-Audio demonstrates superior understanding across sound category (73.27), and speech category (60.66). Similarly, it outperforms other models on the MELD [56] speech emotion understanding task, scoring 59.13. Kimi-Audio also leads on tasks involving non-speech sound classification (VocalSound [23] and Nonspeech7k [59]) and acoustic scene classification (TUT2017 [53] and CochIScene [31]). These results highlight Kimi-Audio’s advanced capabilities in interpreting complex acoustic information beyond simple speech recognition.

6.2.3 Audio-to-Text Chat

We evaluate the ability of Kimi-Audio to engage in text conversations based on audio input using the OpenAudioBench [41] and VoiceBench benchmarks [8]. These benchmarks assess various aspects like instruction following, question answering, and reasoning. Performance metrics are

Table 7: Performance of Kimi-Audio and baseline models on speech conversation. Best results are in bold and second-best results are underlined.

Model	Speed Control	Accent Control	Emotion Control	Empathy	Style Control	Avg
GPT-4o	<u>4.21</u>	3.65	4.05	3.87	4.54	4.06
Step-Audio-chat	3.25	2.87	3.33	3.05	<u>4.14</u>	3.33
GLM-4-Voice	3.83	3.51	3.77	3.07	4.04	3.65
GPT-4o-mini	3.15	2.71	<u>4.24</u>	3.16	4.01	3.45
Kimi-Audio	4.30	<u>3.45</u>	4.27	<u>3.39</u>	4.09	<u>3.90</u>

benchmark-specific, with higher scores indicating better conversational ability. The results are presented in Table 6.

On OpenAudioBench, Kimi-Audio achieves state-of-the-art performance on several sub-tasks, including AlpacaEval, Llama Questions, and TriviaQA, and achieves highly competitive performance on Reasoning QA and Web Questions.

The VoiceBench evaluation further confirms Kimi-Audio’s strengths. It consistently outperforms all compared models on AlpacaEval (4.46), CommonEval (3.97), SD-QA (63.12), MMSU (62.17), OpenBookQA (83.52), Advbench (100.00), and IFEval (61.10). Kimi-Audio’s overall performance across these comprehensive benchmarks demonstrates its superior ability in audio-based conversation and complex reasoning tasks.

6.2.4 Speech Conversation

Finally, we assess the end-to-end speech conversation capabilities of Kimi-Audio based on subjective evaluations across multiple dimensions. As shown in Table 7, Kimi-Audio was compared against models like GPT-4o and GLM-4-Voice based on human ratings (on a 1-5 scale, higher is better).

Excluding GPT-4o, Kimi-Audio achieves the highest scores for emotion control, empathy, and speed control. While GLM-4-Voice shows slightly better accent control, Kimi-Audio achieves a strong overall average score of 3.90. This score is higher than Step-Audio-chat (3.33), GPT-4o-mini (3.45), and GLM-4-Voice (3.65), and remains a small margin with GPT-4o (4.06). Overall, the evaluation results demonstrate Kimi-Audio’s proficiency in generating expressive and controllable speech.

7 Related Work

The application of large language models (LLMs) to audio tasks has led to remarkable progress across a wide range of domains, including automatic speech recognition (ASR), audio understanding, text-to-speech synthesis (TTS), general audio generation, and speech-based human-computer interaction. These efforts explore how to bridge the gap between raw acoustic signals and linguistic reasoning by treating audio as a tokenizable sequence, enabling LLMs to process or generate audio in a language-like manner.

ASR and Audio Understanding A number of LLM-based systems have been developed to improve automatic speech recognition (ASR) and broader audio understanding tasks. Whisper [58] serves as a powerful audio encoder, and when combined with large language models (LLMs), it significantly enhances the performance of speech understanding systems. This approach has been successfully utilized in models such as Qwen-Audio [10], Qwen2-Audio [11], SALMONN [63], and OSUM [21]

These systems, however, are mostly limited to understanding tasks and do not natively support audio output.

TTS and Audio Generation For speech synthesis and general audio generation, models such as AudioLM [3], VALL-E [70], and LLaSA [80] tokenize audio via neural codecs and use decoder-only language models for autoregressive generation. Other efforts like UniAudio [77] and VoiceBox [37] extend these methods with hybrid tokenization or flow matching to improve quality and control. While these models can produce high-fidelity audio, they typically focus on generation only and lack understanding and conversation capabilities or instruction-following speech interaction.

Speech Conversation and Real-Time Dialogue Recent models have moved toward enabling real-time, end-to-end speech interaction. Moshi [14], GLM-4-Voice [84], and Mini-Omni [72] adopt interleaved or parallel decoding to support simultaneous generation of text and audio tokens, facilitating low-latency dialogue systems. OmniFlatten [86] introduces a progressive training pipeline to adapt a frozen LLM for full-duplex conversation. LLaMA-Omni [18] and Freeze-Omni [71] further refine duplex speech interaction through streaming decoders or multi-task alignment strategies. However, these systems often rely heavily on speech-only datasets and compromise language modeling quality or generality due to limited pre-training.

Toward Universal Audio-Language Foundation Models A small number of recent works aim to unify understanding and generation within a single multimodal model. Baichuan-Audio [41] uses a multi-codebook discretization to capture both semantic and acoustic features, enabling real-time interaction and strong question-answering capabilities. However, its focus on speech domain limits its broader applicability, especially for non-speech audio tasks like music or environmental sound. Step-Audio [28], on the other hand, provides a powerful solution for real-time speech interaction with a 130B-parameter unified speech-text multimodal model. While Step-Audio demonstrates strong performance, its dependency on synthetic voice data generation and the high computational costs associated with its 130B parameters pose significant barriers to accessibility and cost-effectiveness for a broader user base. Qwen2.5-Omni [74] introduces a Thinker-Talker architecture for simultaneous text and speech decoding and achieves strong benchmark results, but its design primarily emphasizes streaming inference, and it lacks an extensive pre-training phase on raw audio.

Kimi-Audio Kimi-Audio advances beyond these limitations by introducing a truly universal and open-source audio foundation model that supports speech recognition, audio understanding, audio generation, and speech conversation in a single framework. It adopts a hybrid input representation combining Whisper-derived continuous acoustic features and discrete semantic tokens (12.5Hz), ensuring rich perception and efficient modeling. The audio LLM is initialized from a text LLM, with dual-generation heads for text and audio, and a chunk-wise flow-matching detokenizer paired with BigVGAN [38] to produce expressive and low-latency speech.

Most critically, Kimi-Audio features extensive multimodal pretraining on 13 million hours of curated audio data across speech, music, and environmental sound—a scale far exceeding prior works. The pretraining tasks include audio-only, text-only, audio-to-text, and interleaved modalities, which enable the model to learn generalizable audio reasoning and maintain strong language abilities. This is followed by instruction-based fine-tuning across diverse tasks, leading to state-of-the-art results on ASR, general audio understanding, audio-text chat, and speech conversation benchmarks.

In contrast to existing models that are either limited in scope, lack pre-training, or are not publicly available, Kimi-Audio is a fully open-source, pre-trained, instruction-followable, and real-time capable model. Its comprehensive coverage, scalable architecture, and broad task alignment make it a significant step toward general-purpose audio intelligence.

8 Challenges and Future Trends

Although Kimi-Audio has achieved significant advancements in building universal audio foundation models, several challenges remain in the quest for more capable and intelligent audio processing systems. We describe the challenges and point out several exciting future directions as follows.

- **From Audio Transcription to Audio Description.** Current pre-training paradigms for audio foundation models typically leverage audio-text pre-training to bridge the gap between text and audio, where the text is obtained from audio (speech) by ASR transcription. However, text transcription focuses on the content of spoken words (what is said), neglecting important information in audio, such as paralinguistic information (e.g., emotion, style, timbre, tone), acoustic scene, and non-linguistic sounds. Thus, it is important to introduce descriptive text (i.e., audio caption) to depict the audio in richer context. Incorporating both transcriptive and descriptive text of the audio enables models to better understand and generate not only spoken language but also complex acoustic environments, paving the way for more nuanced, multimodal audio processing systems, and thus more general and versatile audio intelligence.
- **Better Audio Representations.** Current audio leverages semantic tokens or acoustic tokens as its representations. Semantic tokens are typically obtained by ASR-based auxiliary loss, which focuses on transcription-oriented information and fails to capture rich acoustic details crucial for understanding and generation. Acoustic tokens are typically learned by audio reconstruction loss, which focuses on description-oriented acoustic details and fails to capture abstractive semantic information that is crucial to bridge to text intelligence. A valuable research direction is to develop representations that integrate both transcription-oriented semantic information and description-oriented acoustic features, encompassing nuances like speaker identity, emotion, and environmental sounds while maintaining high-level abstractive information, which is paramount for more sophisticated audio understanding and generation.
- **Throw Away ASR and TTS in Audio Modeling.** Current audio foundation models rely heavily on ASR and TTS to generate training data in both the pre-training and fine-tuning stages. The quality of training data is constrained by the text recognition accuracy of ASR and expressiveness/diversity/quality of synthesized speech in TTS. In this way, the audio models behave like a sophisticated distillation of existing ASR and TTS systems. As a result, they can hardly achieve performance far beyond the ceiling of ASR/TTS and cannot achieve truly autonomous audio intelligence. An important future direction is to train audio models without relying on ASR/TTS-based pseudo audio data but relying on native audio data, which can result in much higher performance upperbound.

References

- [1] Rosana Ardila et al. “Common voice: A massively-multilingual speech corpus”. In: *arXiv preprint arXiv:1912.06670* (2019).
- [2] Beijing Academy of Artificial Intelligence (BAAI). “Infinity Instruct”. In: *arXiv preprint arXiv:2406.XXXX* (2024).

- [3] Zalan Borsos et al. “Audiolm: a language modeling approach to audio generation”. In: *IEEE/ACM transactions on audio, speech, and language processing* 31 (2023), pp. 2523–2533.
- [4] Hui Bu et al. “Aishell-1: An open-source mandarin speech corpus and a speech recognition baseline”. In: *2017 20th conference of the oriental chapter of the international coordinating committee on speech databases and speech I/O systems and assessment (O-COCOSDA)*. IEEE. 2017, pp. 1–5.
- [5] Guoguo Chen et al. “Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio”. In: *arXiv preprint arXiv:2106.06909* (2021).
- [6] Honglie Chen et al. “Vggsound: A large-scale audio-visual dataset”. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 721–725.
- [7] Qian Chen et al. “Minmo: A multimodal large language model for seamless voice interaction”. In: *arXiv preprint arXiv:2501.06282* (2025).
- [8] Yiming Chen et al. “VoiceBench: Benchmarking LLM-Based Voice Assistants”. In: *arXiv preprint arXiv:2410.17196* (2024).
- [9] Zesen Cheng et al. “Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms”. In: *arXiv preprint arXiv:2406.07476* (2024).
- [10] Yunfei Chu et al. “Qwen-audio: Advancing universal audio understanding via unified large-scale audio-language models”. In: *arXiv preprint arXiv:2311.07919* (2023).
- [11] Yunfei Chu et al. “Qwen2-audio technical report”. In: *arXiv preprint arXiv:2407.10759* (2024).
- [12] Alexis Conneau et al. “Fleurs: Few-shot learning evaluation of universal representations of speech”. In: *2022 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2023, pp. 798–805.
- [13] DeepSeek-AI. *DeepSeek-V3 Technical Report*. 2024. arXiv: 2412.19437 [cs.CL]. URL: <https://arxiv.org/abs/2412.19437>.
- [14] Alexandre Défossez et al. “Moshi: a speech-text foundation model for real-time dialogue”. In: *arXiv preprint arXiv:2410.00037* (2024).
- [15] Chandeeppa Dissanayake et al. *OpenBezoar: Small, Cost-Effective and Open Models Trained on Mixes of Instruction Data*. 2024. arXiv: 2404.12195 [cs.CL].
- [16] Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. “Clotho: An audio captioning dataset”. In: *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2020, pp. 736–740.
- [17] Jiayu Du et al. “Aishell-2: Transforming mandarin asr research into industrial scale”. In: *arXiv preprint arXiv:1808.10583* (2018).
- [18] Qingkai Fang et al. “Llama-omni: Seamless speech interaction with large language models”. In: *arXiv preprint arXiv:2409.06666* (2024).
- [19] Eduardo Fonseca et al. “Fsd50k: an open dataset of human-labeled sound events”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 30 (2021), pp. 829–852.
- [20] Zhifu Gao et al. “Paraformer: Fast and accurate parallel transformer for non-autoregressive end-to-end speech recognition”. In: *arXiv preprint arXiv:2206.08317* (2022).
- [21] Xuelong Geng et al. “OSUM: Advancing Open Speech Understanding Models with Limited Resources in Academia”. In: *arXiv preprint arXiv:2501.13306* (2025).
- [22] Sreyan Ghosh et al. “Gama: A large audio-language model with advanced audio understanding and complex reasoning abilities”. In: *arXiv preprint arXiv:2406.11768* (2024).
- [23] Yuan Gong, Jin Yu, and James Glass. “Vocalsound: A Dataset for Improving Human Vocal Sounds Recognition”. In: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2022, pp. 151–155. DOI: 10.1109/ICASSP43922.2022.9746828.
- [24] Aaron Grattafiori et al. “The llama 3 herd of models”. In: *arXiv preprint arXiv:2407.21783* (2024).
- [25] Haorui He et al. “Emilia: A Large-Scale, Extensive, Multilingual, and Diverse Dataset for Speech Generation”. In: *arXiv preprint arXiv:2501.15907* (2025).
- [26] Haorui He et al. “Emilia: An extensive, multilingual, and diverse speech dataset for large-scale speech generation”. In: *2024 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2024, pp. 885–890.
- [27] T. Heittola et al. *TAU Urban Acoustic Scenes 2022 Mobile, Development dataset*. Zenodo. Mar. 2022. DOI: 10.5281/zenodo.6337421.
- [28] Ailin Huang et al. “Step-audio: Unified understanding and generation in intelligent speech interaction”. In: *arXiv preprint arXiv:2502.11946* (2025).
- [29] Aaron Hurst et al. “Gpt-4o system card”. In: *arXiv preprint arXiv:2410.21276* (2024).

- [30] Philip Jackson and Sana ul haq. *Surrey Audio-Visual Expressed Emotion (SAVEE) database*. Apr. 2011.
- [31] Il-Young Jeong and Jeongsoo Park. “Cochlscene: Acquisition of acoustic scene data using crowdsourcing”. In: *2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE. 2022, pp. 17–21.
- [32] Zeqian Ju et al. “MoonCast: High-Quality Zero-Shot Podcast Generation”. In: *arXiv preprint arXiv:2503.14345* (2025).
- [33] Wei Kang et al. “Libriheavy: A 50,000 hours ASR corpus with punctuation casing and context”. In: *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2024, pp. 10991–10995.
- [34] Chris Dongjoo Kim et al. “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)*. 2019, pp. 119–132.
- [35] Zhifeng Kong et al. “Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities”. In: *arXiv preprint arXiv:2402.01831* (2024).
- [36] Nathan Lambert et al. “T\” ulu 3: Pushing frontiers in open language model post-training”. In: *arXiv preprint arXiv:2411.15124* (2024).
- [37] Matthew Le et al. “Voicebox: Text-guided multilingual universal speech generation at scale”. In: *Advances in neural information processing systems* 36 (2023), pp. 14005–14034.
- [38] Sang-gil Lee et al. “Bigvgan: A universal neural vocoder with large-scale training”. In: *arXiv preprint arXiv:2206.04658* (2022).
- [39] Guangyao Li et al. “Learning to answer questions in dynamic audio-visual scenarios”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 19108–19118.
- [40] Jia Li et al. “Numinamath: The largest public dataset in ai4maths with 860k pairs of competition math problems and solutions”. In: *Hugging Face repository* 13 (2024), p. 9.
- [41] Tianpeng Li et al. “Baichuan-audio: A unified framework for end-to-end speech interaction”. In: *arXiv preprint arXiv:2502.17239* (2025).
- [42] Wing Lian et al. *OpenOrca: An Open Dataset of GPT Augmented FLAN Reasoning Traces*. <https://huggingface.co/datasets/Open-Orca/OpenOrca>. 2023.
- [43] Samuel Lipping et al. “Clotho-aqa: A crowdsourced dataset for audio question answering”. In: *2022 30th European Signal Processing Conference (EUSIPCO)*. IEEE. 2022, pp. 1140–1144.
- [44] Jingyuan Liu et al. *Muon is Scalable for LLM Training*. 2025. arXiv: 2502.16982 [cs.LG]. URL: <https://arxiv.org/abs/2502.16982>.
- [45] Rui-Bo Liu et al. “Convincing Audio Generation Based on LLM and Speech Tokenization”. In: *2024 IEEE 14th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE. 2024, pp. 591–595.
- [46] Songting Liu. “Zero-shot Voice Conversion with Diffusion Transformers”. In: *arXiv preprint arXiv:2411.09943* (2024).
- [47] Steven R Livingstone and Frank A Russo. “The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English”. In: *PLoS one* 13.5 (2018), e0196391.
- [48] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv preprint arXiv:1711.05101* (2017).
- [49] Yi Luo and Jianwei Yu. “Music Source Separation With Band-Split RNN”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 31 (2023), pp. 1893–1901. DOI: 10.1109/TASLP.2023.3271145.
- [50] Linhan Ma et al. “Wenetspeech4tts: A 12,800-hour mandarin tts corpus for large speech generation model benchmark”. In: *arXiv preprint arXiv:2406.05763* (2024).
- [51] Irene Martín-Morató and Annamaria Mesaros. “What is the ground truth? reliability of multi-annotator data for audio tagging”. In: *2021 29th European Signal Processing Conference (EUSIPCO)*. IEEE. 2021, pp. 76–80.
- [52] Xinhao Mei et al. “Wavcaps: A chatgpt-assisted weakly-labelled audio captioning dataset for audio-language multimodal research”. In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* (2024).
- [53] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen. “TUT Database for Acoustic Scene Classification and Sound Event Detection”. In: *24th European Signal Processing Conference 2016 (EUSIPCO 2016)*. Budapest, Hungary, 2016.
- [54] Vassil Panayotov et al. “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.

- [55] Karol J. Piczak. “ESC: Dataset for Environmental Sound Classification”. In: *Proceedings of the 23rd Annual ACM Conference on Multimedia*. Brisbane, Australia: ACM Press, Oct. 13, 2015, pp. 1015–1018. ISBN: 978-1-4503-3459-4. DOI: 10.1145/2733373.2806390. URL: <http://dl.acm.org/citation.cfm?doid=2733373.2806390>.
- [56] Soujanya Poria et al. “Meld: A multimodal multi-party dataset for emotion recognition in conversations”. In: *arXiv preprint arXiv:1810.02508* (2018).
- [57] Vineel Pratap et al. “MLS: A Large-Scale Multilingual Dataset for Speech Research”. In: *ArXiv abs/2012.03411* (2020).
- [58] Alec Radford et al. “Robust speech recognition via large-scale weak supervision”. In: *International conference on machine learning*. PMLR. 2023, pp. 28492–28518.
- [59] Muhammad Mamunur Rashid, Guiqing Li, and Chengrui Du. “Nonspeech7k dataset: Classification and analysis of human non-speech sound”. In: *IET Signal Processing* 17.6 (2023), e12233.
- [60] S Sakshi et al. “Mmau: A massive multi-task audio understanding and reasoning benchmark”. In: *arXiv preprint arXiv:2410.19168* (2024).
- [61] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello. “A dataset and taxonomy for urban sound research”. In: *Proceedings of the 22nd ACM international conference on Multimedia*. 2014, pp. 1041–1044.
- [62] Yao Shi et al. “Aishell-3: A multi-speaker mandarin tts corpus and the baselines”. In: *arXiv preprint arXiv:2010.11567* (2020).
- [63] Changli Tang et al. “Salmonn: Towards generic hearing abilities for large language models”. In: *arXiv preprint arXiv:2310.13289* (2023).
- [64] Zhiyuan Tang et al. “Kespeech: An open source speech dataset of mandarin and its eight subdialects”. In: *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*. 2021.
- [65] Kimi Team et al. *Kimi k1.5: Scaling Reinforcement Learning with LLMs*. 2025. arXiv: 2501.12599 [cs.AI]. URL: <https://arxiv.org/abs/2501.12599>.
- [66] Teknium. *OpenHermes 2.5: An Open Dataset of Synthetic Data for Generalist LLM Assistants*. 2023. URL: <https://huggingface.co/datasets/teknium/OpenHermes-2.5>.
- [67] Migel Tissera. *Synthia-70b-v1.2: Synthetic intelligent agent*. Hugging Face. 2023. URL: <https://huggingface.co/migtissera/Synthia-13B>.
- [68] Samarth Tripathi, Sarthak Tripathi, and Homayoon Beigi. “Multi-modal emotion recognition on iemocap dataset using deep learning”. In: *arXiv preprint arXiv:1804.05788* (2018).
- [69] Changhan Wang et al. “VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation”. In: *arXiv preprint arXiv:2101.00390* (2021).
- [70] Chengyi Wang et al. “Neural codec language models are zero-shot text to speech synthesizers”. In: *arXiv preprint arXiv:2301.02111* (2023).
- [71] Xiong Wang et al. “Freeze-omni: A smart and low latency speech-to-speech dialogue model with frozen llm”. In: *arXiv preprint arXiv:2411.00774* (2024).
- [72] Zhifei Xie and Changqiao Wu. “Mini-omni: Language models can hear, talk while thinking in streaming”. In: *arXiv preprint arXiv:2408.16725* (2024).
- [73] Jin Xu et al. “Qwen2. 5-omni technical report”. In: *arXiv preprint arXiv:2503.20215* (2025).
- [74] Jin Xu et al. *Qwen2.5-Omni Technical Report*. 2025. arXiv: 2503.20215 [cs.CL]. URL: <https://arxiv.org/abs/2503.20215>.
- [75] Zhangchen Xu et al. “Magpie: Alignment data synthesis from scratch by prompting aligned llms with nothing”. In: *arXiv preprint arXiv:2406.08464* (2024).
- [76] An Yang et al. “Qwen2.5 Technical Report”. In: *arXiv preprint arXiv:2412.15115* (2024).
- [77] Dongchao Yang et al. “Uniaudio: An audio foundation model toward universal audio generation”. In: *arXiv preprint arXiv:2310.00704* (2023).
- [78] Pinci Yang et al. “Avqa: A dataset for audio-visual question answering on videos”. In: *Proceedings of the 30th ACM international conference on multimedia*. 2022, pp. 3480–3491.
- [79] Zehui Yang et al. “Open source magicdata-ramc: A rich annotated mandarin conversational (ramc) speech dataset”. In: *arXiv preprint arXiv:2203.16844* (2022).
- [80] Zhen Ye et al. “Llasa: Scaling Train-Time and Inference-Time Compute for Llama-based Speech Synthesis”. In: *arXiv preprint arXiv:2502.04128* (2025).

- [81] Jianwei Yu et al. “Autoprep: An automatic preprocessing framework for in-the-wild speech data”. In: *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2024, pp. 1136–1140.
- [82] Jianwei Yu et al. “High fidelity speech enhancement with band-split rnn”. In: *arXiv preprint arXiv:2212.00406* (2022).
- [83] Heiga Zen et al. “Libritts: A corpus derived from librispeech for text-to-speech”. In: *arXiv preprint arXiv:1904.02882* (2019).
- [84] Aohan Zeng et al. “Glm-4-voice: Towards intelligent and human-like end-to-end spoken chatbot”. In: *arXiv preprint arXiv:2412.02612* (2024).
- [85] Binbin Zhang et al. “Wenetspeech: A 10000+ hours multi-domain mandarin corpus for speech recognition”. In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2022, pp. 6182–6186.
- [86] Qinglin Zhang et al. “Omniflatten: An end-to-end gpt model for seamless voice conversation”. In: *arXiv preprint arXiv:2410.17799* (2024).
- [87] Yu Zhang et al. “Google usm: Scaling automatic speech recognition beyond 100 languages”. In: *arXiv preprint arXiv:2303.01037* (2023).
- [88] Hang Zhao et al. “MINT: Boosting Audio-Language Model via Multi-Target Pre-Training and Instruction Tuning”. In: *Interspeech 2024*. 2024, pp. 52–56. DOI: 10.21437/Interspeech.2024-1863.
- [89] Kun Zhou et al. “Emotional voice conversion: Theory, databases and ESD”. In: *Speech Communication* 137 (2022), pp. 1–18.

Appendix

A Contributions

Core Contributors

Ding Ding
Zeqian Ju
Yichong Leng
Songxiang Liu
Tong Liu
Zeyu Shang
Kai Shen
Wei Song
Xu Tan[#]
Heyi Tang
Zhengtao Wang
Chu Wei
Yifei Xin
Xinran Xu
Jianwei Yu
Yutao Zhang
Xinyu Zhou[#]

Contributors

Y. Charles
Jun Chen
Yanru Chen
Yulun Du
Weiran He
Zhenxing Hu
Guokun Lai
Qingcheng Li
Yangyang Liu
Weidong Sun
Jianzhou Wang
Yuzhi Wang
Yuefeng Wu
Yuxin Wu
Dongchao Yang
Hao Yang
Ying Yang
Zhilin Yang
Aoxiong Yin
Ruibin Yuan
Yutong Zhang
Zaida Zhou

[#] Project leads.

The contributor list is in alphabetical order based on their last names.