Adapter Based Models - Non Streaming and Non Duplex

Qwen2.5-Omni Technical Report

🤗 Qwen (Qwen)

ModelScope 魔搭社区

GitHub - QwenLM/Qwen2.5-Omni: Qwen2.5-Omni is an end-to-end multimodal model by Qwen team at Alibaba Cloud, capable of understanding text, audio, vision, video, and performing real-time speech generation.

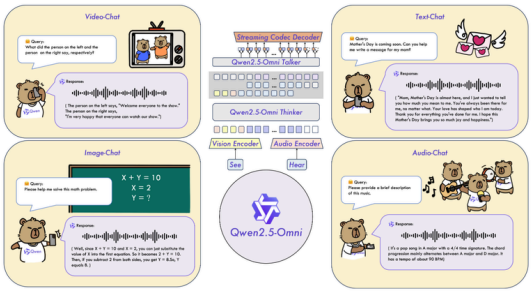


Figure 1: Qwen2.5-Omni is a unified end-to-end model capable of processing multiple modalities, such as text, audio, image and video, and generating real-time text or speech response. Based on these features, Qwen2.5-Omni supports a wide range of tasks, including but not limited to voice dialogue, video dialogue, and video reasoning.

1. Model Architecture

Qwen2.5-Omni is a unified multimodal model built to understand and generate across text, images, audio, video, and speech. Key architectural components include:

- **TMRoPE (Temporal Multimodal Rotary Position Embedding)**: A 3D positional encoding system that handles time, height, and width dimensions across modalities, with dynamic temporal adjustment for variable frame rates.
- **Thinker-Talker Framework**: A specialized architecture that separates "thinking" (reasoning about content) from "talking" (speech generation) to enable real-time speech generation while minimizing cross-modal interference.
- **Modality-Specific Encoders**: Specialized components for handling different input modalities that feed into a unified transformer backbone.
- **Block-wise Processing**: For handling audio and video efficiently through chunked processing.
- **Sliding Window Mechanism**: For code-to-wav generation to maintain real-time capabilities.

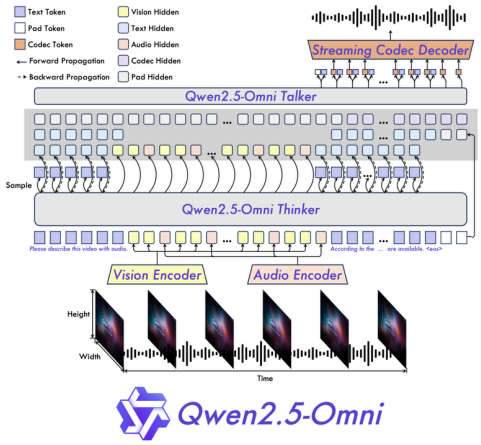


Figure 2: The overview of Qwen2.5-Omni. Qwen2.5-Omni adpots the Thinker-Talker architecture. Thinker is tasked with text generation while Talker focuses on generating streaming speech tokens by receives high-level representations directly from Thinker.

2. Training Process

The training occurred in several phases:

1. **Pre-training**: On a large multimodal dataset to build the foundation model.
2. **In-Context Learning (ICL)**: For zero-shot speech generation capabilities.
3. **Reinforcement Learning (RL)**: To optimize speech generation stability, reducing issues like attention misalignment and pronunciation errors.
4. **Speaker Fine-tuning**: To capture specific voice characteristics for single-speaker models.

3. Data Used

While the paper doesn't fully detail all datasets, it mentions:

- **SEED Dataset** (Anastassiou et al., 2024): Used for evaluating speech generation capabilities.
- **Self-created Dataset**: For subjective naturalness evaluation of speech generation.
- **Multimodal Training Data**: Including text, images, audio, and video, though specific details about the creation process aren't provided in the excerpts.
- The paper mentions they identified issues in tasks like "video OCR and audio-video collaborative understanding," suggesting they created evaluation datasets for these complex multimodal scenarios.

4. Novel Contributions and Approaches

Validated Approaches:

- TMRoPE position encoding for temporal alignment of audio and video
- Thinker-Talker framework for real-time speech
- Time-interleaving method for processing video with audio
- Block-wise encoding for efficiency

Key Innovations:

- Dynamic adjustment of temporal IDs for variable frame rates
- Position numbering continuity across modalities
- Real-time speech generation while preserving emotional context

5. Inference in Real-World Use Cases

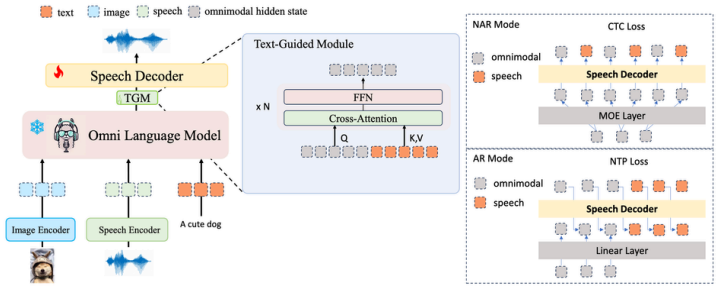
The model shows strong capabilities for:

- Following voice commands
- Zero-shot speech generation with high content consistency
- Single-speaker voice modeling with preserved prosody
- Complex audio-visual interactions
- Emotional context preservation in speech dialogues
- Multimodal reasoning and understanding

OpenOmni: Advancing Open-Source Omnimodal Large Language Models with Progressive Multimodal Alignment and Real-Time Self-Aware Emotional Speech Synthesis. February 23, 2025

Luo, Run, Ting-En Lin, Haonan Zhang, Yuchuan Wu, Xiong Liu, Min Yang, Yongbin Li, et al. “OpenOmni: Advancing Open-Source Omnimodal Large Language Models with Progressive Multimodal Alignment and Real-Time Self-Aware Emotional Speech Synthesis.” arXiv, February 23, 2025. OpenOmni: Advancing Open-Source Omnimodal Large Language Models...

Data - **Code/Model**



Model Architecture:

- LLM: Qwen2.5-7B-Instruct
- Image Encoder: CLIP-ViT-L
- Speech Encoder: Whisper-large-v3
- Speech Decoder: Qwen2.5-0.5B-Instruct
- Audio Tokenizer
  - For the autoregressive mode, GLM4Voice tokenizer, 16k Vocab
  - For non-autoregressive models, we use the CosVoice Tokenizer, 6k vocab

Speech Generation:

- The LLM's hidden states [h1, ..., hN] are passed to the TGM
- This produces transformed hidden states [c1, ..., cN]
- These are fed into the speech decoder layers to get final hidden states [o1, ..., oM] (Based on AR or NAR mode), NAR is fast but AR is good quality
- The discrete units are converted to waveforms by the vocoder

Training and Data

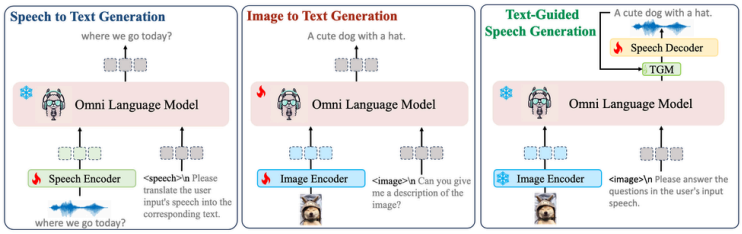


Figure 2: **Progressive Training process of OpenOmni.** To enable zero-shot omnimodal learning and real-time emotional speech generation, OpenOmni undergoes a progressive three-stage training process: (1) **Speech-Text Generation**: A speech encoder extracts continuous speech and text features for alignment learning, equipping the large language model with speech understanding capabilities. (2) **Image-Text Generation**: An image encoder extracts continuous image and text features, facilitating alignment learning that enhances OpenOmni's image comprehension and instruction-following abilities. This process also establishes implicit omnimodal alignment, enabling omni-understanding. (3) **Speech Generation**: A lightweight speech decoder is trained using high-quality synthesized speech dialogue data, with a focus on direct preference optimization for emotional speech. This final stage allows OpenOmni to generate real-time, self-aware emotional speech. A text-guided module (TGM) is utilized to accelerate the training convergence.

Hyperparameter	I	II	III	IV	V
batch size	256	128	128	32	32
lr	$1e^{-3}$	$1e^{-3}$	$5e^{-5}$	$5e^{-4}$	$5e^{-4}$
warmup ratio	0.3	0.3	0.3	0.3	0.3
epoch	1	1	1	3	3
freeze LLM	✔	✔	✘	✔	✔
optimizer	AdamW	AdamW	AdamW	AdamW	AdamW
cost	40 GPU·H	80 GPU·H	500 GPU·H	36 GPU·H	8 GPU·H
dataset	1-1	2-1	2-2	3-1	3-2
loss	$\mathcal{L}_{s2t}$	$\mathcal{L}_{i2t}$	$\mathcal{L}^{I}_{i2t}$	$\mathcal{L}_{ctc}$	$\mathcal{L}_{dpo}$

Table 7: The detailed training setup for OpenOmni and the hyper-parameters across the training stage. All experiments are conducted in 8xA100 setting.

I → Speech-to-text Generation

1. Data: OpenOmni-1-1
   - Made up of WeNetSpeech, LibriSpeech, AIShell-4 and Shorter responses from O2S-300K
   - MMEvol text are translated in Chinese to create Audio-text pairs
   - Total: approximately 8000 hours of bilingual speech-text data
2. Training Process:

II → Image-Text Pretraining

1. Data: OpenOmni-2-1
   - LLaVA-Pretrain-595K dataset

III → Image-Text Instruction Tuning

1. Data Used: OpenOmni-2-2
   - MMEvol dataset: 1.7M

IV → Real-time Speech Generation

1. Data: OpenOmni-3-1
   - O2S-300K: A curated dataset containing:
     - 100K English QA pairs with long responses from MMEvol and UltraChat
     - 50K Chinese translations of these QA pairs
     - 200K text-speech pairs from UltraChat
     - Total: 8000 hours of high-quality bilingual speech

V → Self-aware Emotional Speech Generation

1. Data: OpenOmni-3-2
   - EO2S-9K: A preference dataset containing:
     - 1K bilingual dialogues with emotion labels based on the Plutchik Model
     - Augmented data for underrepresented emotions (like anger and sadness)
     - Preference pairs with emotional vs. neutral speech samples
     - Equal representation of Chinese and English

2. Training Process:
    - Leverages preference pairs of emotionally congruent vs. neutral responses
    - Uses DPO rather than RLHF

Soundwave: Less Is More for Speech-Text Alignment in LLMs. February 18, 2025

Zhang, Yuhao, Zhiheng Liu, Fan Bu, Ruiyu Zhang, Benyou Wang, and Haizhou Li. "Soundwave: Less Is More for Speech-Text Alignment in LLMs." arXiv, February 18, 2025.
Soundwave: Less is More for Speech-Text Alignment in LLMs .

Code

Table 1: The parameters of different modules. The orange represents the number of training parameters.

Modules	#Param.	Training stage	Details
Audio encoder	∼635M	-	Whisper Large V3
Alignment adapter	∼144M	I&II	One projection and Transformer layer
Shrinking adapter	∼67M	II	One cross-attention and layer-norm
LLMs	∼8B	-	Llama3.1
LLM adapter	∼55M	II&III	LoRA
Total	∼9B		

Model Architecture Details

Base Components:

- **Audio Encoder**: Whisper Large V3 (pretrained, frozen during training)
- **Foundation LLM**: Llama-3.1-8B-Instruct
- **Custom Adapters**:
    - Alignment adapter: Projection layer with output size 4096 + Transformer layer
    - Shrinking adapter: CTC-based dynamic sequence compression
- **Fine-tuning**: LoRA applied to Attention module (rank=64, α=16)

Complete Training Process

Stage I: Alignment (6,000 steps)

- **Purpose**: Close the representation space gap between speech and text
- **Method**:
    - Uses auxiliary CTC loss to align without requiring LLM involvement
    - Transforms audio features to match LLM dimensionality through the alignment adapter
- **Data**:
    - Verified ASR data with WER < 10% (filtered using Whisper medium)
    - Standardized sound data (8,000 manually annotated sound categories + 20,000 curated samples)
- **Processing**:
    - Applied SpecAugment for speech robustness
    - Standardized sounds to 3-second durations
    - Speaker identification formatting ("The first speaker says...")

Stage II: Shrinking (6,000 steps)

- **Purpose**: Reduce speech sequence length while preserving information
- **Method**:
    - Dynamic shrinking using CTC probabilities to determine final length
    - Cross-attention mechanism to retain lossless information
    - Temperature-based data sampling to balance task distribution
- **Data**:
    - Same speech/sound data as Stage I
    - Added text data (Wizard SFT dataset) to maintain language understanding
- **Temperature Sampling**:
    - Started at T=1 (sample-level uniform distribution)
    - Increased by 5 per epoch (moving toward task-level uniform distribution)

Stage III: SFT (4,000 steps)

- **Purpose**: Enable generalization across diverse tasks
- **Method**:
    - Fine-tuned only LoRA parameters (speech and text already aligned)
    - Used both text and speech-based instructions

- Implemented Chain of Thought (CoT) approach for complex tasks

- **Data**:
  - Text-based instructions in three QA formats (direct answer, multiple-choice, natural format)
  - Speech instructions from AnyInstruct speech subset
  - CoT datasets for complex tasks (transcribe first, then respond)

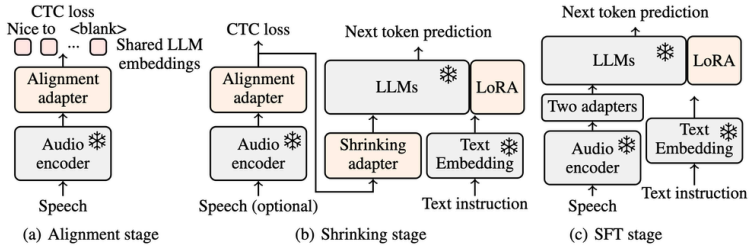


Figure 2: Training progress of Soundwave. The gray modules are frozen while the orange modules are updated.

Table 3: Summary of datasets used in different stages and their total hours.

Dataset	I	II	III	Num.	Hours	Task
GigaSpeech (M) (Chen et al., 2021)	✓	✓		713k	805.11	ASR
TED-LIUM (Hernandez et al., 2018)	✓	✓		144k	244.02	ASR
Multilingual Librispeech (En) (Pratap et al., 2020)	✓	✓		985k	4,081.61	ASR
Europarl-ASR (Garcés Díaz-Munío et al., 2021)	✓	✓		719k	418.42	ASR
TextrolSpeech (Ji et al., 2024b)	✓	✓	✓	215k	301.19	ASR, GR ¦ Emotion Recognition
LibriSpeech (Panayotov et al., 2015)	✓	✓	✓	281k	961.05	ASR, Speech Grounding
MUST-C (En-De) (Cattoni et al., 2021)		✓	✓	283k	388.55	Speech Translation
Common Voice (En) (Ardila et al., 2019)		✓	✓	233k	364.64	AP ¦ Speech Translation
Fisher (Cieri et al., 2004)		✓	✓	132k	1,091.42	ASR, Chat
Europarl-ST (Iranzo-Sánchez et al., 2020)			✓	53k	133.16	Language Identification
Common Voice (Ja) (Ardila et al., 2019)			✓	13k	15.00	Language Identification
SLURP (Bastianelli et al., 2020)			✓	141k	101.49	IC ¦ Entity Recognition
CREMA-D (Cao et al., 2014)			✓	7k	5.26	Emotion Recognition
RAVDESS (Livingstone & Russo, 2018)			✓	1k	1.48	Emotion Recognition
IEMOCAP (Busso et al., 2008)			✓	3k	2.16	Emotion Recognition
MELD (Poria et al., 2019)			✓	9k	8.12	Emotion Recognition
VoxCeleb (Nagrani et al., 2017)			✓	156k	435.17	Speaker Num. Verification
FoR (Reimao & Tzerpos, 2019)			✓	54k	47.55	Synthesized Detection
AnyInstruct (Zhan et al., 2024)			✓	107k	206.30	Speech Instruction
VocalSound (Gong et al., 2022)	✓	✓	✓	20k	23.20	Sound Classification
TUT2017 (Duppada & Hiray, 2017)	✓	✓	✓	5k	13.00	Scene Classification
CochlScene (Jeong & Park, 2022)			✓	75k	208.65	Scene Classification
Total [2]				**4,349k***	**9,856.91***	**15**

[1] GR is for Gender Recognition, AP is for Age Prediction, and IC is for Intent Classification.
[2] '*' means that this table is compiled from the perspective of audio, and an audio file may be used multiple times for different tasks. If multiple usages at different tasks are all counted, the number of data samples is 6301k, and the total duration is 14068.77 hours.

Total Training Resources

- **Hardware**: 32 A800 GPUs
- **Time**: ~5 days total (4 days for Stages I & II, 1 day for SFT)
- **Data Volume**: Only 10,000 hours (~1/50th of comparable models)
- **Speech/Sound Ratio**: 98.61% English speech, ~244 hours of sound data

Novel Contributions & Validated Approaches

1. Two-Adapter Architecture: Successfully validated that separating alignment and shrinking into distinct adapters significantly improves training efficiency and performance.
2. Data Quality Over Quantity: Experimentally proved that carefully curated data (10k hours) outperforms much larger datasets (500k hours in competing models).
   - Fig. 7 shows unstable training with uncleaned speech and significantly worse performance with unprocessed sound data
3. Dynamic Shrinking Method: Demonstrated superior compression (2.5% ratio) while maintaining performance, unlike static approaches.
   - Proved that auxiliary information retention is critical for performance during compression
4. Temperature-Based Sampling: Successfully validated this approach for balanced multi-task learning without requiring extensive prior knowledge.
5. Chain of Thought for Speech: Demonstrated that breaking complex tasks into smaller steps (transcribe first, then process) significantly improves performance on limited training data.
6. Single-Encoder Architecture: Proved that a single frozen encoder can effectively process both speech and sound, contradicting the need for specialized sound encoders in previous works.
7. Data Scaling Efficiency: Validated remarkably strong performance even with just 1,000 hours of data (Fig. 8), challenging the assumption that massive datasets are required.
8. Knowledge Retention: Demonstrated that the alignment approach preserves the LLM's knowledge even after fine-tuning for speech understanding.

Rejected Alternatives (Based on Experimental Evidence)

1. Direct Speech-Text Integration: Experiments showed poor convergence without the initial alignment stage (Fig. 5).
2. Static Downsampling: Demonstrated inferior performance compared to dynamic compression.
3. Two-Encoder Approach: Results showed the single-encoder architecture outperforms two-encoder models for processing both speech and sound.

4. Unbalanced Dataset Training: Temperature sampling proved essential, as naive mixed training leads to domination by majority tasks.

Inference in Real-World Use Cases

1. **Speech Understanding Flow**:
   - Audio input → Whisper encoder → Alignment adapter → Shrinking adapter → Llama LLM → Response
2. **Inference Speed Advantages**:
   - 15-25% faster than competing adapter methods (Tab. 8)
   - Reduced Time To First Token (TTFT) due to efficient sequence compression
3. **Real-World Applications**:
   - **Multilingual Translation**: Strong zero-shot capabilities for cross-language understanding
   - **Content Understanding**: Superior performance in speech emotion recognition and context comprehension
   - **Knowledge-Based QA**: Retains LLM knowledge capabilities for complex questions asked via speech (Fig. 9)
   - **Sound Event Recognition**: Can identify environmental sounds embedded within conversations
4. **Limitations in Production**:
   - Still underperforms on ASR compared to specialized models (though competitive)
   - Limited language identification capabilities due to English-dominant training
   - Sound classification still has room for improvement with only 244 hours of sound data

The model represents a significant advancement in efficient speech LLM training, proving that high-quality, focused data and clever architecture design can dramatically reduce resource requirements while maintaining or exceeding performance.

Open Source

- Model
- Evaluation code
- Data

Step-Audio: Unified Understanding and Generation in Intelligent Speech Interaction. February 17, 2025

Huang, Ailin, Boyong Wu, Bruce Wang, Chao Yan, Chen Hu, Chengli Feng, Fei Tian, et al. "Step-Audio: Unified Understanding and Generation in Intelligent Speech Interaction." arXiv, February 17, 2025. Step-Audio: Unified Understanding and Generation in Intelligent... .

Open Source - Step-Audio, TTS, Eval Benchmark, Tokenizer

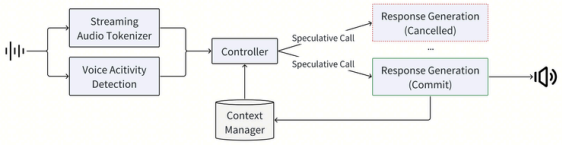


Figure 3: The architecture of the real - time inference pipeline aims to enable real-time interactions. When audio is input, it's processed concurrently by the streaming audio tokenizer and the voice activity detection module. The controller manages state transitions. A pause in user speech triggers speculative response generation, with multiple calls made but only one response committed. The context manager handles the conversation history in text format for continuity. Once the user finishes speaking, the system enters the reply state, commits a speculative response, and outputs audio. After that, it returns to the idle state for the next interaction.

Model Details and Architecture

- **Core Model**: 130B-parameter unified speech-text multi-modal model (Step-Audio-Chat version open-sourced)
- **Architecture**: AQTA (Audio Question Text Answer) + TTS framework for real-time voice dialogue
- **Component Structure**:
  - Dual-codebook tokenization framework with linguistic (16.7Hz, 1024-codebook) and semantic (25Hz, 4096-codebook) tokenizers in 2:3 temporal interleaving
  - 130B-parameter LLM based on Step-1, enhanced through audio-contextualized training
  - 3B-parameter speech decoder consisting of a language model, flow-matching model, and mel-to-wave vocoder
  - Voice Activity Detection (VAD) module

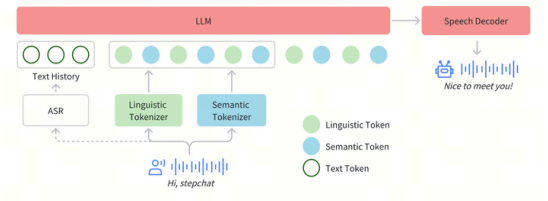


Figure 2: Architecture of Step-Audio. Step-Audio primarily consists of three components: the speech tokenizer, the LLM, and the speech decoder. The speech tokenizer is responsible for discretizing the input speech into tokens. The LLM models both text and speech tokens, while the speech decoder generates the waveform output.

Training

Data:

1. Audio:
   - 1.1T tokens of audio continuation data (730K hours)
   - 113B tokens of TTS synthesized speech data (about 70K hours)
   - 105B tokens of ASR data (around 65K hours)
   - 350B tokens of audio-text alternating data (approximately 200K hours).
2. Text: 800B tokens.
3. Image: 800B tokens of image-text paired/alternating data

Pretraining (Step-Omni)

Stage 1:

- Expanded vocabulary with 5,120 audio tokens and integrated image encoder to form Step-Omni
- Used differential learning rates: backbone (2e-5), embedding and LM head (5x higher)
- Image encoder remained frozen
- Used 2:1:1 ratio of audio, text, and image data (audio data = pure audio continuation tasks)
- Trained on 1.2T tokens

Stage 2:

- Incorporated audio-text interleaved data at 1:1 ratio with audio continuation
- Maintained 2:1:1 ratio of modalities
- Trained on 800B tokens

Stage 3:

- Added ASR and TTS data in 1:1:1:1 ratio with previous data types
- Adjusted modality ratio to 4:3:3 (audio, text, image)
- Synchronized learning rates with cosine schedule (2e-5 to 5e-6)
- Total of 3.3T tokens across all stages

Text-to-Speech (TTS) Post-Training

- Used synthetic data-driven framework with 3 components:
  a. Step-2 LLM to generate diverse textual content
  b. Pre-trained Step-Audio model with audio-token cooldown for speaker-specific generation
  c. Audio-Edit Model for emotional expressions and speaking styles
- **Data Creation**:
  a. Language/Dialect: Generated through continuation by translating chat text
  b. Emotions/Styles: Used comparative pair format and iterative enhancement
  c. Singing/RAP: Built from 10,000+ hours of tracks with extensive cleaning (17.8% retained)
  d. Target Speaker: Combined native speaker codes with target speaker prompts
  e. Quality Assessment: Used ASR accuracy, VAD performance, speaker diarization, emotion recognition, and DNS effectiveness
- **Training**:
  a. 3B-parameter model with two-turn dialogue configuration
  b. Training duration: One epoch
  c. Learning rate: 2×10^-5 with cosine decay to 2×10^-6
  d. Used descriptive and comparative instruction tags

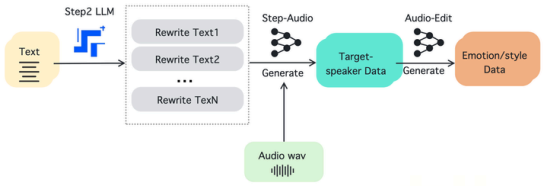


Figure 5: The process starts with text input which is processed by a Step-2 LLM to generate multiple rewritten texts. Then, a Step-Audio model generates target-speaker data using the rewritten texts and existing audio wav data. Finally, an Audio-Edit model refines the data to produce emotion/style data, addressing the scarcity of high - quality speech data in TTS tasks.

Audio Question Text Answer (AQTA) Post-Training

- **SFT Data Types**: TQTA, AQTA, TAQTA, and others (AQAA, VAQTA)
- **SFT Training**: 1 epoch with learning rate from 5.656×10^-5 to 5.656×10^-6
- **Reward Model Training**:
  - Two-stage approach: TQTA pretraining followed by AQTA fine-tuning
  - Data: Human preference data on instruction following, conversational naturalness, and safety
  - Learning rate: 1.24×10^-5 with cosine decay to 6×10^-6
  - Achieved 70.51% pair-wise accuracy on the test set
- **PPO Training**:
  - Critic model warmed up with 80 initial training steps
  - PPO clip threshold ε = 0.2
  - Initial learning rate of 1×10^-6 with cosine decay to 2×10^-7
  - KL penalty coefficient β = 0.05

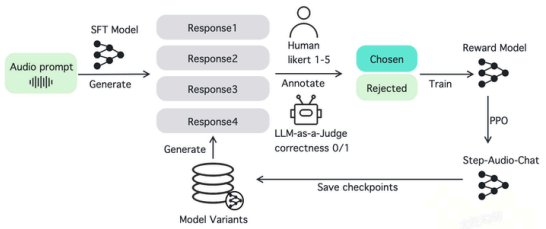


Figure 6: At each training iteration, we collect multiple responses from different versions of the model. Then, through manual scoring, as well as evaluated by a LLM, high-quality pairs are selected to train the reward model. Finally, we use PPO algorithm to train the final Step-Audio-Chat model.

Novel Contributions and Validated Approaches

1. **Dual-Codebook Tokenization**:
   - Validated: Improved Character Error Rate from 25.5 to 18.4 compared to single semantic codebook
   - Demonstrated lower perplexity for both token types while maintaining speaker similarity
   - Enhanced multi-turn instruction following and pronunciation clarity
2. **Generative Data Engine**:
   - Novel approach: Using LLM to generate content then using audio models to create synthetic training data
   - Validated: Step-Audio-TTS-3B achieved state-of-the-art results in CER and WER
   - Solved the problem of scarce emotional, dialect, and singing voice data
3. **Speculative Response Generation**:
   - Novel approach: Preemptively generating responses during user pauses
   - Validated: Successfully committed 40% of speculative responses
   - Reduced per-response latency by approximately 500ms
4. **Addressing "Deaf Hacking"**:
   - Identified pattern bias in reward model's training data
   - Novel solution: Constructed specific rejected responses exhibiting hacking behavior
   - Improved RLHF quality by avoiding responses that default to "I didn't hear clearly"
5. **Disaggregated Data Processing**:
   - Introduced StarWeaver: RPC-based distributed data processing library
   - Relocated CPU-intensive pre-processing to remote processes
   - Enhanced training efficiency with better load balancing
6. **Disaggregated Model Placement**:
   - Allocated dedicated resources with tailored parallelism for each sub-model
   - Minimized pipeline bubbles caused by model heterogeneity

Real-Time Inference Process

The real-time inference pipeline operates through a state machine:

1. **Silence State**: System waits for user input
2. **UserSpeaking State** (when VAD detects speech):
   - Streaming Audio Tokenizer converts audio to tokens in real-time
   - Parallel tokenizer pipelines with fixed-duration segmentation
   - Tokens merged with 2:3 interleaving ratio
3. **UserPaused State** (when user temporarily pauses):
   - Speculative response generation triggered
   - System preemptively generates possible responses
4. **BotReplying State** (when user finishes speaking):
   - Most recent speculative response committed or new response generated
   - Speech decoder generates audio output
   - Context Manager maintains history as text (not raw audio) for efficiency
   - System can handle interruptions by prioritizing new input
5. **Tool Call Handling**:
   - Asynchronous tool invocation that maintains voice interaction
   - Parallel execution of external service queries and speech synthesis
   - Eliminates waiting time for audio rendering during tool calls

This process creates a fluid, low-latency conversation experience with approximately 40% of speculative responses being successfully used, reducing response latency by 500ms compared to non-speculative methods.

DuplexMamba: Enhancing Real-Time Speech Conversations with Duplex and Streaming Capabilities. February 16, 2025

Lu, Xiangyu, Wang Xu, Haoyu Wang, Hongyun Zhou, Haiyan Zhao, Conghui Zhu, Tiejun Zhao, and Muyun Yang. "DuplexMamba: Enhancing Real-Time Speech Conversations with Duplex and Streaming Capabilities." arXiv, February 16, 2025. DuplexMamba: Enhancing Real-time Speech Conversations with Duplex... .

[Code]

Datasets

1. For the raw data of Stage 1 and Stage 2, you can download LibriSpeech, TED-LIUM, mls_eng_10k, and VoiceAssistant-400K.
2. The state discrimination dataset we used can be accessed here.
3. For the preprocessed data for Stage 3 and Stage 4, you can download it from here.

Model Architecture Details

1. **Speech Encoder (ConMamba)**:
   - Based on ConMamba (Jiang et al., 2024)
   - CNN frontend for compressing spectrograms into tokens
   - Multiple stacked ConMamba blocks
   - Bidirectional Mamba processing for capturing both past and future context
   - Fixed-size state representation
2. **Speech Adapter**:
   - Downsampling mechanism: Concatenates every k consecutive frames along feature dimension
   - Two-layer perceptron with ReLU activation between layers
   - Maps speech representations to language model embedding space
3. **Language Model**:
   - Mamba-based language model (2.8B parameters)
   - Uses selective state space model (SSM) instead of attention
   - Linear computational complexity relative to sequence length
   - Fixed-size contextual memory

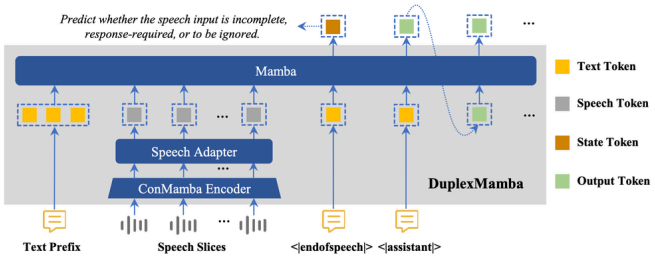


Figure 1: The model architecture of DuplexMamba.

Complete Training Process

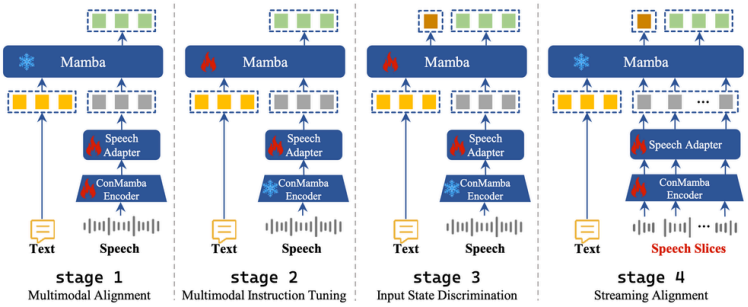


Figure 2: The four-stage training of DuplexMamba.

Stage 1: Multimodal Alignment

- **Purpose**: Align speech encoder with language model representation space
- **Data**: ~11,000 hours from LibriSpeech, TEDLIUM 3, and Multilingual LibriSpeech
- **Text Preprocessing**: All text normalized to lowercase English without punctuation
- **Training Focus**: Only ConMamba encoder and speech adapter parameters
- **Loss**: Cross-entropy on transcription prediction
- **Language Model**: Kept frozen

Stage 2: Multimodal Instruction Tuning

- **Purpose**: Enhance instruction-following capability across multimodal contexts
- **Data**:
  - 50,000 ASR samples from Multilingual LibriSpeech
  - ~200,000 QA samples from VoiceAssistant dataset (filtered to remove identity data)
  - Only first-round instructions from multi-turn conversations retained
- **Tasks**: ASR and speech-to-text QA tasks
- **Training Focus**: Only Mamba language model and speech adapter
- **Loss**: Cross-entropy on response generation
- **Speech Encoder**: Kept frozen

Stage 3: Input State Discrimination

- **Purpose**: Enable the model to identify input states for duplex processing
- **Data**:
  - 5,000 ASR samples from Multilingual LibriSpeech
  - Custom state discrimination dataset with three categories:
    i. Response-required samples from VoiceAssistant
    ii. Incomplete samples (truncated audio with placeholder responses)
    iii. Ignored samples from GigaSpeech L (podcasts, audiobooks, YouTube)
- **State Tokens**:
  - `<response>`: Input complete, needs response
  - `<incomplete>`: Input not yet complete
  - `<ignore>`: Input complete but should be ignored
- **Training Focus**: Only Mamba language model and speech adapter
- **Loss**: Combined loss of state token prediction and response generation
- **Speech Encoder**: Kept frozen

Stage 4: Streaming Alignment

- **Purpose**: Enable real-time processing of speech slices
- **Data**: Same as Stage 3, but with audio sliced into 3-second intervals
- **Training Focus**: Only ConMamba encoder and speech adapter
- **Loss**: Same as Stage 3
- **Language Model**: Kept frozen

Stage	Task	Dataset	Items
1	ASR	LibriSpeech	960h
		TED-LIUM 3	450h
		Multilingual LibriSpeech	10000h
2	ASR	Multilingual LibriSpeech	50k
	QA	VoiceAssistant	200k
3,4	ASR	Multilingual LibriSpeech	5k
	QA	Response-Required Data	10k
		Incomplete Data	5k
		Ingored Data	5k

Table 1: Training data for each stage.

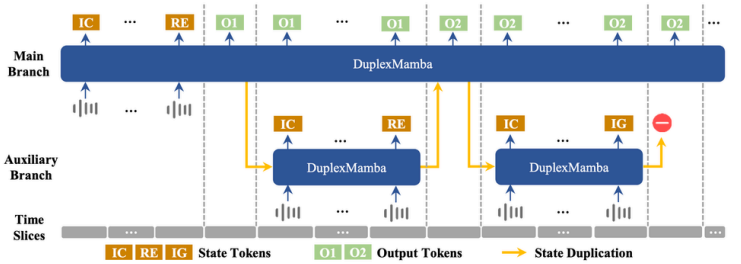


Figure 3: The duplex decoding strategy of DuplexMamba. "IC" is short for the "<incomplete>" token, "RE" for the "<response>" token, and "IG" for the "<ignore>" token. "O1" and "O2" represent the output tokens for query 1 and query 2, respectively. Due to the fixed state size in Mamba-based models, creating an auxiliary branch simply involves duplicating the model's current state.

Novel Contributions

1. **Mamba-Based Speech-Text Duplex Model**:
   - First application of Mamba architecture for duplex speech-text conversation
   - Validated through competitive performance despite smaller parameter count (2.8B vs. 7-8B)
2. **State Token System**:
   - Novel approach using three distinct state tokens for interaction control
   - Validated through superior performance in interruption/non-awakening classification (Table 4)
   - Outperformed VITA and Qwen2-Audio in precision, recall, and F1 scores
3. **Linear Memory Usage**:
   - Demonstrated fixed memory usage regardless of context length
   - Validated through direct comparison with Qwen2-Audio (Figure 4)
   - Qwen2-Audio failed at 16,384 tokens while DuplexMamba maintained stable memory usage
4. **Time-Sliced Processing**:
   - Innovative approach to process both input and output in 2-3 second chunks
   - Validated through case studies showing successful real-time interactions
5. **Branch-Based Decoding Strategy**:
   - Leverages Mamba's fixed-state representation for efficient parallel processing
   - Enables true duplex operation with minimal computational overhead
   - Validated through real-world interaction scenarios

Inference in Real-World Use Cases

The paper demonstrates real-world inference through two key interaction scenarios:

1. **Interruption Interaction**:
   - User begins with "Tell me about the most significant scientific discoveries"
   - Model responds with "The most significant scientific discoveries include..."
   - While generating, user interrupts with "create a list of the 3" followed by "in the 20th century"
   - Model detects important interruption via state token (`<response>`)
   - Seamlessly transitions to answering new query: "Here are 3 significant scientific discoveries in the 20th century..."
2. **Non-Awakening Interaction**:
   - Model is actively generating a response

- Background noise or conversation occurs ("I went into shock")
- Model detects irrelevant input via state token (`<ignore>`)
- Continues with original response without disruption

Key inference advantages:

- Fixed memory usage regardless of conversation length
- Efficient handling of multiple input streams
- Near-instantaneous response to relevant interruptions
- Automatic filtering of background noise/conversation
- Smooth transitions between different user queries

Baichuan-Omni-1.5 Technical Report. January 26, 2025

Li, Yadong, Jun Liu, Tao Zhang, Tao Zhang, Song Chen, Tianpeng Li, Zehuan Li, et al. "Baichuan-Omni-1.5 Technical Report." arXiv, January 26, 2025. Baichuan-Omni-1.5 T echnical Report .

Model Architecture & Specifications

- **Backbone**: LLM (7B parameters) with multimodal extensions
- **Visual Branch**: NaViT encoder from Qwen2-VL with 2×2 compression via two-layer MLP projector
- **Audio Branch**:
  - Baichuan-Audio-Tokenizer: 8-layer RVQ, 12.5 Hz frame rate
  - Flow matching decoder: U-Net structure with single down/up sampling blocks, 12 intermediate blocks
  - HiFi-GAN vocoder for 24 kHz audio waveform generation

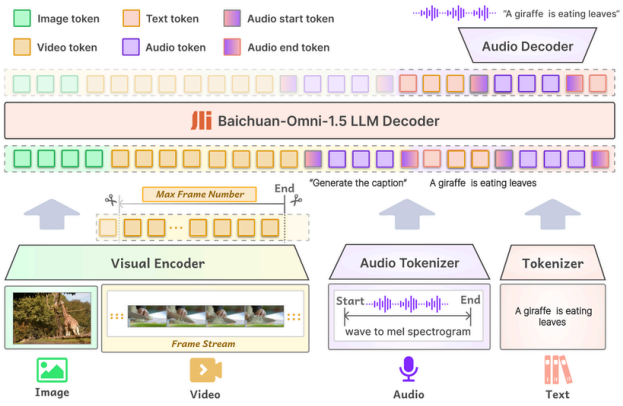


Figure 2: **Architecture of Baichuan-Omni-1.5 .** Our model is designed to process both pure text/audio inputs and combinations of video/image with text/audio. When generating audio, the Baichuan-Omni-1.5 LLM Decoder alternately predicts text tokens and audio tokens. The audio tokens are then decoded by the Audio Decoder to produce the final audio.

Complete Training Process

1. Image-Text Pretraining (300B samples)

- **Stage I**:
  - Frozen LLM and visual encoder
  - Train only visual projector (learning rate 1e-3)
  - Use LAION-5B and other image captioning datasets
- **Stage II**:
  - Unfreeze visual encoder and LLM
  - Train LLM/projector (1e-5) and visual encoder (1e-6)
  - Data mix: interleaved image-text, captions, OCR/chart data + 40% pure text

2. Image-Audio-Text Pretraining (887k hours of speech-text)

- **Stage I**:
  - Frozen LLM, visual modules, audio tokenizer
  - Train only audio embedding layers and audio head (learning rate 1e-4)
  - Use ASR, TTS, interleaved and audio-text paired data
- **Stage II**:
  - Train all parameters except visual encoder and audio tokenizer (learning rate 1e-5)
  - Data mix: 20% audio, 40% image, 40% pure text

3. Omni-Modal Pretraining

- Train all parameters with cross-modal interaction datasets
- Extend maximum sequence length to 64k
- Video processing: 1 frame/second, max 32 frames, 560×1120 resolution
- Low learning rate (4e-6) for fine-grained cross-modal alignment

4. Multimodal Supervised Fine-Tuning (SFT)

- 17 million data pairs across modalities
- **Image SFT**: Balanced across multiple competencies (GeneralQA, OCR (26.51%), Graphical, Mathematics, Spatiotemporal, Captioning, Medical)
- **Video SFT**: 100K high-quality samples across task types, curated using GPT-4o
- **Audio SFT**: Created using 10,000 distinct voices with ASR validation

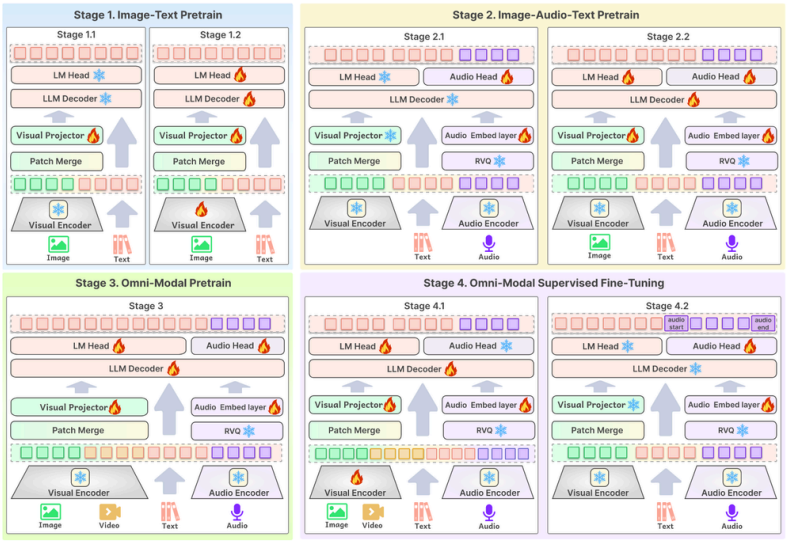


Figure 5: **Training Pipeline of Baichuan-Omni-1.5** . The pretraining phase is divided into three stages to incrementally incorporate vision and audio into the LLM while relieving modality conflicts. Stage 1 focuses on image-text training, which extends an LLM to process and understand visual input. Stage 2 extends an LLM pre-trained on visual data to understand audio input in end-to-end manner by incorporating our Baichuan-Audio-Tokenizer, a newly introduced audio embedding layers and an independent audio head. Stage 3 focuses on training Baichuan-Omni-1.5 using high-quality cross-modal interaction datasets encompassing image-audio-text and video-audio-text format, and extends the maximum sequence length to 64k to support long audio and video stream. Stage 4 enhances the model's instruction following and audio capabilities through supervised fine-tuning with omni-modal data. Stage 4.1: Freeze the Audio Head using omni-modal understanding data to boost modality interactivity and multitasking comprehension. Stage 4.2: Activate only the Audio Head and Audio Embed layer, with audio generation data to improve speech generation capabilities.

Training Data Composition

- **Text**: 150.7 million entries of pure text data
- **Image**: DenseFusion-1M, Synthdog, DreamLIP, LAION-5B, plus in-house books/papers, specialized OCR/chart data
- **Video**: ShareGPT4Video, Koala, WebVid, ActivityNet-QA, plus GPT-4o enhanced YouTube captions
- **Audio**: ASR, AQA, Speech-to-Text Translation, TTS data, interleaved text-speech
- **Cross-modal**: 100B tokens of image-audio-text and video-audio-text interaction data
- **Medical**: PubMed and specialized medical datasets (dermatology, pathology, ophthalmology)

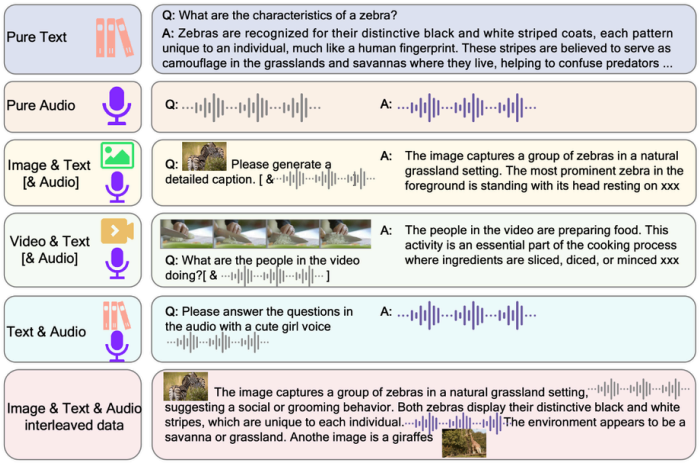


Figure 3: **Pretrain Data illustration of Baichuan-Omni-1.5 .** We construct an extensive omni-modal dataset, including text, image-text, video-text, audio-text, and their interactions. Our collection also contains interleaved image-audio-text and video-audio-text data.

Table 1: Detailed statistics of the training data of image pretrain.

Phase	Type	Public Datasets	Public	In-House
Pretrain	Pure-Text	-	-	150.7M
	Caption	[86][67][189][23]	33.2M	49.1M
	Interleaved	[71]	19.1M	28.7M
	OCR	[57]	12.4M	7.8M
Total	-	-	71.3M	238.2M

Table 2: Detailed statistics of the training data of video pretrain.

QA Type	Dataset Name	Public Datasets	Questions
Description	Synthetic Data	-	300K
	ShareGPT-4o	[29]	2K
	Koala	[150]	30M
QA	Synthetic Data	[80][82][157]	164K
	VideoChatGPT-Plus	[107]	318K
	ShareGemini	[131]	205K
Total	-	-	31M

Table 3: Detailed statistics of the training data of audio pretrain.

Type	Task	Data Format	Hours (k)
Audio Understanding	Automatic Speech Recognition (ASR)	`<prompt, audio, transcript>`	185
	Audio Query Answer (AQA)	`<prompt, audio, response>`	21
	Speech-to-Text Translation (S2TT)	`<prompt, audio, translated_text>`	15
	Audio-Text Interleaved (INTLV)	`<audio_1, text_2, audio_3, text_4, ...>`	393
Audio Generation	Text-to-Speech (TTS)	`<text, audio>`	51
	Interleaved Text-to-Speech (ITTS)	`<text_1, audio_1, text_2, audio_2, ...>`	142
	Pure Audio	`<audio>`	80
Total	-	-	887

Table 4: **Omni-modal SFT data statistics for Baichuan-Omni-1.5** . Here we summarize the category and quantities of our SFT dataset.

Category	**Text**	**Image**	**Video**	**Audio**	**Image-Audio**
Quantity	400K	16M	100K	282K	60K

Table 5: **Image SFT data for Baichuan-Omni-1.5** . This table summarizes the image SFT dataset categories, their sources, and proportions for various tasks.

Scene	**Source**	**Proportion**
GeneralQA	Leopard-Instruct [61], LLaVA-OneVision-Data [76], MMInstruct-GPT4V [99], the Cauldron [72], GeoGPT4V-1.0 [15], MMDU [102], Lova3 [188], CaD-Inst [14], VisionArena-Battle [25], Q-Instruct-DB [154], MultipanelVQA [41], ConMe [58], FABAInstruct [90], ScienceQA [128], MapQA [16], Others	32.26%
OCR	MathWriting [48], WebSight [73], ST-VQA [11], GQA [60], HME100K [169], UberTextQA [10], OCR-VQA [117], TallyQA [2], SlideVQA [139], VizWiz [10], NorHand-v3 [140], LLaVAR [186], Textualization [40], PViT [182], Others	26.51%
Graphical	DVQA [64], TinyChart [179], Chart2Text [66], ArxivQA [84], ChartLlama [52], InfographicVQA [112], FlowVQA [137], MultiChartQA [194], ChartGemma [111], UniChart [109], TAT-DQA [193], PlotQA [116], FigureQA [65], MMTab [191], Others	9.04%
Mathematics	MathV-360K [132], Geo170k [46], R-COT [34], A-OKVQA [130], Super-CLEVR [94], CLEVR-Math [96], TabMWP [105], GeoQA+ [5], MAVIS [181], Iconqa [106], UniGeo [17], PUMA_VarsityTutors [195], Others	10.31%
Spatiotemporal	CCTSDB2021 [177], SODA10M [51], EmbSpatial [36], LLaVA-VSD [62], SpatialSense [163], SpatialMM [133], Whatsup [12], VSR [180], SpatialSense [163], Others	2.63%
Captioning	TextCaps [134], MMsci [92], Synthetic Data, Others	8.23%
Medical	PubMed [113], HAM10000 [144], PMC-VQA [184], PathVQA [54], AIROGS [30]), MedFMC [147], Kvasir-VQA [47], IU X-ray [33], VQA-RAD [70], DME VQA [141], and other specialized medical datasets	11.02%

Novel Contributions

1. **Audio Tokenizer Design**: The 8-layer RVQ audio tokenizer (Baichuan-Audio-Tokenizer) optimally balances semantic and acoustic information, enabling seamless integration with the LLM.
2. **Multi-stage Training Strategy**: Progressive integration of modalities preserves individual modality strengths while enabling cross-modal synergy.
3. **Cross-modal Interaction Data**: Creation of 100B tokens of cross-modal data by converting 25% of text in image-text pairs to audio using 44 voice types.
4. **Medical Benchmarking**: Collection of comprehensive medical evaluation benchmark (OpenMM-Medical) for standardized testing.
5. **Maintaining Text Performance**: Successfully preserved strong text understanding capabilities despite multimodal training - addressing a common weakness in omni-modal models.

Inference Use Case

The model can:

- Process images, videos, text inputs, and audio simultaneously
- Generate both text and speech responses in real-time
- Understand complex medical images at expert level
- Perform end-to-end audio interactions without separate ASR/TTS modules
- Support bilingual (Chinese/English) voice conversations

Open-Sourced Resources

1. **Baichuan-Audio-Tokenizer**: 8-layer RVQ tokenizer for audio processing
2. **OpenAudioBench**: End-to-end audio understanding and generation benchmark
3. **OpenMM-Medical**: Comprehensive medical understanding benchmark

---

VITA-1.5: Towards GPT-4o Level Real-Time Vision and Speech Interaction. January 21, 2025

Fu, Chaoyou, Haojia Lin, Xiong Wang, Yi-Fan Zhang, Yunhang Shen, Xiaoyu Liu, Haoyu Cao, et al. “VITA-1.5: Towards GPT-4o Level Real-Time Vision and Speech Interaction.” arXiv, January 21, 2025. VITA-1.5: Towards GPT-4o Level Real-Time Vision and Speech Interaction .

[Link]

Model Architecture Details

VITA-1.5 uses a "Multimodal Encoder-Adaptor-LLM" configuration with the following components:

1. **Visual Processing**:
   - Visual Encoder: InternViT-300M (448×448 pixel input, generating 256 visual tokens per image)
   - Vision Adapter: Two-layer MLP to map visual features to LLM-compatible tokens
   - Dynamic patching for high-resolution images to capture local details
2. **Audio Processing**:
   - Speech Encoder: Multiple downsampling convolutional layers (4x downsampling) with 24 Transformer blocks (1024 hidden size, ~350M parameters)
   - Speech Adapter: Multiple convolutional layers with 2x downsampling
   - Speech Decoder: TiCodec with a single 1024-size codebook design for simpler decoding
   - Dual Speech Generation Decoders (after text tokens):
     - Non-Autoregressive (NAR) Speech Decoder: Processes text tokens globally for initial speech token distribution
     - Autoregressive (AR) Speech Decoder: Generates higher quality speech tokens sequentially
     - Both decoders use 4 LLaMA decoder layers (896 hidden size, ~120M parameters each)
3. **Video Processing**:
   - Treats videos as multiple-image input with adaptive frame sampling:
     - <4 seconds videos: 4 uniformly sampled frames
     - 4-16 seconds videos: 1 frame per second
     - 16 seconds videos: 16 uniformly sampled frames
   - No dynamic patching for video frames to maintain processing efficiency

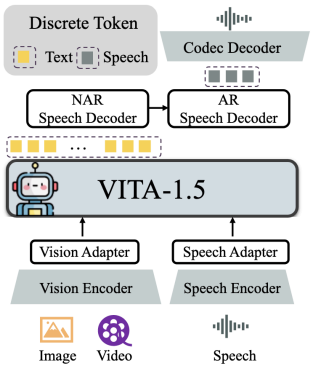


Figure 2: **Overall Architecture of VITA-1.5**. The input side consists of vision and audio encoders, along with their adapters connected to a LLM. The output side has an end-to-end speech generation module, rather than directly using an external TTS model as the initial VITA-1.0 version [16].

Complete Training Process and Data Used

Stage 1: Vision-Language Training

Stage 1.1: Vision Alignment

- **Objective**: Bridge the gap between vision and language
- **Data**: 20% of descriptive caption data
- **Trainable Components**: Only visual adapter (other modules frozen)
- **Process**: Initial alignment of visual modality with LLM

Stage 1.2: Vision Understanding

- **Objective**: Teach LLM to transcribe image content
- **Data**: All descriptive caption data
- **Trainable Components**: Visual encoder, adapter, and LLM
- **Process**: Establish strong vision-language connection through descriptive texts

Stage 1.3: Vision SFT

- **Objective**: Enhance instruction-following for visual QA tasks
- **Data**: All QA data + 20% of descriptive caption data
- **Trainable Components**: Visual encoder, adapter, and LLM
- **Process**: Enable model to understand visual content and answer questions following instructions

Stage 2: Audio Input Tuning

Stage 2.1: Audio Alignment

- **Objective**: Reduce discrepancy between audio and language
- **Data**: 11,000 hours of speech-transcription pairs
- **Two-step approach**:
  - **Speech Encoder Training**: Using CTC loss to predict transcription from speech
  - **Speech Adapter Training**: Integrate encoder with LLM to output transcription text
- **Special Features**: Trainable input tokens to guide speech understanding

Stage 2.2: Audio SFT

- **Objective**: Introduce QA functionality with speech questions and text answers
- **Data**: 4% of caption data + 20% of QA data (with ~50% of text questions replaced by speech versions)
- **Trainable Components**: Visual encoder/adapter, audio encoder/adapter, and LLM
- **Special Feature**: Addition of a classification head to distinguish between speech and text inputs

Stage 3: Audio Output Tuning

Stage 3.1: Codec Training

- **Objective**: Train a codec model with a single codebook using speech data
- **Function**: Map speech to discrete tokens and back to speech stream
- **Note**: Only the decoder is used during inference

Stage 3.2: NAR + AR Decoder Training

- **Data**: 3,000 hours of text-speech paired data
- **Process**:
  - Text fed through tokenizer and embedding layer to get vectors
  - Speech fed into codec encoder to get speech tokens
  - Text embedding vectors sent to NAR decoder for global semantic features
  - Features sent to AR decoder to predict corresponding speech tokens
- **Key Point**: LLM is frozen during this stage to preserve multimodal performance

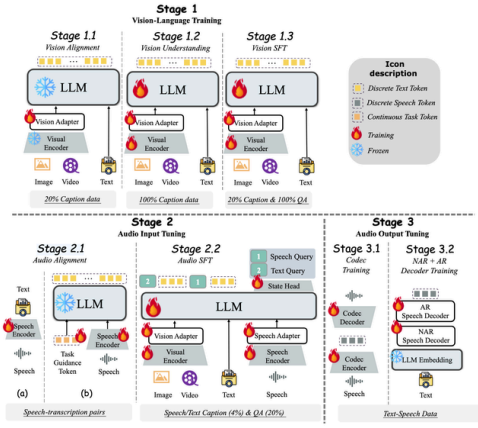


Figure 3: **Training Pipeline of VITA-1.5**. The training process is divided into three stages to incrementally incorporate vision and audio into the LLM while relieving modality conflicts. Stage I focuses on **Vision-Language Training**, including vision alignment (Stage 1.1, using 20% caption data from Table 1), vision understanding (Stage 1.2, using 100% caption data), and instruction tuning for visual QA (Stage 1.3, using 20% caption data and 100% QA data). Stage 2 introduces **Audio Input Tuning**, with audio alignment (Stage 2.1, utilizing 11,000 hours of speech-transcription pairs) and instruction tuning for speech QA (Stage 2.2, sampling 4% caption data and 20% QA data). Finally, Stage 3 focuses on **Audio Output Tuning**, including the training of the codec model (Stage 3.1, using 3,000 hours of text-speech data) and speech decoder training (Stage 3.2). The percentages shown in the image correspond to the data sampling ratios specified in Table 1.

Training Data Details

1. **Image Captioning**:
   - ShareGPT4V, ALLaVA-Caption, SharedGPT4o-Image, synthetic data
2. **Image QA**:
   - LLaVA-150K, LLaVA-Mixture-sample, LVISInstruct, ScienceQA, ChatQA
   - Subsets from LLaVA-OV including general image QA and mathematical reasoning
3. **OCR & Diagram Understanding**:
   - Anyword-3M, ICDAR2019-LSVT, UReader, SynDOG, ICDAR2019-LSVT-QA
   - Related data from LLaVA-OV
4. **Video Data**:
   - ShareGemini and synthetic data
5. **Speech Data**:
   - 110,000 hours of internal speech-transcription paired ASR data (Chinese and English)
   - 3,000 hours of text-speech paired data (TTS-generated) for speech decoder training

Table 1: Training data of multimodal instruction tuning. The images of the synthetic data come from open-source datasets like Wukong [19], LAION [46], and CC12M [5].

Data Scenario	QA Type	Dataset Name	Questions (K)	Language
General Image	Description	ShareGPT4V	99.50	Eng
		ALLaVA-Caption	697.40	Eng
		ShareGTP4o-Image	55.50	Eng
		Synthetic Data	593.70	CN
	QA	LLaVA-150K	218.36	CN
		LLaVA-Mixture-sample	1872.10	Eng
		LVIS-Instruct	939.36	Eng
		ScienceQA	12.72	Eng
		ChatQA	7.39	Eng
		LLaVA-OV General	1754.65	Eng
		LLaVA-OV Math Reasoning	1140.92	Eng
		Synthetic Data	212.68	CN
OCR & Diagram	Description	Anyword-3M	1709.30	CN
		ICDAR2019-LSVT	366.30	CN
		UReader	100.00	Eng
		SynDOG-EN	100.00	Eng
		SynDOG-CN	101.90	CN
	QA	ICDAR2019-LSVT-QA	630.08	CN
		LLaVA-OV Doc Chart Screen	4431.50	Eng
		LLaVA-OV General OCR	404.20	Eng
General Video	Description	ShareGemini	205.70	CN
		Synthetic Data	569.40	CN & Eng
	QA	Synthetic Data	4336.30	CN & Eng
Pure Text	QA	Synthetic Data	1574.20	CN & Eng
Total			22133.16	CN & Eng

Novel Contributions and Validated Approaches

1. **Three-Stage Progressive Training Methodology**:
   - Successfully validated that gradually introducing modalities reduces training conflicts
   - Proved that this approach preserves existing capabilities while adding new ones

- Demonstrated that vision capabilities are maintained even after audio training stages

2. **Single-Codebook Design for Speech Generation**:
   - Simplified the decoding process during inference
   - Improved processing efficiency without sacrificing speech quality
3. **End-to-End Speech Handling**:
   - Eliminated separate ASR and TTS modules
   - Validated that this approach significantly accelerates multimodal response speed
   - Demonstrated near real-time vision and speech interaction
4. **Dual Speech Decoder Approach**:
   - NAR + AR decoder combination proved effective for high-quality speech generation
   - Successfully implemented with LLaMA decoder layers
5. **Modality Conflict Resolution**:
   - Proved that the three-stage training strategy effectively mitigates conflicts between modalities
   - Demonstrated that speech data can be incorporated without degrading vision performance

Inference in Real-World Use Cases

In real-world applications, VITA-1.5 enables:

1. **Multimodal Interaction**:
   - User speaks a query while showing an image/video
   - Model processes both speech and visual inputs simultaneously
   - Responds with speech output without additional processing steps
2. **End-to-End Processing**:
   - No external ASR system needed to convert user speech to text
   - No external TTS system needed to convert model response to speech
   - Reduced latency due to elimination of these external systems
3. **Inference Pipeline**:
   - Speech input → Speech Encoder → Speech Adapter → LLM
   - Image/video input → Visual Encoder → Vision Adapter → LLM
   - LLM generates text tokens
   - For speech output: Text tokens → NAR Speech Decoder → AR Speech Decoder → Speech Decoder of Codec model → Waveform
4. **Performance Characteristics**:
   - Near real-time vision and speech interaction
   - Maintained strong vision-language capabilities while adding speech functionality
   - Leading accuracy in both Mandarin and English ASR tasks

Open-Sourced Resources

The authors have open-sourced the following resources:

1. **Training Code**: Complete codebase for implementing the three-stage training methodology
2. **Inference Code**: Code for deploying and running the model in real-world applications
3. **Model Weights**: Pre-trained VITA-1.5 model
4. **GitHub Repository**: GitHub - VITA-MLLM/VITA: ✨✨VITA-1.5: Towards GPT-4o Level Real-Time Vision and Speech Interaction (with over 2,000 stars at the time of publication)

The paper does not explicitly mention whether the training datasets are being open-sourced, though the model architecture details and training methodologies are fully disclosed to enable reproduction of their work.

MinMo: A Multimodal Large Language Model for Seamless Voice Interaction. January 10, 2025

Chen, Qian, Yafeng Chen, Yanni Chen, Mengzhe Chen, Yingda Chen, Chong Deng, Zhihao Du, et al. “MinMo: A Multimodal Large Language Model for Seamless Voice Interaction.” arXiv, January 10, 2025. MinMo: A Multimodal Large Language Model for Seamless Voice Interaction .

Model Architecture

- **Overall**: ~8 billion parameters, aligned multimodal approach on a pretrained text LLM
- **Components**:
  - **Voice Encoder**: Initialized with SenseVoice-large encoder for robust voice understanding
  - **Input Projector**: Two-layer Transformer with CNN for dimensional alignment and downsampling
  - **Text LLM**: Qwen2.5-7B-instruct as the foundation model
  - **Output Projector**: Single-layer linear module for dimensional alignment
  - **Voice Token LM**: Uses CosyVoice 2 LM module for efficient audio synthesis
  - **Token2wav Synthesizer**: Flow-matching model (tokens to mel spectrograms) + vocoder (mel spectrograms to waveforms)

- **Full Duplex Predictor**: Single-layer Transformer with linear softmax output layer

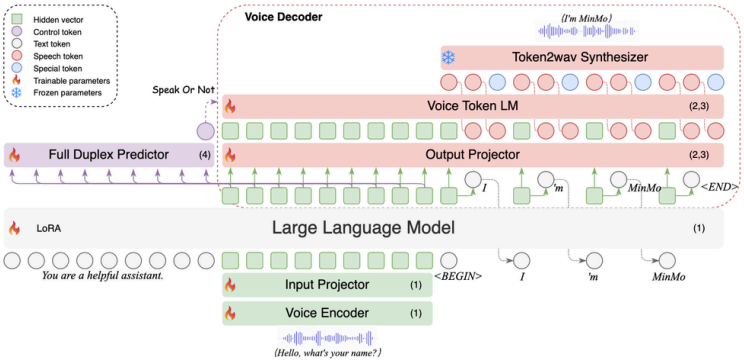


Figure 3: The overall architecture of MinMo. Table 1 provides detailed descriptions of each module in this diagram.

Module	Description	Number of Parameters
Voice Encoder	Initialized with the encoder parameters of the pre-trained SenseVoice-Large audio understanding model (An et al., 2024)	∼ 636M
Input Projector	2 Transformer layer and 1 CNN layer for dimensional transformation and perform 2x downsampling on the input	∼ 170M
Large Language Model	Initialized with Qwen2.5-7B-instruct (Team, 2024)	7B
Output Projector	Linear layer for dimensional transformation	∼ 6M
Voice Token LM	Initialized with the LLM of the pre-trained CosyVoice2 (Du et al., 2024b)	∼ 370M
Full Duplex Predictor	1 Transformer layer and 1 linear-softmax output layer, both randomly initialized	∼ 18M

Table 1: Descriptions of the modules in MinMo as depicted in Figure 3. MinMo has approximately 8 billion parameters in total.

Training Data

- **Total**: 1.4+ million hours of diverse speech data
- **Speech-to-Text (~1.2M hours)**:
  - ASR, speech-to-text translation, language identification
  - Contextual biasing speech recognition, speech emotion recognition
  - Audio event detection, speaker analysis, spoken language smoothing
- **Text-to-Speech (~171K hours)**:
  - 170K hours from CosyVoice 2 (Chinese, English, Korean, Japanese)
  - 1K hours of instruction-controlled audio generation
- **Speech-to-Speech (~10.1K hours)**:
  - 10K hours of multi-turn conversational speech
  - 100 hours of style-controllable multi-turn speech
- **Speech-to-ControlToken (~4K hours)**:
  - 3K hours from existing datasets (Alimeeting, Fisher, in-house data)
  - 1K hours simulated from MOSS dataset and in-house dialogues

Category	Specific Tasks	Hours
Speech-to-Text	Automatic Speech Recognition (ASR)	630k
	Speech-to-Text Translation (S2TT)	451k
	Language Identification (LID)	34k
	Contextual Bias Speech Recognition	50k
	Speech Emotion Recognition (SER)	48k
	Audio Event Detection (AED)	11k
	Speaker Analysis	24k
	Spoken Language Smoothing	0.4k
	Speech-to-Text Chat	10k
Text-to-Speech	Speech Synthesis	170k
	Instruct Speech Synthesis	1k
Speech-to-Speech	Speech-to-Speech chat	10k
	Style-controllable Speech-to-Speech Chat	0.1k
Speech-to-ControlToken	Full Duplex Interaction	4k

Table 2: The multitask training data for MinMo. Task specifications can be found in Section 4.

Training Process (4 Progressive Stages)

1. **Speech-to-Text Alignment**:
   - **Pre-align**: Updates only Input Projector using ~10% of data
   - **Full-Align**: Updates Input Projector and Voice Encoder with LLM frozen
   - **SFT**: Fine-tunes LLM using LoRA with ~1.3M samples across tasks
2. **Text-to-Speech Alignment**:
   - First trains Output Projector independently
   - Then jointly trains Output Projector and Voice Token LM
   - Includes instruction-based control for expression (emotion, speaking rate, accent)
3. **Speech-to-Speech Alignment**:
   - Updates only Output Projector and Voice Token LM
   - Uses 10K hours of paired audio including style variations
   - Achieves control capabilities with minimal instruction data (< 150 hours)
4. **Duplex Interaction Alignment**:
   - Trains exclusively the Full Duplex Predictor
   - Uses 4K hours of long-form human-human conversations
   - Enables decisions on when to respond or stop generating to listen

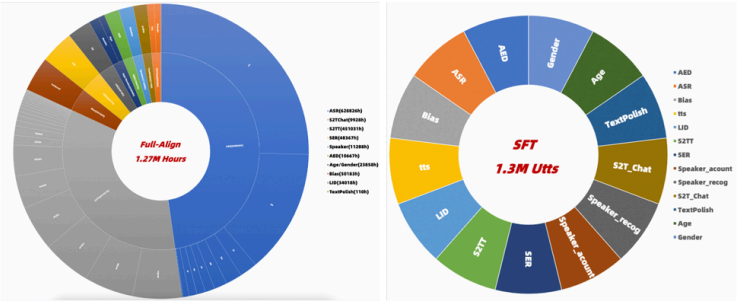


Figure 4: Detailed training data for the Speech-to-Text Alignment stage. **Left:** Data distribution for *Full-Align* training. **Right:** Data distribution for instruction fine-tuning (SFT).

Eval Summary

Description	Dataset	Metric
Speech Recognition and Translation		
Multilingual Speech Recognition	Aishell-2 (Du et al., 2018) Wenetspeech (Zhang et al., 2022) Librispeech (Panayotov et al., 2015) Fleurs (Conneau et al., 2023) CommonVoice (Ardila et al., 2019)	CER&WER
Multilingual Speech Translation	Fleurs (Conneau et al., 2023) CoVoST2 (Wang et al., 2021)	BLEU
Language Identification	Fleurs (Conneau et al., 2023)	Accuracy
Contextual Biasing Speech Recognition	Aishell-1-NE	CER&P&R
Speech Analysis and Understanding		
Speech Emotion Recognition	EMOBox (Ma et al., 2024a)	UA&WA&F1
Audio Event Understanding Speaker Analysis for Gender and Age Vocal Sound Sound Question	AirBench(Yang et al., 2024b)	Accuracy
Speech-to-Text Enhancement		
Spoken Language Smoothing	SWAB (Liu et al., 2025)	S-Faithful&S-Formal
Punctuation&ITN	In-house testset	ChatGPT Score
Voice Generation		
Text-to-Speech	Seed-TTS (Anastassiou et al., 2024)	WER&CER&NMOS
Instruction-following Voice Generation	In-house testset	Accuracy
Voice Chat		
Spoken Question Answering	Web Questions (Berant et al., 2013) Llama Questions (Nachmani et al., 2024) TriviaQA (Joshi et al., 2017)	Accuracy
Spoken Dialogue	AlpacaEval (Li et al., 2023) In-house ChitChat	ChatGPT Score
Full Duplex	Alimeeting (Yu et al., 2022) Fisher (Cieri et al., 2004) Simulation (Sun et al., 2024)	Positive F1-Score

Table 4: Summary of evaluation benchmarks for MinMo in this report.

Novel Contributions

1. **Novel Voice Decoder**:
   - Transforms LLM text outputs into speech with low latency
   - Intermixes semantic vectors and speech tokens in a 5:15 ratio
   - Theoretical latency formula: 5dllm + 15dlm + 15dsyn
   - Balances structural simplicity with high performance
2. **Validation of Style Control in Aligned Models**:
   - Disproves previous claims that aligned multimodal models cannot control speech style
   - Demonstrates 98.4% instruction-following accuracy for style control
   - Enables generation with specified emotions, dialects, speaking rates, and voice mimicking
3. **Efficient Full-Duplex Implementation**:
   - Achieves ~600ms theoretical latency (800ms in practice)
   - Speech-to-text latency of ~100ms
   - Leverages LLM's semantic understanding for real-time interaction decisions
4. **Catastrophic Forgetting Mitigation**:
   - Successfully maintains text LLM capabilities despite large-scale speech training
   - Achieves this through strategic freezing and selective parameter updates

Real-world Inference

- End-to-end latency: ~600ms on L20 GPU
- **Full-duplex functionality**: Allows users to interrupt while system is speaking
- System can:
  - Continue speaking when appropriate
  - Concede and listen to user's new query
  - Respond to user's new query appropriately
- **Style control**: Users can specify emotions, dialects, speaking rates, and request voice mimicking
- Multilingual support across Chinese, English, Korean, and Japanese

Open-sourcing Plans

The paper indicates: "The MinMo project web page is MinMo , and the code and models will be released soon."

This suggests they plan to open-source:

- Model weights/checkpoints
- Implementation code
- Possibly evaluation benchmarks

However, no specific details are provided about which components will be open-sourced or what licensing terms will apply. The paper also doesn't explicitly mention whether training data will be released.

Advancing Speech Language Models by Scaling Supervised Fine-Tuning with Over 60,000 Hours of Synthetic Speech Dialogue Data. December 3, 2024

Zhao, Shuaijiang, Tingwei Guo, Bajian Xiang, Tongtang Wan, Qiang Niu, Wei Zou, and Xiangang Li. “Advancing Speech Language Models by Scaling Supervised Fine-Tuning with Over 60,000 Hours of Synthetic Speech Dialogue Data.” arXiv, December 3, 2024. Advancing Speech Language Models by Scaling Supervised Fine-Tuning... .

[Info]

Model Architecture and Components

- **KEOmni** consists of three main components:
  a. **Speech Encoder**: Whisper-large-v3 encoder with a lightweight speech adapter
  b. **Large Language Model**: LLaMA-3.1-8B-Instruct
  c. **Speech Decoder**: Duration predictor, speech unit generator, and unit-based vocoder

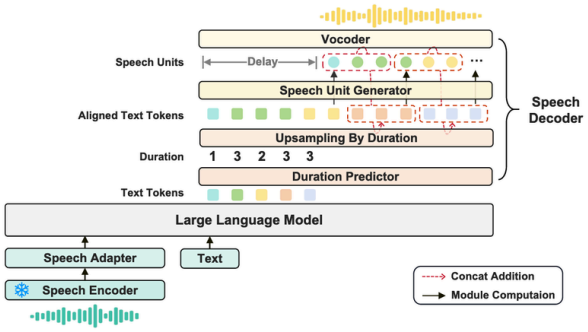


Figure 3: Model architecture of KE-Omni.

Training Process and Data

Dataset Creation

1. **Textual Dialogue Creation**:
   - Source: IndustryInstruction, LaMini-instruction, BELLE datasets
   - Process:
     - Rewriting instructions with Qwen2.5-14B-Instruct
     - Filtering for speech appropriateness, clarity, and safety
     - Post-processing with Qwen2.5-72B-Instruct for conversational style
2. **Speech Synthesis**:
   - Voice library created from WenetSpeech4TTS "Premium" clips
   - 5,000+ real speakers identified and used to create virtual voices
   - CosyVoice used for synthesis with AudioSeal watermarking
   - Quality filtering (rejecting >5% CER for Chinese, >10% WER for English)
3. **Final Dataset - Ke-SpeechChat**:
   - 5.1M+ Chinese dialogues (40,884 hours)
   - 1.7M+ English dialogues (19,484 hours)
   - 42,002 speakers (40,000 users + 2 agents)
   - Split into 5 progressive subsets: XS, S, M, L, XL

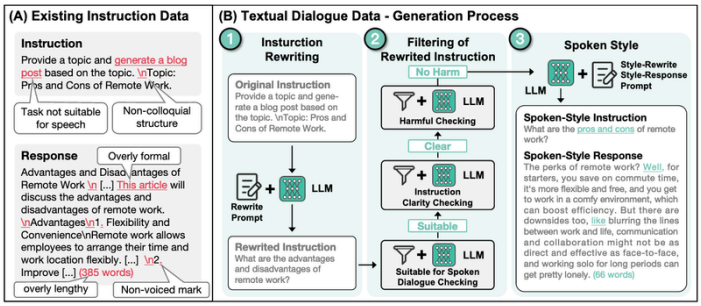


Figure 1: An overview of textual dialogue data process

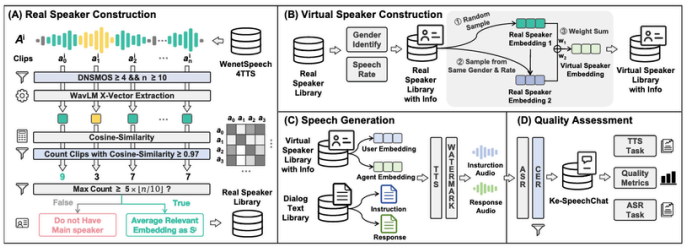


Figure 2: The construction of speech dialogue data. (A) Construction of real speakers. (B) Construction of virtual speakers. (C) Speech generation given textual dialogues and virtual speakers. (D) Quality filtering and assessment.

Model Training

1. **Two-Stage Process**:
   - **Stage 1**: LLM fine-tuning and speech adapter training
     - All dialogues used to enhance reasoning for audio input
     - AdamW optimizer with peak learning rate of 1e-4
     - 2 epochs
   - **Stage 2**: Speech decoder training
     - LLM frozen at this stage
     - Dialogues separated by agent speaker
     - AdamW optimizer with peak learning rate of 2e-4
     - 2 epochs
     - Varying batch sizes to maintain similar training steps

Novel Contributions and Validations

1. **Large-Scale Synthetic Data Approach**:
   - Created the first large-scale speech interaction dataset for Chinese
   - Demonstrated synthetic data can match or exceed real datasets in quality
   - Validated through DNSMOS (3.41±0.14) and UTMOS (3.47±0.35) scores
2. **Modality Alignment Scaling**:
   - Proved significant improvement in speech-text alignment with larger datasets
   - Sharp decline in CER/WER when scaling from XS to S subset
   - Best alignment achieved with L subset (not XL), suggesting optimal training dynamics
3. **Speech Quality Assessment**:
   - Showed progressive enhancement of speech quality with larger training data
   - Demonstrated competitive performance on VoiceBench compared to specialized models
4. **Dataset Quality Validation**:
   - ASR task: Despite less training data, KeASR showed competitive performance
   - TTS task: KeTTS outperformed CosyVoice in CER and UTMOS while maintaining similar speaker similarity

Inference Process in Real Use Case

While not explicitly detailed, the inference flow would likely be:

1. User speaks into the system
2. Speech encoder (Whisper) processes audio and converts to features (10 frames per second)
3. LLM receives the speech embeddings and generates text response
4. Duration predictor determines timing for each text token
5. Speech unit generator creates discrete speech units chunk by chunk
6. HiFi-GAN vocoder synthesizes the final audio response
7. System delivers the audio response to the user with low latency

Open-Sourcing Plans

The paper mentions plans to release the following after thorough risk assessment:

- The KE-Omni model
- The Ke-SpeechChat dataset (with its various subsets)
- Code for model training and evaluation

The authors have already provided a demo at https://huggingface.co/spaces/KE-Team/KE-Omni, allowing users to experience the model firsthand.

Freeze-Omni: A Smart and Low Latency Speech-to-Speech Dialogue Model with Frozen LLM. November 1, 2024

Wang, Xiong, Yangze Li, Chaoyou Fu, Lei Xie, Ke Li, Xing Sun, and Long Ma. "Freeze-Omni: A Smart and Low Latency Speech-to-Speech Dialogue Model with Frozen LLM." arXiv, November 1, 2024. Freeze-Omni: A Smart and Low Latency Speech-to-speech Dialogue... .

Code

Model Details

Core Architecture

- **Foundation LLM**: Qwen2-7B-Instruct (used as frozen backbone)
- **Speech Encoder**:
  - Multi-layer convolution with 4× downsampling
  - 24 transformer layers with 1024 hidden size
  - Adapter with multi-convolution layer (2× downsampling)
  - Total parameters: ~350M
  - Output frame rate: 12.5Hz
  - Input: Mel-filter bank features (25ms window, 10ms shift)
- **Speech Decoder**:
  - Codec: Customized TiCodec with 1024-size single-codebook
  - Speech token frequency: 40Hz
  - NAR and AR speech decoders: 4-layer Llama decoder layers (896 hidden size)
  - Total parameters: ~120M
  - Output sample rate: 24,000Hz

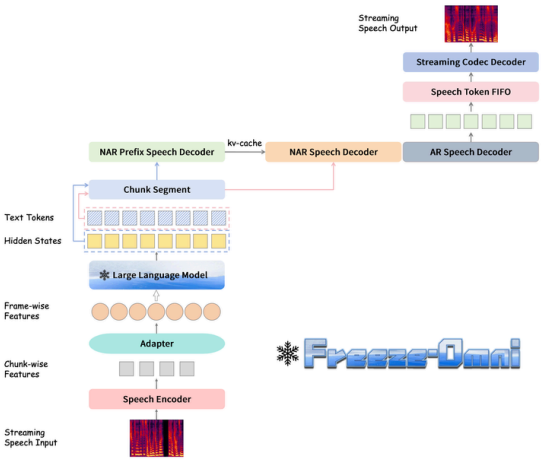


Figure 1: Overview of proposed Freeze-Omni. The streaming speech input forms chunk-wise features through the speech encoder, and then is connected to the LLM through the adapter. The LLM generates hidden states and text tokens, which are sent to the NAR prefix speech decoder and the NAR speech decoder in the form of chunks respectively after chunk segmentation. Finally, the AR speech decoder sends the generated tokens into the speech token FIFO and the streaming codec decoder generates streaming speech output from the FIFO according to a fixed speech token chunk size.

Complete Training Process and Data Usage

Speech Input Training

Stage 1: Speech Recognition Pretraining

- **Data**: 110,000 hours internal speech-text paired ASR data (Chinese and English)
- **Objective**: Train standard speech recognition with CTC loss
- **Learning rate**: 2e-4
- **What's trained**: Speech encoder only

Stage 2: LLM Integration

- **Data**: Same 110,000 hours ASR data
- **Objective**: Connect speech encoder to LLM via adapter
- **Method**: Added trainable special tokens to guide LLM
- **Learning rate**: 1e-4
- **What's trained**: Speech encoder and adapter (LLM frozen)

Stage 3: Speech-to-Text Dialogue

- **Data**: 60,000 multi-round Q&A examples (synthetic speech created with zero-shot TTS)
- **Objective**: Enable full speech input to text output capability
- **Method**: Added trainable prompt embeddings before each question
- **Learning rate**: 6e-4
- **What's trained**: Only prompt embeddings (speech encoder and LLM frozen)

Speech Output Training

Stage 1: Codec Model Training

- **Data**: 3,000 hours of TTS-generated speech data
- **Objective**: Train single-codebook codec for token extraction
- **Method**: Standard TiCodec training approach

Stage 2: Decoder Training

- **Data**: 3,000 hours text-speech paired data (TTS-generated)
- **Objective**: Train NAR and AR decoders to generate speech from text
- **Method**: Text tokens through LLM embedding layer → NAR decoder → AR decoder
- **Learning rate**: 5e-5
- **What's trained**: NAR and AR decoders (LLM embedding layer frozen)

Stage 3: LLM Output Alignment

- **Data**: Same 60,000 multi-round Q&A examples
- **Objective**: Align speech decoder with LLM's output style
- **Method**: Added NAR prefix speech decoder to model LLM hidden states
- **Learning rate**: 5e-5
- **What's trained**: Only NAR prefix speech decoder (all other components frozen)

Duplex Dialogue Training

- Incorporated during Stage 3 of speech input training
- **Method**: Multi-task optimization with chunk-level state prediction
- **States**:
  - State 0: Continue receiving speech
  - State 1: End of speech, LLM can interrupt
  - State 2: End of speech, no interruption needed
- **Training**: Added cross-entropy loss for state classification alongside LLM loss

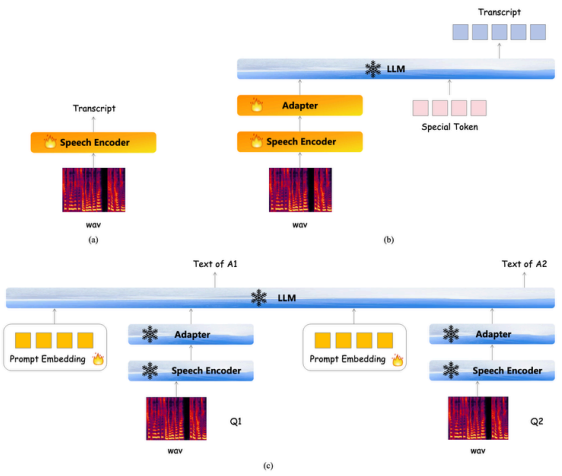


Figure 2: The 3-stage training method for modeling of speech input, the speech encoder in (c) is used in Freeze-Omni finally.

Validation Results

1. **Speech Input Performance**: Detailed ASR accuracy on different evaluation datasets
   - Best configuration achieved with specific chunk size settings
2. **Speech Output Quality**:
   - Measured with ASR accuracy on 1,000 synthesized utterances
   - Proved NAR prefix decoder reduced bad cases (lower CER%)
3. **Spoken Question-Answering**:
   - Evaluated on LlaMA-Questions, Web Questions, and Trivia QA
   - Smaller accuracy gap between speech and text compared to other models (Moshi)

- Proved preservation of LLM intelligence in speech modality

4. **Latency Analysis**:
   - Statistical latency measured in components
   - Non-statistical latency measured manually (160-320ms)
   - Overall real-world latency around 1.2 seconds including network delay

Novel Contributions and Validation

Key Innovations

1. **Completely Frozen LLM Architecture**:
   - Validated by showing minimal performance gap (compared to text modality) on question-answering tasks
   - First approach to achieve speech-to-speech capability without any LLM parameter updates
2. **Three-Stage Training Strategy**:
   - Demonstrated efficient training with minimal resources (only 60,000 Q&A examples)
   - Completed on just 8 GPUs (much less than typical multimodal model training)
3. **NAR Prefix Speech Decoder**:
   - Novel approach for aligning speech output with LLM hidden states
   - Validated by showing reduced CER (%) compared to models without prefix fine-tuning
   - Higher top-k values showed better robustness with prefix decoder
4. **Dynamic Chunk Training**:
   - Enhanced robustness for different chunk sizes during inference
   - Trade-off validated between performance and flexibility
5. **Chunk-level State Prediction**:
   - Enabled natural duplex dialogue with interruption capability
   - Implemented as multi-task training without affecting LLM parameters

Real-World Inference Process

The real-world inference uses a "model as a server" approach:

1. Multiple models started simultaneously as a server pool
2. When user speech detected by VAD (Voice Activity Detection):
   - Speech sent to server in chunks
   - Server schedules idle model to process current chunk
   - Each model maintains separate kv-cache and CNN cache
   - Server saves inference cache for each user
3. Inference flow:
   - VAD detects speech → Speech sent in chunks
   - Chunk-wise streaming speech encoder processes audio
   - LLM + state classifier determines whether to continue listening, interrupt, or wait
   - When generating response, LLM outputs text tokens and hidden states
   - NAR prefix decoder processes hidden states to kv-cache
   - NAR decoder generates semantic features from text tokens
   - AR decoder produces speech tokens, converted to waveform by codec
   - Output delivered in streaming mode with ~1.2s total latency

Open-Source Components

The paper does not explicitly mention which components are being open-sourced. There is no specific section dedicated to listing available resources, models, code, or datasets. This suggests that at the time of publication, there may not have been immediate plans to open-source the complete system, though this could change after publication.

Potential candidates for open-sourcing based on their importance:

- Training methodology and configuration details
- System architecture design
- Implementation code for the frozen LLM approach
- Fine-tuned models (particularly the speech encoders and decoders)

Without explicit mention in the paper, it remains unclear what exactly will be made available to the research community.

Ichigo: Mixed-Modal Early-Fusion Realtime Voice Assistant. October 20, 2024.

Dao, Alan, Dinh Bach Vu, and Huy Hoang Ha. “Ichigo: Mixed-Modal Early-Fusion Realtime Voice Assistant.” arXiv, October 20, 2024. Ichigo: Mixed-Modal Early-Fusion Real time Voice Assistant .

[Link]

Model Architecture

Ichigo is a tokenized early-fusion multimodal model based on:

- **Base LLM**: Llama-3.1-8B-Instruct
- **Tokenization**: WhisperVQ for speech conversion to discrete tokens
- **Sound Representation**: 512-token codebook with 64 dimensions
- **Fusion Approach**: Unified transformer architecture for both speech and text modalities
- **Special Tokens**: <|sound_start|>, <|sound_end|>, and sound tokens formatted as <|sound_dddd|>
- **Initialization Strategy**: New token embeddings initialized by averaging all embeddings in the current vocabulary (improving convergence speed)

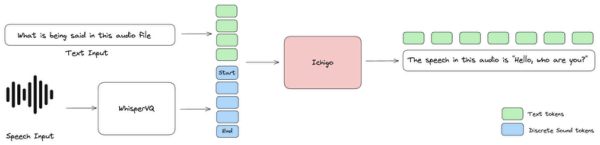


Figure 1. Ichigo represents speech and text modalities as discrete tokens and uses a uniform transformer-based architecture. It uses WhisperVQ to quantize speech into discrete tokens in the same manner with original text modality.

Complete Training Process

Phase 1: Continual Pre-training on Multilingual Speech

- **Objective**: Introduce speech representation into the model vocabulary
- **Data**:
  - 10,000 hours of English audio + 6,000 hours across 7 other languages
  - Sources: MLS English 10k dataset, Multilingual LibriSpeech
  - Languages: English, German, Dutch, Spanish, French, Italian, Portuguese, Polish
- **Processing**: Audio converted to discrete tokens using WhisperVQ
- **Results**: MMLU performance dropped from 0.69 → 0.42 (significant degradation)

Phase 2: Balancing Original Performance and Speech Modality

- **Objective**: Recover general capabilities while enhancing speech skills
- **Data**:
  - Increased from 0.92M to 1.89M samples
  - Distribution: 70% speech instruction prompts, 20% speech transcription prompts, 10% text-only prompts
  - Replaced single transcription token with six diverse prompts
- **Results**: MMLU recovered from 0.42 to 0.63 (reducing degradation to ~10%)

Phase 3: Enhancement Fine-tuning

- **Objective**: Improve handling of inaudible inputs and multi-turn conversations
- **Data**:
  - 150,000 samples for multi-turn conversations
    - 90% two-turn conversations
    - 10% with four or more turns
  - 8,000 inaudible input samples
    - Created by randomizing the 513 sound tokens
    - Used Qwen2.5-72B to generate refusal responses
    - Matched sequence length distribution between audible and inaudible data

Table 1. Training Hyper-parameters for Ichigo's three-stage process.

Parameter	Pre-training	Instruction FT	Enhancement FT
Weight Decay		0.005	
Learning Scheduler		Cosine	
Optimizer		AdamW Fused	
Precision		bf16	
Hardware	10x A6000	8x H100	8x H100
Train time	45h	10h	3h
Steps	8064	7400	644
Global batch size	480	256	256
Learning Rate	2×10^{-4}	7×10^{-5}	1.5×10^{-5}
Warmup Steps	50	73	8
Max length	512	4096	4096

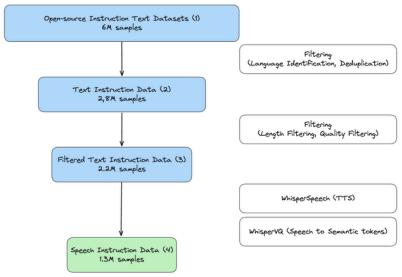


Figure 2. Data Processing Pipeline for Speech Instruction Dataset Generation. This chart illustrates the multi-stage filtering and conversion process, starting from 6M samples of multiple open-source instruction text datasets. The data undergoes filtering process results in 2.2M samples. Finally, these samples are converted to speech instruction data using WhisperSpeech (TTS) and WhisperVQ (speech to semantic tokens), creating the 1.3M pairs of Speech instruction and Text answer.

Novel Approaches and Technical Insights

1. **Tokenized Early-Fusion Approach**:
   - Unlike other models that use separate encoders for speech, Ichigo converts speech to tokens
   - Uses same transformer architecture for both modalities, eliminating need for adaptors
2. **Optimized Embedding Initialization**:
   - Rejected default new token initialization from HuggingFace (resulted in slow convergence)
   - Validated that averaging all embeddings of current vocabulary significantly improved convergence
3. **Transcription Token Experiments**:
   - Rejected using a single special token <|transcribe|> (caused catastrophic forgetting)
   - Validated that using six diverse instruction prompts improved model recovery
4. **Inaudible Input Handling**:
   - Novel approach to generate synthetic "inaudible" inputs by randomizing sound tokens
   - Insight: With 50 sound tokens, the space of possible arrangements (513^50) is mostly inaudible
   - Validated that exposing model to these chaotic arrangements taught it to distinguish audible vs. inaudible
5. **Data Balance Optimization**:
   - Through permutation testing, discovered optimal ratio (70/20/10) for speech instruction, transcription, and text
   - Validated that this balance was critical for maintaining both speech and text capabilities
6. **Latency Improvements**:
   - Achieved 111ms latency to first token (4× faster than cascaded systems, 3× faster than Qwen2-Audio)
   - Maintained lower VRAM usage (19GB vs. 32GB for Qwen2-Audio)

Real-World Inference

In a practical deployment, Ichigo:

1. Receives either speech or text input directly (no need for separate ASR)
2. Processes the input through its unified architecture
3. Generates responses with extremely low latency (111ms to first token)
4. Maintains context across multi-turn conversations
5. Appropriately handles unclear inputs by requesting clarification
6. Can follow system prompts in text while engaging in speech-based conversations
7. Supports real-time use cases via demos (web-based, Hugging Face, and self-hosted options)

Open-Sourced Resources

The Ichigo team has open-sourced:

1. **Model Weights**: Complete model checkpoints
2. **Code**:
   - Training and inference code on GitHub
   - Implementation of tokenized early-fusion approach
3. **Data**:
   - Instruction Speech dataset (2,000 hours of tokenized speech audio paired with text responses)
   - Six different prompts used for transcription data

Qwen2-Audio Technical Report.” arXiv, July 15, 2024.

Chu, Yunfei, Jin Xu, Qian Yang, Haojie Wei, Xipin Wei, Zhifang Guo, Yichong Leng, et al. “Qwen2-Audio Technical Report.” arXiv, July 15, 2024. Qwen2-Audio Technical Report .

Model

Model Details

- **Architecture**: Qwen2-Audio combines an audio encoder with a large language model
  - **Audio Encoder**: Based on Whisperlarge-v3 (Radford et al., 2023)
  - **Language Model**: Qwen-7B (Bai et al., 2023)
  - **Total Parameters**: 8.2B
- **Audio Processing Pipeline**:
  - Audio resampled to 16kHz
  - Converts raw waveform to 128-channel mel-spectrogram
  - Uses 25ms window size and 10ms hop size
  - Incorporates a pooling layer with stride of two to reduce representation length
  - Each output frame corresponds to approximately 40ms of original audio

Complete Training Process

The training follows a three-stage process as illustrated in Figure 2 (mentioned in the paper):

1. **Pre-training Stage**:
   - Replaced hierarchical tags (used in previous Qwen-Audio) with natural language prompts
   - Significantly scaled up the training dataset volume compared to previous models
   - Used paired data (a, x) where a = audio sequences and x = text sequences
   - Training objective: maximize next text token probability conditioned on audio representations
   -

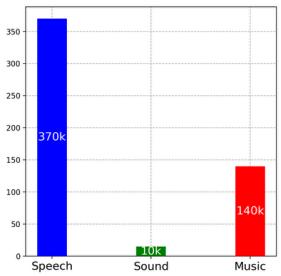


Figure 3: Statistics (hours) of pre-training dataset.

2. **Supervised Fine-tuning (SFT) Stage**:
   - Employed instruction-based fine-tuning to align with human intent
   - Meticulously curated high-quality SFT data with rigorous quality control
   - Jointly trained two interaction modes:
     - **Audio Analysis Mode**: For analyzing diverse audio with instructions via audio/text
     - **Voice Chat Mode**: For natural voice conversations
3. **Direct Preference Optimization (DPO) Stage**:
   - Used triplet data (x, yw, yl) where:
     - x = input sequence with audio
     - yw = human-annotated good response
     - yl = human-annotated bad response
   - Applied the DPO optimization formula as shown in equation (2) to align with human preferences
   - Used reference model (Pref) initialized with the SFT model (Pθ)

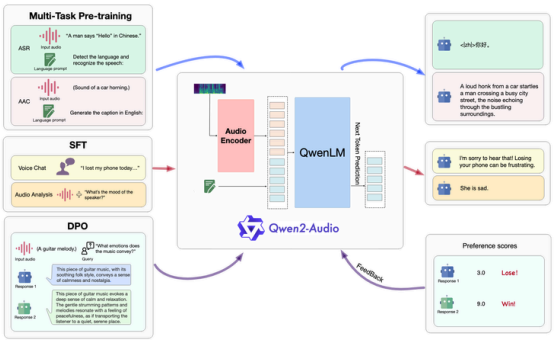


Figure 2: The overview of three-stage training process of Qwen2-Audio.

Novel Contributions

1. **Natural Language Prompts**:
   - Replaced complex hierarchical tags with natural language prompts for different data and tasks
   - Validated that this approach improves generalization ability and instruction-following capability

2. **Seamless Dual-Mode Operation**:
   - Implemented two distinct audio interaction modes (voice chat and audio analysis)
   - Innovatively trained both modes jointly, eliminating the need for mode-switching system prompts
   - The model autonomously determines the appropriate mode based on input
3. **Intelligent Audio Understanding**:
   - Developed capability to comprehend complex audio containing multiple elements simultaneously
   - Can identify command segments within audio containing other sounds and conversations
   - Demonstrated ability to follow voice commands directly
4. **Evaluation Strategy**:
   - Validated performance primarily through AIR-Bench, finding it more representative of real-world use
   - Demonstrated state-of-the-art performance across multiple benchmarks
   - Outperformed Gemini-1.5-pro in audio-centric instruction-following capabilities

Real-World Inference Examples

The paper provides an example of how inference works in a real use case:

- **Mixed Audio Analysis**: If a user inputs an audio clip containing keyboard typing sounds followed by a spoken question "What is this sound?", Qwen2-Audio directly responds with "This is the sound of a keyboard."
- **Mode Determination**: The model automatically detects whether to operate in:
  a. **Audio Analysis Mode**: When users want to analyze audio files (speech, sound, music, mixed audio)
  b. **Voice Chat Mode**: When users engage in conversational dialogue
- **Command Recognition**: The model can discern command segments within complex audio, allowing users to provide instructions through either audio or text
- **Flexible Interaction**: During voice chat, users can seamlessly switch between audio and text input without explicit mode changes

The paper emphasizes that unlike other systems, Qwen2-Audio doesn't require specific system prompts to switch between modes, making the interaction more natural and seamless for users.

While the paper doesn't provide exhaustive details on the specific datasets used at each training stage or the hyperparameters employed during training, it does highlight that the evaluation datasets were "rigorously excluded from the training data to avoid data leakage" and that the model was evaluated on 13 different datasets spanning ASR, S2TT, SER, and VSC tasks.

Audio-Language Models for Audio-Centric Tasks: A Survey. January 25, 2025.

Su, Yi, Jisheng Bai, Qisheng Xu, Kele Xu, and Yong Dou. “Audio-Language Models for Audio-Centric Tasks: A Survey.” arXiv, January 25, 2025. Audio-Language Models for Audio-Centric Tasks: A survey .

This comprehensive survey examines Audio-Language Models (ALMs), which are trained on audio-text data to process, understand, and reason about sounds. Unlike traditional supervised learning approaches that rely on predefined labels, ALMs use natural language as supervision, making them better suited for describing complex real-world audio recordings.

Key Aspects of ALMs

Foundations

- **Architectures**: Four main types: Two Towers (separate encoders with joint embedding space), Two Heads (separate encoders with a language model on top), One Head (unified multimodal encoder), and Cooperated Systems (LLM as planning agent coordinating multiple models)
- **Training Objectives**: Contrastive (bringing matching audio-text pairs closer), generative (audio/text reconstruction or generation), and discriminative (classification/retrieval)
- **Evaluation Methods**: Zero-shot, linear probe, supervised fine-tuning, and instruction-following evaluation

Pre-training Approaches

- **Audio Pre-training**: CNN-based, Transformer-based, and codec-based models
- **Language Pre-training**: Models categorized by parameter scale from small to very large (LLMs)
- **Audio-Language Pre-training**: Models applying contrastive, generative, or discriminative objectives to learn audio-language correlations

Downstream Transfer Methods

- **Task-specific Fine-tuning**: For discriminative tasks (classification, retrieval) and generative tasks (audio generation, source separation, language generation)
- **Multi-task Tuning**: Unified frameworks using sequence-to-sequence modeling, instruction tuning, or in-context learning
- **Audio-Language Agent Systems**: Using LLMs as central coordinators to leverage specialized models for complex tasks

Datasets and Benchmarks

- **Audio-Text Paired Datasets**: Captions, transcriptions, and translations
- **Audio Question Answering Datasets**: Triplets of audio, question, and answer
- **Benchmarks**: Task-specific, cross-task, and instruction benchmarks for standardized evaluation

Challenges and Future Directions

Data Challenges

- Limited diversity in audio sources
- Text annotation quality and scale limitations
- Need for more comprehensive evaluation benchmarks

Model Challenges

- Exploring unified multimodal encoders
- Developing stronger pre-trained audio foundation models
- Enabling continual learning without complete retraining
- Improving data efficiency in training

This survey provides a structured roadmap of ALM development, highlighting how these models are pushing computer audition research toward more human-like hearing capabilities through natural language understanding. The field shows promise for creating more general, flexible, and powerful audio understanding systems.

Recent Advances in Speech Language Models: A Survey. February 6, 2025

Cui, Wenqian, Dianzhi Yu, Xiaoqi Jiao, Ziqiao Meng, Guangyan Zhang, Qichao Wang, Yiwen Guo, and Irwin King. “Recent Advances in Speech Language Models: A Survey.” arXiv, February 6, 2025. Recent Advances in Speech Language Models: A Survey .

This survey provides the first comprehensive overview of Speech Language Models (SpeechLMs), which are end-to-end models designed to process and generate speech directly without converting to and from text.

Background and Motivation

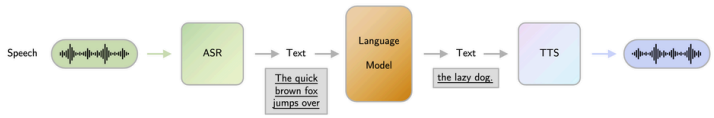
Traditional voice interaction uses a pipeline of "ASR + LLM + TTS" (speech recognition, language model, text-to-speech), which suffers from:

- Information loss during modality conversion (particularly paralinguistic aspects)
- High latency due to the complex pipeline
- Error accumulation across the three stages

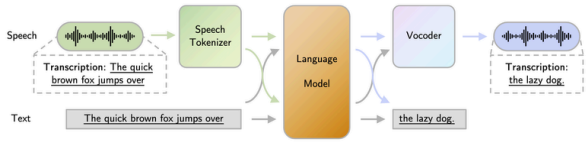
SpeechLMs address these limitations by directly processing speech waveforms into tokens, modeling them autoregressively, and synthesizing them back to speech.

Core Components of SpeechLMs

1. **Speech Tokenizer**: Converts audio waveforms into tokens
   - Semantic understanding tokenizers (HuBERT, wav2vec 2.0)
   - Acoustic generation tokenizers (neural audio codecs)
   - Mixed objective tokenizers (balancing semantic and acoustic aspects)
2. **Language Model**: Processes tokens autoregressively
   - Typically borrows transformer-based architectures from TextLMs
   - Can jointly model text and speech by expanding vocabulary
3. **Token-to-Speech Synthesizer (Vocoder)**: Converts tokens back to waveforms
   - GAN-based vocoders are most common (HiFi-GAN)
   - Can use direct synthesis or input-enhanced approaches



(a) Illustration of the “ASR + LLM + TTS” framework.



(b) Illustration of the architecture of a SpeechLM.

Fig. 1. Architectures of the “ASR + LLM + TTS” framework and a SpeechLM. We emphasize that, for SpeechLM, the same content can be used across both speech and text modalities, meaning that any input modality will yield any output modality of the same results. This intentional repetition of input/output contents in the figure highlights this point.

Training Approaches

1. **Features Modeled**:
   - Discrete tokens (semantic, paralinguistic, acoustic)
   - Continuous representations (spectrograms, neural embeddings)
2. **Training Stages**:
   - Pre-training (cold initialization or continued pre-training from TextLMs)

- Instruction-tuning (teaching models to follow specific task instructions)
- Post-alignment (aligning with human preferences)

3. **Speech Generation Paradigms**:
   - Real-time interaction (handling interruptions, simultaneous speaking)
   - Interactive period recognition (knowing when to speak vs. listen)

Downstream Applications

1. **Semantic-Related**: Spoken dialogue, speech translation, ASR, TTS, intent classification
2. **Speaker-Related**: Speaker identification, verification, diarization, voice cloning
3. **Paralinguistic**: Emotion recognition, speech separation, style/tone-specific generation

Evaluation Methods

1. **Automatic Evaluation**: Representation quality, linguistic accuracy, paralinguistic preservation, generation quality, real-time interaction, downstream tasks
2. **Human Evaluation**: Mean Opinion Scores for naturalness, prosody, and timbre similarity

Challenges and Future Directions

1. Better understanding of component trade-offs
2. End-to-end training approaches
3. Real-time speech generation
4. Safety risks (toxicity, privacy concerns)
5. Performance on rare languages with limited text resources

The paper highlights SpeechLMs as a promising alternative to the traditional pipeline approach, offering more natural, efficient, and expressive voice interactions by preserving both semantic content and paralinguistic information.
