RIVA: AN OPEN RECIPE FOR AGENTIC FULL-DUPLEX SPEECH MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Full-duplex speech language models let users interact with an AI assistant as they would with another person. Existing open models, however, struggle to combine native interaction, in which the model itself decides when to listen, speak, or yield, with the language and agentic capabilities of a strong large language model (LLM). Tool use is also typically handled outside the conversational stream, through separate action channels or backend models. We introduce Riva,[1] an open full-duplex speech model in which a pretrained LLM **listens, speaks, and acts within a single stream**. Every 480 ms, the model processes incoming speech, decides whether to listen or speak, generates text that a lightweight talker renders as speech, and can emit tool calls in its native format while the conversation continues. Three components make this possible under real speech: a pre-training recipe that reduces the speech–text modality gap; Riva-Dialog, approximately 500K skilled, speech-native conversations spanning 13K hours, complemented by Riva-Entity for named entities and structured identifiers; and reinforcement learning over short decision windows, combining GRPO for interaction-level decisions with on-policy self-distillation for exact tool arguments. Riva is competitive with or outperforms prior open full-duplex models across six benchmarks, including Audio MultiChallenge (18.4 vs. 12.8 APR) and Full-Duplex-Bench v3 (42.0 vs. 35.5 P@1), and achieves an average P@1 of 28.6 on multi-minute, multi-tool customer-service interactions in τ^2-Voice.

1 INTRODUCTION

Human conversation is full-duplex: people listen while they speak, backchannel, overlap, and interrupt, and the floor passes between speakers within fractions of a second (Sacks et al., 1974; Schegloff, 2000; Stivers et al., 2009; Levinson, 2016). Full-duplex speech language models (FD-SLMs) aim to bring this form of interaction to AI assistants, so that users can talk with them as naturally as with another person (Défossez et al., 2024; Chen & Yu, 2025). At the same time, what users expect of such assistants increasingly goes beyond conversation. Large language models (LLMs) now follow complex instructions, track goals across many turns, and call tools to act on the world (Achiam et al., 2023; Schick et al., 2023; Xi et al., 2025). A voice agent must use these capabilities in the middle of a conversation: keep talking while a tool call is in progress, stop when the user cuts in to correct a detail, and pass exactly the name or code the user spoke to the tool. It therefore needs both *native interaction*, in which the model itself decides when to listen, speak, or yield rather than an external voice-activity detector or controller, and the agentic capabilities of a strong LLM.

Existing full-duplex speech models obtain one of these at the expense of the other. Models that generate speech tokens in parallel streams, such as Moshi (Défossez et al., 2024) and its successors (Roy et al., 2026; Hsiao et al., 2026; Tyagi et al., 2026), make turn-taking native but retain little of an LLM's language capabilities: Moshi scores 2.15 on VoiceBench AlpacaEval (Chen et al., 2026), compared with 3.73 for our model. Models that keep a pretrained LLM and add speech through lightweight adaptation (Wang et al., 2025; Yu et al., 2025b; Cui et al., 2026; Kim et al., 2026) preserve more of its capabilities, but do not call tools during full-duplex conversation. Where open full-duplex agents do act, the action is moved out of the conversational stream, onto a separate function head (Balam et al., 2026) or action channel (Zhang et al., 2026), or to a backend model

[1] We will release the training and inference code, model checkpoints, and data upon acceptance.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

that supplies knowledge, reasoning, and tool calls on the conversational model's behalf (Kuroki et al., 2026; Huang et al., 2026). This separates what the agent says from what it does. Proprietary systems (OpenAI, 2026; Google DeepMind, 2026) combine interaction, capability, and tool use, but they are closed. To our knowledge, *no open model listens, speaks, and acts within a single stream*, and no open full-duplex model has been evaluated on long-horizon agentic tasks such as τ^2-Voice (Ray et al., 2026).

We present Riva, an open full-duplex speech model built on one idea: a single autoregressive stream in which a pretrained LLM listens, speaks, and acts. We adopt the thinker–talker design (Xu et al., 2025) and the time-division multiplexing of recent full-duplex models (Yu et al., 2025b; Cui et al., 2026). The thinker consumes streaming speech in 480 ms micro-turns, decides at each micro-turn whether to listen or speak, and emits text, which a lightweight talker renders as speech (Figure 1). We depart from prior work in how the model acts: tool calls are part of the same stream. The thinker emits them in its backbone's native format, spread across micro-turns, and keeps talking while a call executes and its result returns (Sec. 2.1). Because the thinker generates text throughout, the backbone's language and tool-use capabilities carry over to speech rather than being approximated by a separate channel or delegated to another model.

A single stream is not sufficient on its own. Making it work under real speech requires overcoming three obstacles, which we address in turn. *Representation.* Capabilities that the backbone has under text input degrade when the same content arrives as speech (Chen et al., 2026; Cuervo et al., 2026), and at our scale of about 25K hours, streaming speech recognition as the only pre-training objective leaves much of this gap open. We add an audio-text continuation objective that places speech and text in the same predictive context, which yields the largest gain among our pre-training objectives (Sec. 2.2). *Data.* Rendering instruction data as speech (Zhang et al., 2023; Fang et al., 2025) preserves the tasks but not the dynamics of conversation, whereas real conversations (Cieri et al., 2004; Reece et al., 2023) have the dynamics but not the skills. We build Riva-Dialog with a data engine grounded in Conversation Analysis, which produces skilled conversations with interruptions, repairs, backchannels, and tool use, including long customer-service calls. We complement it with Riva-Entity, because for a spoken agent a misheard name or identifier becomes a wrong tool argument (Sec. 3.1). *Decisions.* After supervised fine-tuning, many of the remaining errors concern *when* the model acts and *exactly what* it heard, which next-token supervision on synthetic conversations does not teach. For example, interruptions in synthetic training data are represented in text only, without overlapping speech, so the model learns to stop mid-sentence but not to yield *because* the user starts speaking. We therefore apply reinforcement learning to short decision windows around such events, with verifiable rewards for timing and for exact tool arguments. We optimize timing with GRPO (Shao et al., 2024), and for identifiers we add on-policy self-distillation (Zhao et al., 2026), in which the same model, given the exact transcript as privileged context, provides a dense signal where sparse rewards give none (Sec. 3.2, App. E).

Across six benchmarks, Riva is competitive with or outperforms prior open full-duplex models across general knowledge, alignment, interactivity, conversational intelligence, tool use, and long-horizon agentic interaction (Table 1). In particular, it achieves 18.4 APR on Audio MultiChallenge, 42.0 P@1 on Full-Duplex-Bench v3, and an average P@1 of 28.6 across the Airline, Retail, and Telecom domains of τ^2-Voice. To our knowledge, Riva is the first open full-duplex model evaluated on τ^2-Voice. Our contributions are:

- **A single stream to listen, speak, and act.** We present Riva, a full-duplex speech model whose thinker emits turn-taking decisions, response text, and tool calls in its backbone's native format within one autoregressive stream, while continuing the conversation as tools execute. It achieves 42.0 P@1 on Full-Duplex-Bench v3, outperforming prior open full-duplex models, and runs in real time on a single A100 GPU with approximately 510 ms median conversational response latency (Sec. 2.1, App. A.4).
- **Speech–text aligned pre-training.** We introduce a streaming pre-training mixture of transcription, audio-text continuation, and spoken question answering that reduces the speech–text modality gap, with audio-text continuation providing the largest gain among the pre-training objectives (Sec. 2.2).
- **Riva-Dialog and Riva-Entity.** We introduce a data engine grounded in Conversation Analysis that produces skilled, interactive, and agentic spoken conversations, together with targeted supervision for named entities and structured identifiers. Riva-Entity improves Full-

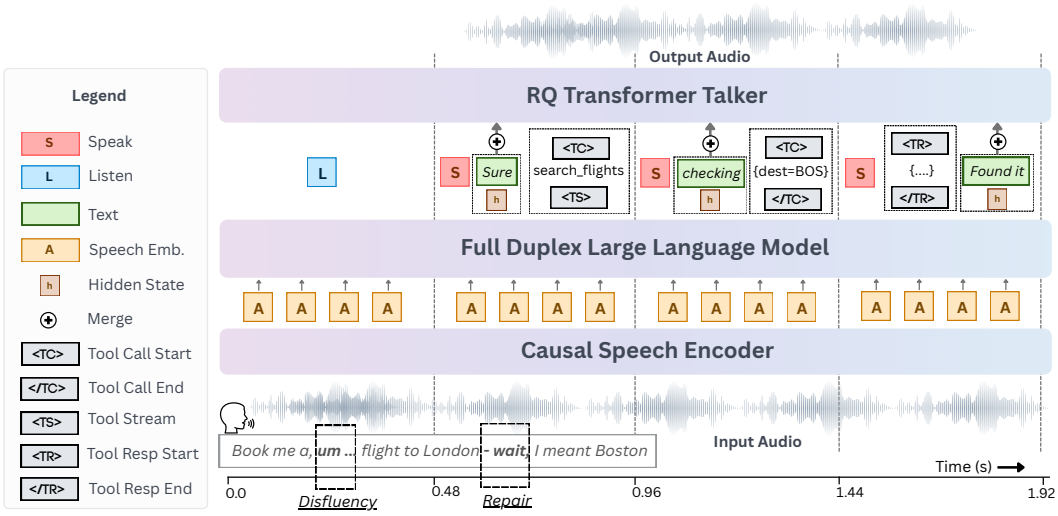


Figure 1: **Full-duplex speech interaction with streaming tool use.** A causal speech encoder converts the incoming real-time speech stream into continuous representations, which are processed in 480 ms micro-turns. At each micro-turn, the model emits a turn-taking decision (LISTEN or SPEAK) and interleaved text and tool-call tokens in a single stream. Tool calls may span multiple micro-turns, and tool responses are consumed inline while the talker generates speech from emitted text and LLM hidden states.

Duplex-Bench v3 argument accuracy from 41.0 to 53.6 and raises average τ^2-Voice P@1 from 14.9 to 28.6 (Sec. 3.1, Sec. 4.2).

- **Reinforcement learning for full-duplex decisions.** We introduce RL over short decision windows with verifiable rewards for turn-taking and exact tool arguments, combining GRPO with privileged on-policy self-distillation for identifier-heavy tool calls (Sec. 3.2, App. E).

2 Building Real-time Speech Understanding & Generation

This stage builds real-time speech understanding and generation into a pretrained language model. Doing so raises two challenges. First, what architectural design can extend the model to real-time speech interaction with native turn-taking, while also allowing the resulting model to reuse the backbone's native tool-calling capabilities? Second, how can we adapt the model to speech without introducing a large modality gap that erodes the capabilities learned in text pretraining?

2.1 Architecture for Full-Duplex Speech Interaction

Speech Understanding and Generation. We build speech understanding and generation around a thinker–talker design, where the thinker consumes speech and produces intermediate text outputs, while a separate lightweight talker generates the corresponding speech. Rather than representing speech directly as discrete tokens in the language-model backbone, we provide the thinker with continuous speech representations and leave acoustic generation to the talker. This offers two key advantages. First, it avoids the high token rate of discrete speech representations, substantially reducing the sequence length processed by the backbone and making long-form conversations more tractable. Second, the thinker's autoregressive output remains in text, keeping generation closer to the distribution on which the backbone's language capabilities were learned.

Full-Duplex Design. To model interaction natively, we adopt a time-division multiplexing (TDM) scheme that interleaves incoming speech and model outputs over fixed-duration micro-turns. At each micro-turn, the thinker consumes the newly arrived audio and emits a state token, such as `LISTEN` or `SPEAK`, followed by text when speaking. In this way, turn-taking becomes part of

the model's autoregressive prediction process rather than a decision delegated to an external voice activity detector or interaction controller. We formalize this stream design in Appendix A.1.

Native Tool Calling. Our design allows the backbone's native tool-call format to be emitted directly within the duplex stream. Because each micro-turn has a bounded generation budget to preserve real-time operation, tool calls often extend beyond a single micro-turn. We therefore introduce a tool stream, which serializes a tool call across consecutive micro-turns.

The tool stream can be interleaved with short floor-holding utterances,[2] allowing the model to remain conversationally active while constructing and executing the call. Once the tool result becomes available, it is consumed in a subsequent micro-turn and the conversation continues normally.

Thinker Design. The thinker follows the encoder–adapter–LLM architecture of large audio-language models (Ghosh et al., 2024; 2025), which achieve strong audio understanding and reasoning (Sakshi et al., 2025). We initialize its causal audio encoder from Voxtral Realtime (Mistral-AI et al., 2026) and use Qwen3-8B (Yang et al., 2025) as the language-model backbone. A lightweight MLP adapter projects and temporally downsamples the encoder representations into the LLM input space, reducing the sequence length required for long-context speech interaction. Operating over continuous speech representations avoids introducing a dense acoustic-token sequence while preserving the backbone's text-autoregressive interface.

Talker Design. The talker must generate speech causally as intermediate text arrives from the thinker. We use a lightweight RQ-Transformer over Mimi codec tokens (Défossez et al., 2024), with a temporal transformer initialized from the first half of Qwen3-0.6B and a depth transformer predicting the residual codebooks within each frame. We additionally condition the talker on thinker hidden states to provide linguistic and conversational context. Following Delayed Streams Modeling (Zeghidour et al., 2025), an action head predicts at each frame whether to advance the text stream, allowing speech timing to be decoupled from when the thinker generates each token.

Real-Time Inference. Full-duplex interaction requires a real-time factor below one, so that audio is processed at least as fast as it arrives. While conventional LLM tasks can tolerate slower offline inference, a real-time speech model does not have this flexibility. We therefore treat inference efficiency as an important part of the model design and build an optimized implementation on top of vLLM-Omni (Yin et al., 2026). Further implementation details are provided in Appendix A.4.

2.2 Pre-training for Streaming Speech

We pre-train the model in two stages, first training the thinker for streaming speech understanding and then freezing it while training the talker for causal speech generation.

Thinker Pre-training. Existing FD-SLM pre-training commonly relies on large-scale full-duplex conversational data (Kim et al., 2026), which is scarce, or on streaming speech recognition at million-hour scale (Cui et al., 2026). At our scale of roughly 25K hours, streaming ASR alone leaves a substantial speech–text modality gap. We therefore pre-train the thinker on three objectives targeting streaming recognition, cross-modal alignment, and spoken response generation.

- *Streaming Transcription.* We use word-level forced alignments to construct streaming targets, such that each micro-turn emits the words completed within that unit, or a `PAD` token otherwise.
- *Audio-Text Continuation.* Transcription alone does not require speech and text to function as interchangeable predictive context. We therefore additionally train on cross-modal continuation, where part of a text sequence is replaced by its corresponding speech (Nguyen et al., 2025; Mistral-AI et al., 2026) and the model predicts the remaining text. We adapt this objective to our streaming setting by aligning the substituted audio to the thinker's shared micro-turn clock, so that continuation runs causally over the same interleaved speech–text stream as at inference.
- *Spoken Question Answering.* We train the thinker to directly answer questions posed in speech, moving beyond recognition and alignment toward response generation from spoken input.

Riva-Entity. We find that a persistent failure mode in speech-based tool use is the mis-recognition of named entities, structured identifiers, and spelled-out sequences. These errors matter most in

[2] Only the number of these utterances depends on the tool's latency, never their wording, so the model cannot learn to predict how long a call will take from its own first words.

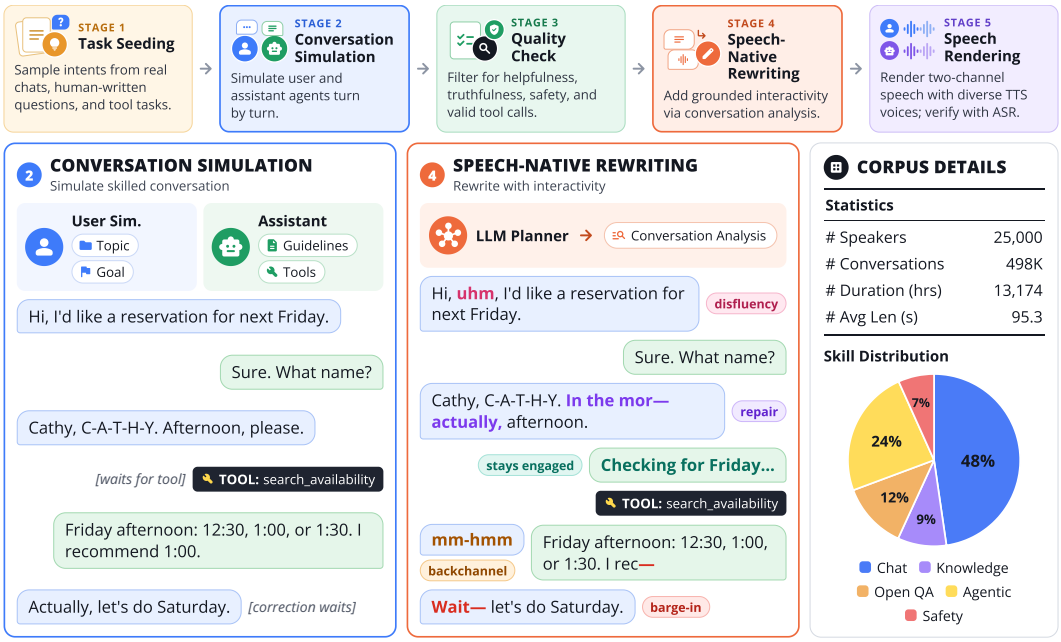


Figure 2: **Constructing Riva-Dialog for speech-native interaction.** Our five-stage data engine seeds tasks, simulates multi-turn conversations, filters them for quality, rewrites them with speech-native interaction phenomena, and finally renders them as duplex speech. The lower panels illustrate conversation simulation and speech-native rewriting in more detail, while the right panel summarizes the resulting corpus composition and statistics.

agentic settings, where such spans often become exact tool-call arguments. We therefore curate a synthetic corpus of entity-rich and structured utterances. We source named entities from public resources such as Wikipedia, synthetically generate identifiers such as emails and user IDs, and place them in varied spoken contexts, including spelled forms. The resulting samples are rendered with TTS using a diverse bank of voices. Further generation details are provided in Appendix D, and Sec. 4.2 analyzes their effect on argument accuracy and tool-use success.

Training Details. For thinker pre-training, streaming transcription and audio-text continuation are constructed from Loquacious (Parcollet et al., 2025), augmented with our named-entity-rich speech data. Spoken question answering uses a mixture of single-turn spoken-QA datasets. We sample transcription, audio-text continuation, and spoken QA with approximate proportions of 48/39/13, and use an audio-text replacement ratio of $r = 0.3$. The audio encoder stays frozen, while the adapter and the Qwen3-8B LoRA parameters are trained jointly from the start. We use a higher learning rate for the newly initialized adapter (3×10^{-4}) than for the LLM LoRA parameters (3×10^{-5}), with short warmups of 400 and 1,000 steps respectively, avoiding a separate adapter-only alignment stage. We train for 14.5K steps using AdamW with cosine decay.

For talker pre-training, we use roughly 30K hours of speech from Emilia (He et al., 2024). The thinker remains frozen while the talker is trained for approximately 116K steps on 8 H100 GPUs using AdamW with a learning rate of 1×10^{-4}. We use two-token look-ahead with look-ahead dropout. For both stages, we pack training examples to maximize sequence utilization.

3 Post-Training Conversational & Agentic Capabilities

3.1 Curating Speech-Native Instruction Data

Riva-Dialog. High-quality post-training data is central to eliciting the capabilities of pretrained models (Lambert et al., 2024; Deitke et al., 2025). For full-duplex speech models, this requires both general language capabilities, such as instruction following and tool use, and speech-native conversational behavior. Existing spoken instruction data (Xie & Wu, 2024; Zhang et al., 2023) is largely

derived from text instruction-tuning corpora and provides little supervision for interruptions, repairs, confirmations, and backchannels. Conversely, natural conversational corpora such as Fisher (Cieri et al., 2004) and CANDOR (Reece et al., 2023) capture these phenomena but provide little supervision for broader assistant capabilities. We therefore curate Riva-Dialog, a post-training corpus that combines skilled instruction data with speech-native conversational interaction (Fig. 2).

We construct Riva-Dialog using a five-stage data engine. **1) Task Seeding.** We seed each conversation with an underlying user intent or task drawn from real human–AI interactions and human-authored questions, supplemented with synthetic scenarios for capabilities such as tool use where suitable human-authored data is scarce. **2) Conversation Simulation.** Given a seed task, we simulate separate user and assistant agents turn by turn, rather than generating an entire multi-turn dialogue in a single pass, to encourage more varied and realistic interactions. **3) Quality Check.** We filter the resulting conversations for quality, using an LLM judge to evaluate helpfulness, truthfulness, and safety, while executing tool calls to verify that actions are valid. **4) Speech-Native Rewriting.** In preliminary experiments, we find that simply prompting an LLM to add interactive behaviors often produces surface-level insertions, such as backchannels or interruptions, without grounding them in the surrounding conversational context. We therefore draw on principles from Conversation Analysis, which characterizes how spoken interaction is organized through turn-taking, repair, listener feedback, and overlapping speech. For each target phenomenon, we construct a behavior-specific prompt that describes its different forms, the conversational cues that make it appropriate, and in-context examples of when and how it should occur. This encourages the model to introduce interactional phenomena where they are contextually warranted, rather than merely inserting surface-level markers.**5) Speech Rendering.** Finally, we render the rewritten conversations as two-channel duplex audio using diverse voice-cloning and TTS systems, and filter synthesis errors using ASR-based verification. The final dataset contains approximately 500K conversations, spanning roughly 13K hours of dual-channel conversational speech.

Data Generation Details. We use Qwen3-8B to simulate the user and assistant for most conversations, with Qwen3-30B-A3B serving as the quality judge. For tool-use conversations, where we find that stronger teacher models improve trajectory quality, we use Qwen3-30B-A3B, Kimi-K2.6 and GLM-5.2. For speech rendering, we use multiple open-source TTS systems and filter synthesis errors using Qwen3-ASR and Parakeet. Further details on dataset composition, skill-specific generation, and filtering are provided in Appendix C.

3.2 Training Methodology

Thinker SFT. We first fine-tune the thinker on Riva-Dialog to acquire the conversational, interactive, and agentic capabilities required for full-duplex interaction.

Reinforcement Learning. While SFT provides a strong base, we find that several remaining errors in interaction and tool use are naturally suited to reinforcement learning. In interaction, the model must make local decisions about when to speak, yield, or continue listening. In tool use, failures often arise from exact argument generation, particularly for spoken identifiers and structured values. These outcomes are directly measurable and provide clear reward signals.

Rather than rolling out full conversations, we therefore train on short *decision windows* around the relevant event. A teacher-forced prefix establishes the conversational context, after which the policy generates several micro-turns that are scored with rule-based rewards. Interaction windows reward appropriate behavior around interruptions, backchannels, pauses, background speech, noise,[3] and turn ends, while tool-use windows reward the correct tool and exact arguments.

We optimize these windows using GRPO (Shao et al., 2024), with decoupled clipping and dynamic sampling (Yu et al., 2025a) and a bounded KL penalty to the SFT model. For identifier-heavy tool calls, however, exact-match rewards are often sparse, leaving many rollout groups without a useful gradient. We therefore additionally use on-policy self-distillation (Zhao et al., 2026). The frozen SFT model receives the ground-truth transcript of the caller's speech as privileged context and scores

[3]We also add a colored-noise floor of -80 to -55 dBFS to synthesized user audio. Synthesized speech is otherwise separated by digital silence, whereas real recordings end in room tone (about -73 dBFS), which a model trained only on synthetic audio treats as ongoing speech and therefore does not respond to.

Table 1: Main results. Higher is better (↑) unless marked ↓. Among open models, the best result is in **bold** and the second best is underlined; proprietary systems (grey) are shown for reference and are not ranked. "–" marks models without a tool-call interface. AdvBench reports refusal rate. VoiceBench results cover four of eight subsets.

Benchmark	Gemini Live	GPT Live	Kyutai Moshi	Persona Plex	MiniCPM -o 4.5	Raon SpeechChat	NVIDIA VoiceChat	**Riva 8B**
General Knowledge — OpenAudioBench (Acc ↑)								
Llama Questions	82.0	85.7	50.7	56.3	71.8	57.0	59.0	**75.7**
Web Questions	89.1	89.8	19.3	21.9	53.8	35.4	36.0	**61.5**
TriviaQA	80.9	81.6	23.1	36.6	**60.2**	43.6	44.3	57.5
Alignment — VoiceBench								
AlpacaEval	4.64	4.79	2.15	2.45	**3.80**	2.40	3.16	3.73
CommonEval	4.33	4.26	1.94	2.29	3.22	2.08	2.97	**3.42**
IFEval	21.8	48.2	19.4	17.2	20.1	14.7	16.2	**21.4**
AdvBench	97.1	98.5	68.2	5.3	**98.0**	96.0	97.5	97.5
Multi-Turn Interactivity — Full-Duplex Bench v2 (1–5 ↑)								
TT Fluency	3.55	3.50	3.27	3.70	3.98	3.55	3.44	**4.01**
Instr. Foll.	2.99	3.24	2.53	3.16	**3.53**	3.04	2.92	3.49
Task Metric	3.45	3.86	2.25	3.11	3.24	2.94	2.97	**4.20**
Conversational Intelligence — Audio MultiChallenge (% ↑)								
APR	24.1	-	4.0	6.6	12.8	6.3	6.0	**18.4**
ARS	54.3	-	14.3	20.6	45.0	16.5	17.1	**54.8**
Tool Use — Full-Duplex Bench v3								
P@1	54.0	41.0	–	–	–	–	35.5	**42.0**
Arg. Acc.	79.6	48.0	–	–	–	–	**61.7**	53.6
Resp. Quality	76.9	64.0	–	–	–	–	65.0	**69.0**
Agentic Tasks — τ^2-Voice (P@1 % ↑)								
Airline	80.0	72.0	–	–	–	–	0.0	**41.0**
Retail	63.1	52.0	–	–	–	–	0.0	**15.0**
Telecom	78.0	80.0	–	–	–	–	0.0	**29.8**

the student's generated tokens. Its clipped log-ratio is added to the group-normalized advantage,

$$A_t = A^{\text{env}} + \lambda \, \text{clip}\left(\log \pi_{\text{T}}(y_t) - \log \pi_\theta(y_t), \pm 8\right), \qquad \lambda = 0.5. \tag{1}$$

We apply this auxiliary objective only to identifier windows, while timing decisions are optimized with GRPO alone. Full reward definitions and objective details are provided in Appendix E.2.

Thinker Training Details. For thinker SFT, we update only the LoRA parameters of the Qwen3-8B backbone while keeping the audio encoder and adapter frozen. We train on a mixture of 90% multi-turn conversational data, 5% streaming ASR, and 5% audio-text continuation, with online noise and reverberation augmentation applied to the input speech. We train for 4,881 steps with a global batch size of 32 using AdamW, a learning rate of 2×10^{-5}, 150 warmup steps, cosine decay, and a maximum sequence length of 32K tokens.

RL again updates only the thinker's LoRA parameters. We train for 150 steps with a learning rate of 1×10^{-6} and groups of 8 rollouts. The first 100 steps use GRPO across all decision windows, followed by 50 steps in which on-policy self-distillation is applied to identifier windows. Further optimization, augmentation, and reward details are provided in Appendix E.2.

Talker SFT. After thinker post-training, we freeze the thinker and fine-tune the talker on synthetic assistant speech from Riva-Dialog. This adapts speech generation to the conversational patterns encountered during full-duplex interaction, including short acknowledgements, interrupted turns, and speech produced around tool execution.

Table 2: Ablations. **Left:** effect of pretraining tasks. **Middle:** effect of Riva-Entity data. **Right:** effect of SFT data on full-duplex behavior (↑: higher is better).

Pretraining tasks	CommonEval	LLaMA Q.
ASR	2.58	65.3
+ Spk. QA	2.82	73.5
+ A2T Cont.	**3.42**	**76.0**

Riva-Entity	FDBv3		τ^2-Voice
	P@1	Arg. Acc.	P@1
without	33.0	41.0	14.87
with	**42.0**	**53.6**	**28.60**

SFT data	Backchannel	Interruption
	C_{resume} ↑	C_{respond} ↑
VA400K	72%	11%
Riva-Dialog	**95%**	**65%**

4 RESULTS

4.1 EXPERIMENTAL SETUP

Baselines. We compare Riva against recent open-source full-duplex speech language models, including Moshi (Défossez et al., 2024), PersonaPlex (Roy et al., 2026), MiniCPM-o 4.5 (Cui et al., 2026), RaonSpeechChat (Kim et al., 2026), and Nemotron VoiceChat (Balam et al., 2026). We additionally report results for proprietary full-duplex systems, including Gemini 3.8 Live (Google DeepMind, 2026) and GPT Live (OpenAI, 2026).

Benchmarks. We evaluate Riva on VoiceBench (Chen et al., 2026) (free-form tasks only), OpenAudioBench (Li et al., 2025), Full-Duplex-Bench V1–V3 (Lin et al., 2025; 2026c;b;a), AudioMultiChallenge (Gosai et al., 2026), and τ^2-Voice (Ray et al., 2026).

4.2 MAIN RESULTS

General Knowledge. Speech adaptation can degrade knowledge inherited from the underlying LLM (Défossez et al., 2024). On OpenAudioBench, Riva outperforms or is competitive with other open FD-SLMs across the evaluated knowledge tasks (Table 1), indicating that it largely preserves the backbone's language capabilities under speech input.

Alignment. We evaluate response quality, instruction following, and safety using VoiceBench (Chen et al., 2026). In Table 1, Riva is best or second best among open FD-SLMs on every subset and is competitive with MiniCPM-o 4.5, indicating a much smaller speech–text modality gap.

Multi-Turn Interactivity. Full-duplex interaction must be learned during speech training. On Full-Duplex Bench v2 (Lin et al., 2026b), Riva leads the open FD-SLMs in turn-taking fluency and task performance and is competitive with MiniCPM-o 4.5 in instruction following (Table 1), showing that it sustains natural turn-taking without sacrificing task performance.

Conversational Intelligence. Audio MultiChallenge (Gosai et al., 2026) evaluates retention, coherence, and natural spoken phenomena such as repairs and backtracking across multi-turn conversations. Riva substantially outperforms the other open FD-SLMs on both APR and ARS (Table 1).

Tool Use. We evaluate native tool use on Full-Duplex Bench v3, which tests function calling under pauses, hesitations, and self-corrections. Riva achieves 42.0 P@1 and 69.0 response quality, the highest among open full-duplex models (Table 1).

Agentic Evaluation. Finally, τ^2-Voice tests whether these capabilities compose over long-horizon interactions requiring policy following, goal tracking, and sequential tool use.[4] Riva achieves Pass@1 scores of 41.0, 15.0, and 29.8 on Airline, Retail, and Telecom, respectively, averaging 28.6 across the three domains (Table 1).

Impact of Pre-training Tasks. We ablate the thinker pre-training objectives while holding the data and training budget fixed. As shown in Table 2 (left), spoken QA improves downstream response quality, while adding audio-to-text continuation provides the largest gains.

Impact of Riva-Dialog. We compare Riva-Dialog with VoiceAssistant-400K on FDBv1.5. As shown in Table 2 (right), Riva-Dialog substantially improves both backchannel recovery and interruption handling, highlighting the value of speech-native interaction supervision.

[4] In our airline and retail evaluation runs, a τ^2-Voice call contains a median of 3.1 minutes of audio (6.3 minutes at the 90th percentile), with a median of 7 user turns and 5 tool calls.

Impact of Riva-Entity. We ablate Riva-Entity while keeping the remaining training setup fixed. As shown in Table 2 (middle), entity supervision improves FDBv3 argument accuracy and task pass rate, with corresponding gains on the more challenging τ^2-Voice benchmark.

5 Related Work

Full-Duplex Speech Language Models. Early full-duplex speech models represented conversation directly as discrete speech-token streams within the language-model backbone (Nguyen et al., 2023; Veluri et al., 2024; Zhang et al., 2025b). While this makes interaction native to the model, it can degrade the language capabilities of the underlying LLM. Prior work has explored mechanisms to mitigate this, such as the text-based inner monologue used by Moshi (Défossez et al., 2024). An alternative line instead preserves the LLM through lightweight speech adaptation (Xie & Wu, 2024; Wang et al., 2025), reducing degradation of its language capabilities, but often requiring additional mechanisms to support real-time conversational interaction. A growing line of work explores time-division multiplexing (TDM), which interleaves input and output in short time intervals and makes turn-taking part of the model's autoregressive prediction process (Cui et al., 2026; Kim et al., 2026; Yu et al., 2025b). We adopt this TDM-based approach as the basis for our full-duplex architecture.

Tool Use in Full-Duplex Speech Language Models. Earlier speech systems support tool use only at turn boundaries (Xu et al., 2025), which is poorly suited to FD-SLMs where tool use must be integrated into a continuously unfolding conversation. One approach is to delegate reasoning and tool use to a backend LLM while a frontend speech model maintains the interaction. KAME (Kuroki et al., 2026) follows this tandem design, while DuplexOmni (Huang et al., 2026) similarly separates interaction from a backend thinking layer. An alternative is native tool calling, where the FD-SLM itself manages tool invocation and tool results. DuplexSLA (Zhang et al., 2026) uses a parallel action channel, while NemotronLabs VoiceChat (Balam et al., 2026) uses a separate output stream for structured function calls. Our model instead emits tool calls in the backbone's native format within the duplex stream itself.

Speech-Native Post-Training Data. Prior work has shown that post-training data can substantially improve instruction following, reasoning, and other downstream capabilities (Lambert et al., 2024; Deitke et al., 2025; Ghosh et al., 2025). Existing spoken instruction datasets largely adapt turn-based assistant data to speech (Xie & Wu, 2024), providing strong supervision for language capabilities but little for speech-native behaviors such as backchannels, interruptions, and repairs. Real conversational corpora contain these dynamics (Reece et al., 2023; Cieri et al., 2004; Godfrey et al., 1992; Lee et al., 2025), but provide little supervision for skilled assistant behavior. Public speech tool-calling data is also limited, and largely evaluation-focused or domain-specific.

6 Conclusion

We introduce Riva, an open full-duplex speech language model that brings together native interaction, strong language capabilities, and tool use within a single model. We further introduce Riva-Dialog and Riva-Entity to support speech-native conversational and agentic post-training. Across a broad evaluation suite, Riva demonstrates that these capabilities can be combined in an open model at reproducible scale, including long-horizon agentic interaction on τ^2-Voice. We hope this work provides a foundation for building more capable and interactive open speech agents.

AI Use Statement

We used generative AI tools to polish the writing of this paper, to support the discovery of related work, and to generate synthetic training data. The use of generative models for synthetic data generation is described in the main paper and Appendix C. All final manuscript content, citations, experimental results, and analyses were reviewed and verified by the authors.

Reproducibility Statement

We provide detailed descriptions of the model architecture, training objectives, and data-construction pipeline in the main paper and appendices. Training configurations, including optimization hyperparameters and compute settings, are reported in Sec. 3.2, while Appendix C describes dataset construction and filtering and Appendix E.2 specifies the reinforcement-learning rewards and objectives. We will release the training and inference code, model checkpoints, and data upon acceptance.

Ethics Statement

Our work develops a real-time speech model capable of open-ended conversation and tool use, which introduces potential risks including harmful content generation, impersonation, privacy concerns, and misuse of automated actions. We mitigate some of these risks through safety-focused post-training data and evaluations of refusal behavior, but these measures do not eliminate all potential misuse. Our training corpus combines public datasets and synthetically generated conversations and speech; further information about data sources and generation is provided in Appendix C. Synthetic speech is used for research and model training rather than to reproduce the identity of specific individuals. Tool-use evaluations are conducted in simulated environments rather than against real user accounts or external services. We encourage deployment with appropriate safeguards, access controls, and monitoring for applications involving consequential actions.

References

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

Jagadeesh Balam, Travis Bartley, Edresson Casanova, Sanjay Chauhan, Chen Chen, Zhehuai Chen, Zijia Chen, Francesco Ciannella, Slyne Deng, Mikyas Desta, Harishchandra Dubey, Slim Essid, Nourchene Ferchichi, Boris Ginsburg, Mariana Graterol Fuenmayor, Negar Habibi, Kevin Hu, Anand Joseph, Viraj Karandikar, Myungjong Kim, Viacheslav Klimkov, Seelan Lakshmi Narasimhan, Lily Lee, Jason Li, Eileen Long, Ameya Mahabaleshwarkar, Aditya Malte, Adi Margolin, Sasha Meister, Valentin Mendelev, Oluwatobi Olabiyi, Ankita Pasad, Yifan Peng, Elena Rastorgueva, Jayda Ritchie, Jason Roche, Nikhil Srihari, Yuanhang Su, Yoshi Suhara, Viet Anh Trinh, Jinhan Wang, Piotr Zelasko, Hui Wang, Puhui Meng, Chaosen Zhang, Yunsheng Liu, Shawn Wang, Wenjing Li, and Zhonglei He. Nemotronlabs voicechat: An open full-duplex speech-to-speech model with tool calling capabilities, 2026. URL https://arxiv.org/abs/2609.21967.

Victor Barres, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. *arXiv preprint arXiv:2506.07982*, 2025.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pp. 1533–1544, Seattle, Washington, USA, 2013. Association for Computational Linguistics.

Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T. Tan, and Haizhou Li. VoiceBench: Benchmarking LLM-based voice assistants. *Transactions of the Association for Computational Linguistics*, 14:378–398, 2026. doi: 10.1162/tacl.a.628. URL https://aclanthology.org/2026.tacl-1.18/.

Yuxuan Chen and Haoyuan Yu. From turn-taking to synchronous dialogue: A survey of full-duplex spoken language models. *arXiv preprint arXiv:2509.14515*, 2025.

Christopher Cieri, David Miller, and Kevin Walker. The fisher corpus: A resource for the next generations of speech-to-text. In *LREC*, volume 4, pp. 69–71, 2004.

Santiago Cuervo, Skyler Seto, Maureen de Seyssel, Richard Bai, Zijin Gu, Tatiana Likhomanenko, Navdeep Jaitly, and Zakaria Aldeneh. Closing the gap between text and speech understanding in llms. In *International Conference on Learning Representations*, volume 2026, pp. 146313–146335, 2026.

Junbo Cui, Bokai Xu, Chongyi Wang, Tianyu Yu, Weiyue Sun, Yingjing Xu, Tianran Wang, Zhihui He, Wenshuo Ma, Tianchi Cai, et al. Minicpm-o 4.5: Towards real-time full-duplex omni-modal interaction. *arXiv preprint arXiv:2604.27393*, 2026.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: a speech-text foundation model for real-time dialogue. *arXiv preprint arXiv:2410.00037*, 2024.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 91–104. IEEE, 2025.

Qingkai Fang, Shoutao Guo, Yan Zhou, Zhengrui Ma, Shaolei Zhang, and Yang Feng. Llama-omni: Seamless speech interaction with large language models. In *International Conference on Learning Representations*, volume 2025, pp. 57607–57624, 2025.

Marcel Gabrea and Douglas O'Shaughnessy. Detection of filled pauses in spontaneous conversational speech. In *6th International Conference on Spoken Language Processing (ICSLP 2000)*, volume 3, pp. 678–681, Beijing, China, 2000. doi: 10.21437/ICSLP.2000-626.

Jiaxuan Gao, Jiaao Chen, Chuyi He, Shusheng Xu, Di Jin, and Yi Wu. From self-evolving synthetic data to verifiable-reward rl: Post-training multi-turn interactive tool-using agents, 2026. URL https://arxiv.org/abs/2601.22607.

Sreyan Ghosh, Sonal Kumar, Ashish Seth, Chandra Kiran Reddy Evuru, Utkarsh Tyagi, S Sakshi, Oriol Nieto, Ramani Duraiswami, and Dinesh Manocha. GAMA: A large audio-language model with advanced audio understanding and complex reasoning abilities. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 6288–6313, Miami, Florida, USA, 2024. Association for Computational Linguistics.

Sreyan Ghosh, Arushi Goel, Jaehyeon Kim, Sonal Kumar, Zhifeng Kong, Sang-gil Lee, Chao-Han Yang, Ramani Duraiswami, Dinesh Manocha, Rafael Valle, and Bryan Catanzaro. Audio flamingo 3: Advancing audio intelligence with fully open large audio language models. In *Advances in Neural Information Processing Systems*, volume 38, pp. 41819–41886, 2025. doi: 10.52202/085713-1396.

John J Godfrey, Edward C Holliman, and Jane McDaniel. Switchboard: Telephone speech corpus for research and development. In *[Proceedings] ICASSP-92: 1992 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 1, pp. 517–520. IEEE, 1992.

Google DeepMind. Build real-time voice applications with Gemini 3.8 Live and 3.5 transcribe. https://blog.google/innovation-and-ai/technology/developers-tools/build-real-time-voice-applications-gemini-audio/, September 2026. Google blog post, September 15, 2026. Models `gemini-3.8-live` and `gemini-3.8-live-extended-thinking` served via the Gemini Live API. Accessed: 2026-09-26.

Advait Gosai, Tyler Vuong, Utkarsh Tyagi, Steven Li, Wenjia You, Miheer Bavare, Arda Uçar, Zhongwang Fang, Brian Jang, Bing Liu, et al. Audio multichallenge: A multi-turn evaluation of spoken dialogue systems on natural human interaction. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 35740–35770, 2026.

Haorui He, Zengqiang Shang, Chaoren Wang, Xuyuan Li, Yicheng Gu, Hua Hua, Liwei Liu, Chen Yang, Jiaqi Li, Peiyang Shi, Yuancheng Wang, Kai Chen, Pengyuan Zhang, and Zhizheng Wu.

Emilia: An extensive, multilingual, and diverse speech dataset for large-scale speech generation. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pp. 885–890, 2024. doi: 10.1109/SLT61566.2024.10832365.

Chi-Yuan Hsiao, Ke-Han Lu, Yu-Kuan Fu, Guan-Ting Lin, Hsiao-Tsung Hung, and Hung-yi Lee. ASPIRin: Action space projection for interactivity-optimized reinforcement learning in full-duplex speech language models. *arXiv preprint arXiv:2604.10065*, 2026.

Muye Huang, Lingling Zhang, Xingyu Yu, Lei Shi, Zhanyu Ma, Jun Xu, Jiuchong Gao, Jinghua Hao, Renqing He, and Jun Liu. Duplexomni: Real-time listening, seeing, thinking, and speaking for full-duplex interaction. *arXiv preprint arXiv:2606.09186*, 2026.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. Camels in a changing climate: Enhancing LM adaptation with Tulu 2. *arXiv preprint arXiv:2311.10702*, 2023.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147.

Beomsoo Kim, Changho Choi, Dohyun Kim, Dongki Lee, Ethan Ewer, Eunchong Kim, Gyeongman Kim, Haechan Kim, Hyeonghwan Kim, Inkyu Park, et al. Raon-speech technical report. *arXiv preprint arXiv:2605.23912*, 2026.

So Kuroki, Yotaro Kubo, Takuya Akiba, and Yujin Tang. KAME: Tandem architecture for enhancing knowledge in real-time speech-to-speech conversational AI. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 19362–19366. IEEE, 2026.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019. doi: 10.1162/tacl_a_00276.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

Sehun Lee, Kang-wook Kim, and Gunhee Kim. Behavior-SD: Behaviorally aware spoken dialogue generation with large language models. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 9574–9593, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.484. URL https://aclanthology.org/2025.naacl-long.484/.

Willem J. M. Levelt. Monitoring and self-repair in speech. *Cognition*, 14(1):41–104, 1983. doi: 10.1016/0010-0277(83)90026-4.

Stephen C Levinson. Turn-taking in human communication–origins and implications for language processing. *Trends in cognitive sciences*, 20(1):6–14, 2016.

Tianpeng Li, Jun Liu, Tao Zhang, Yuanbo Fang, Da Pan, Mingrui Wang, Zheng Liang, Zehuan Li, Mingan Lin, Guosheng Dong, et al. Baichuan-audio: A unified framework for end-to-end speech interaction. *arXiv preprint arXiv:2502.17239*, 2025.

Guan-Ting Lin, Jiachen Lian, Tingle Li, Qirui Wang, Gopala Anumanchipalli, Alexander H Liu, and Hung-yi Lee. Full-duplex-bench: A benchmark to evaluate full-duplex spoken dialogue models on turn-taking capabilities. In *2025 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 1–8. IEEE, 2025.

Guan-Ting Lin, Chen Chen, Zhehuai Chen, and Hung-yi Lee. Full-duplex-bench-v3: Benchmarking tool use for full-duplex voice agents under real-world disfluency. *arXiv preprint arXiv:2604.04847*, 2026a.

Guan-Ting Lin, Shih-Yun Shan Kuan, Jiatong Shi, Kai-Wei Chang, Siddhant Arora, Shinji Watanabe, and Hung-yi Lee. Full-duplex-bench-v2: A multi-turn evaluation framework for duplex dialogue systems with an automated examiner. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 27–36, 2026b.

Guan-Ting Lin, Shih-Yun Shan Kuan, Qirui Wang, Jiachen Lian, Tingle Li, Shinji Watanabe, and Hung-yi Lee. Full-duplex-bench v1.5: Evaluating overlap handling for full-duplex speech models. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 19447–19451. IEEE, 2026c.

Mistral-AI, Alexander H. Liu, Andy Ehrenberg, Andy Lo, Chen-Yo Sun, Guillaume Lample, Jean-Malo Delignon, Khyathi Raghavi Chandu, Patrick von Platen, Pavankumar Reddy Muddireddy, Rohin Arora, Sanchit Gandhi, Sandeep Subramanian, et al. Voxtral realtime. *arXiv preprint arXiv:2602.11298*, 2026.

Kumiko Murata. Intrusive or co-operative? A cross-cultural study of interruption. *Journal of Pragmatics*, 21(4):385–400, 1994. doi: 10.1016/0378-2166(94)90011-6.

Tu Anh Nguyen, Eugene Kharitonov, Jade Copet, Yossi Adi, Wei-Ning Hsu, Ali Elkahky, Paden Tomasello, Robin Algayres, Benoît Sagot, Abdelrahman Mohamed, and Emmanuel Dupoux. Generative spoken dialogue language modeling. *Transactions of the Association for Computational Linguistics*, 11:250–266, 2023. doi: 10.1162/tacl_a_00545.

Tu Anh Nguyen, Benjamin Muller, Bokai Yu, Marta R. Costa-jussà, Maha Elbayad, Sravya Popuri, Christophe Ropers, Paul-Ambroise Duquenne, Robin Algayres, Ruslan Mavlyutov, Itai Gat, Mary Williamson, Gabriel Synnaeve, Juan Pino, Benoît Sagot, and Emmanuel Dupoux. SpiRit-LM: Interleaved spoken and written language model. *Transactions of the Association for Computational Linguistics*, 13:30–52, 2025. doi: 10.1162/tacl_a_00728.

OpenAI. Introducing GPT-Live. `https://openai.com/index/introducing-gpt-live/`, July 2026. Blog post, July 8, 2026. Accessed: 2026-09-23.

Titouan Parcollet, Yuan Tseng, Shucong Zhang, and Rogier C. van Dalen. Loquacious set: 25,000 hours of transcribed and diverse english speech recognition data for research and commercial use. In *Interspeech 2025*, pp. 4053–4057, 2025. doi: 10.21437/Interspeech.2025-720.

Akshara Prabhakar, Zuxin Liu, Ming Zhu, Jianguo Zhang, Tulika Manoj Awalgaonkar, Shiyu Wang, Zhiwei Liu, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Weiran Yao, Huan Wang, Silvio Savarese, and Caiming Xiong. APIGen-MT: Agentic pipeline for multi-turn data generation via simulated agent-human interplay. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-2184. Datasets and Benchmarks Track.

Soham Ray, Keshav Dhandhania, Victor Barres, and Karthik Narasimhan. τ-Voice: Benchmarking full-duplex voice agents on real-world domains. *arXiv preprint arXiv:2603.13686*, 2026.

Andrew Reece, Gus Cooney, Peter Bull, Christine Chung, Bryn Dawson, Casey Fitzpatrick, Tamara Glazer, Dean Knox, Alex Liebscher, and Sebastian Marin. The candor corpus: Insights from a large multimodal dataset of naturalistic conversation. *Science advances*, 9(13):eadf3197, 2023.

Rajarshi Roy, Jonathan Raiman, Sang-gil Lee, Teodor-Dumitru Ene, Robert Kirby, Sungwon Kim, Jaehyeon Kim, and Bryan Catanzaro. PersonaPlex: Voice and role control for full duplex conversational speech models. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 16137–16141. IEEE, 2026.

Harvey Sacks, Emanuel A. Schegloff, and Gail Jefferson. A simplest systematics for the organization of turn-taking for conversation. *Language*, 50(4):696–735, 1974. doi: 10.2307/412243.

S Sakshi, Utkarsh Tyagi, Sonal Kumar, Ashish Seth, Ramaneswaran Selvakumar, Oriol Nieto, Ramani Duraiswami, Sreyan Ghosh, and Dinesh Manocha. MMAU: A massive multi-task audio understanding and reasoning benchmark. In *The Thirteenth International Conference on Learning Representations*, 2025.

Emanuel A. Schegloff. Overlapping talk and the organization of turn-taking for conversation. *Language in Society*, 29(1):1–63, 2000. doi: 10.1017/S0047404500001019.

Emanuel A. Schegloff, Gail Jefferson, and Harvey Sacks. The preference for self-correction in the organization of repair in conversation. *Language*, 53(2):361–382, 1977. doi: 10.2307/413107.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in neural information processing systems*, 36:68539–68551, 2023.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Elizabeth Shriberg. Disfluencies in Switchboard. In *Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, pp. 11–14, Philadelphia, PA, USA, 1996.

Elizabeth Shriberg. To 'errrr' is human: Ecology and acoustics of speech disfluencies. *Journal of the International Phonetic Association*, 31(1):153–169, 2001. doi: 10.1017/S0025100301001128.

Tanya Stivers, Nicholas J Enfield, Penelope Brown, Christina Englert, Makoto Hayashi, Trine Heinemann, Gertie Hoymann, Federico Rossano, Jan Peter de Ruiter, Kyung-Eun Yoon, et al. Universals and cultural variation in turn-taking in conversation. *Proceedings of the National Academy of Sciences*, 106(26):10587–10592, 2009.

Utkarsh Tyagi, Ramaneswaran Selvakumar, Advait Gosai, Sonal Kumar, Nikhil Barhate, Isabell Sagar, Steven Li, Miheer Bavare, Daniel Quigley, Fabiola Tapia Carrillo, Jose M Patron E, Diego Macías Gutiérrez, Paul Song, Ramani Duraiswami, Dinesh Manocha, and Yunzhong He. SteerDuplex: Steerable duplex speech dialogue models. *arXiv preprint arXiv:2609.12623*, 2026.

Bandhav Veluri, Benjamin N Peloquin, Bokai Yu, Hongyu Gong, and Shyamnath Gollakota. Beyond turn-based interfaces: Synchronous llms as full-duplex dialogue agents. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 21390–21402, 2024.

Xiong Wang, Yangze Li, Chaoyou Fu, Yike Zhang, Yunhang Shen, Lei Xie, Ke Li, Xing Sun, and Long Ma. Freeze-omni: A smart and low latency speech-to-speech dialogue model with frozen LLM. In *Forty-second International Conference on Machine Learning*, 2025. URL `https://openreview.net/forum?id=s1EImzs5Id`.

Nigel Ward and Wataru Tsukahara. Prosodic features which cue back-channel responses in English and Japanese. *Journal of Pragmatics*, 32(8):1177–1207, 2000. doi: 10.1016/S0378-2166(99)00109-5.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China information sciences*, 68(2):121101, 2025.

Zhifei Xie and Changqiao Wu. Mini-omni: Language models can hear, talk while thinking in streaming. *arXiv preprint arXiv:2408.16725*, 2024.

Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfa Zhu, et al. Qwen3-omni technical report. *arXiv preprint arXiv:2509.17765*, 2025.

Jing Xu, Arthur Szlam, and Jason Weston. Beyond goldfish memory: Long-term open-domain conversation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5180–5197, 2022.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Peiqi Yin, Jiangyun Zhu, Han Gao, Chenguang Zheng, Yongxiang Huang, Taichang Zhou, Ruirui Yang, Weizhi Liu, Weiqing Chen, Canlin Guo, Didan Deng, Zifeng Mo, Cong Wang, James Cheng, Roger Wang, and Hongsheng Liu. vLLM-Omni: Fully disaggregated serving for any-to-any multimodal models. *arXiv preprint arXiv:2602.02204*, 2026.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025a.

Wenyi Yu, Siyin Wang, Xiaoyu Yang, Xianzhao Chen, Xiaohai Tian, Jun Zhang, Guangzhi Sun, Lu Lu, Yuxuan Wang, and Chao Zhang. SALMONN-omni: A standalone speech LLM without codec injection for full-duplex conversation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 24617–24643, 2025b. doi: 10.52202/085713-0829.

Neil Zeghidour, Eugene Kharitonov, Manu Orsini, Václav Volhejn, Gabriel de Marmiesse, Edouard Grave, Patrick Pérez, Laurent Mazaré, and Alexandre Défossez. Streaming sequence-to-sequence learning with delayed streams modeling. *arXiv preprint arXiv:2509.08753*, 2025.

Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 15757–15773, 2023.

Haoyang Zhang, Jun Chen, Donghang Wu, Yuxin Li, Yuxin Zhang, Xiangyu Tony Zhang, Che Liu, Qingjian Lin, Yizhou Peng, Hexin Liu, et al. Duplexsla: A full-duplex spoken language model with synchronized speech, language, and action. *arXiv preprint arXiv:2605.20755*, 2026.

Jiayi Zhang, Simon Yu, Derek Chong, Anthony Sicilia, Michael R. Tomz, Christopher D. Manning, and Weiyan Shi. Verbalized sampling: How to mitigate mode collapse and unlock LLM diversity. *arXiv preprint arXiv:2510.01171*, 2025a.

Qinglin Zhang, Luyao Cheng, Chong Deng, Qian Chen, Wen Wang, Siqi Zheng, Jiaqing Liu, Hai Yu, Chao-Hong Tan, Zhihao Du, et al. Omniflatten: An end-to-end gpt model for seamless voice conversation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14570–14580, 2025b.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

Wenting Zhao, Xiang Ren, Jack Hessel, Claire Cardie, Yejin Choi, and Yuntian Deng. WildChat: 1M ChatGPT interaction logs in the wild. In *Proceedings of ICLR*, 2024.

A ARCHITECTURAL DETAILS

A.1 FULL-DUPLEX STREAM FORMULATION

The thinker operates on fixed 480 ms micro-turns. At each micro-turn, it consumes the next chunk of user audio and predicts a control token indicating whether to listen or speak. A listening step contains only <LISTEN>, while a speaking step begins with <SPEAK>, followed by response tokens and either <chunk_eos> if generation continues or <turn_eos> if the response is complete. The resulting stream therefore alternates between encoded user-audio chunks and model outputs.

For example, while the user is speaking, the model may emit

<LISTEN>

and once the user yields the floor,

<SPEAK> Sure, let me check <chunk_eos>

followed by

<SPEAK> that for you. <turn_eos>.

Because the control decision is conditioned on the current audio chunk, an interruption can be handled within the same micro-turn: if new user speech arrives while the model is speaking, the thinker may immediately emit <LISTEN> and yield the floor.

Streaming Tool Calls. Real-time inference limits the thinker to approximately 20 generated tokens per micro-turn. Tool calls can exceed this budget, so we stream them across consecutive micro-turns using <tool_stream> until the call is complete. Tool-call tokens use the underlying LLM's native tool format and are not passed to the talker.

Importantly, tool use does not suspend spoken interaction. During the micro-turns in which a tool call is being committed, the thinker may also emit short spoken text, such as "let me check that," while the structured tool-call tokens continue to stream. Once execution completes, the tool response is inserted into the thinker's context before processing the corresponding micro-turn. The thinker first consumes this tool response and then makes its normal <LISTEN>/<SPEAK> decision, allowing it to begin verbalizing the result immediately.

Thus, tool calls and tool responses are interleaved with the same full-duplex stream rather than handled as a separate blocking stage. The model can continue speaking while a call is being produced and can produce speech in the same micro-turn in which returned tool information is consumed.

Streaming Tool Calls. Real-time inference limits the thinker to approximately 20 generated tokens per micro-turn, so longer tool calls may span several micro-turns. We therefore stream tool-call tokens across consecutive micro-turns using <tool_stream>, while allowing ordinary spoken text to be emitted alongside them. Tool-call tokens are filtered before reaching the talker.

For example, a lookup request may be emitted as

<SPEAK> Let me check that <tool_call> lookup_customer(...)

followed in the next micro-turn by

<SPEAK> <tool_stream> ... </tool_call>.

Once execution completes, the returned tool result is inserted into the thinker context before the next decoding decision. The model can then consume the result and immediately decide whether to continue listening or speak, for example,

<SPEAK> I found your account. <turn_eos>.

Thus, tool use remains part of the same full-duplex stream: the model can speak while a call is being committed and can begin verbalizing a tool result as soon as it is consumed.

A.2 Thinker Architecture and Objective

The thinker consists of a causal streaming audio encoder, a lightweight audio adapter, and a decoder-only language-model backbone. User audio is processed in fixed 480 ms micro-turns. For each micro-turn, the audio encoder produces causal frame-level representations, which are concatenated and projected by a two-layer MLP into six embeddings in the hidden space of the language model. These embeddings are interleaved with the model's autoregressive token stream.

After consuming each audio micro-turn, the thinker predicts a control token indicating whether it should continue listening or speak. A <LISTEN> prediction produces no textual response for that micro-turn, while <SPEAK> is followed by a short segment of response tokens. Speaking segments terminate with <chunk_eos> when the response continues into a later micro-turn or <turn_eos>

when the turn is complete. Because this decision is conditioned on the current audio chunk, the same mechanism handles taking the floor, holding it, and immediately yielding when the user interrupts.

For a sequence of audio micro-turns $A_1, \ldots, A_U$ and corresponding output segments $o_1, \ldots, o_U$, the thinker operates over the interleaved context

$$X = [\, p, A_1, o_1, A_2, o_2, \ldots, A_U, o_U \,], \tag{2}$$

where p is the system prompt. We train only on the autoregressive output tokens, minimizing

$$\mathcal{L}_{\text{thinker}} = -\sum_{u=1}^{U} \sum_{j=1}^{|o_u|} \log p_\theta \left(o_{u,j} \mid p, A_{\le u}, o_{<u}, o_{u,<j} \right). \tag{3}$$

Responses and interaction decisions are thus learned with one next-token objective.

Native Tool Use. Tool calls are generated directly by the language-model backbone using its native textual tool-call format. Since real-time inference limits each micro-turn to approximately 20 generated tokens, long calls may span several micro-turns and are continued using `<tool_stream>`. Tool-call tokens are intercepted by the runtime and are not passed to the talker, while ordinary text generated alongside the call can still be spoken.

When a tool returns, its result is inserted into the thinker's context before decoding the corresponding micro-turn. The thinker therefore consumes the tool result before making its next `<LISTEN>` or `<SPEAK>` decision, allowing speech generation to continue while calls are being committed and to begin immediately once returned tool information becomes available. This keeps tool use within the same streaming full-duplex loop rather than introducing a separate blocking agent stage.

A.3 Talker Architecture and Objective

Our talker follows the temporal–depth transformer design used in Moshi (Défossez et al., 2024) and operates on Mimi codec tokens at 12.5 Hz. A 14-layer temporal transformer processes the text and delayed audio streams once per frame, predicts the semantic codec token, and conditions a 5-layer depth transformer that generates the remaining seven acoustic codebooks.

We adopt Delayed Streams Modeling (DSM) (Zeghidour et al., 2025). Text is placed on the frame grid using word-level forced alignment. Audio is delayed by $\delta = 6$ frames (480 ms) relative to text, and acoustic codebooks 1–7 are delayed by an additional $\alpha = 2$ frames relative to the semantic codebook. In addition to the text and codec streams, the talker receives a lookahead stream containing tokens from the word $L = 2$ positions ahead and a projected thinker hidden state corresponding to the most recently started word.

At frame t, the temporal transformer receives

$$x_t = E_{\text{txt}}(w_t) + P_\ell E_{\text{txt}}(\ell_t) + \sum_{k=0}^{7} E^{(k)}_{\text{aud}}(c^{(k)}_t) + g(z_t), \tag{4}$$

where ℓ_t is the lookahead stream and z_t is the thinker conditioning state. The semantic codebook is predicted directly from the temporal-transformer output, while the depth transformer autoregressively predicts the remaining acoustic codebooks within each frame.

Following DSM, we additionally attach an action head that predicts whether the next text input should begin a new word. At inference time, thinker outputs are buffered in a word queue and the action head determines when each word is released, decoupling text generation from speech timing.

The training objective is

$$\mathcal{L}_{\text{talker}} = \mathcal{L}_{\text{sem}} + \frac{1}{7} \sum_{k=1}^{7} \mathcal{L}^{(k)}_{\text{ac}} + \mathcal{L}_{\text{act}}, \tag{5}$$

where $\mathcal{L}_{\text{sem}}$ is cross-entropy on the semantic codec stream, $\mathcal{L}^{(k)}_{\text{ac}}$ is cross-entropy for acoustic codebook k, and $\mathcal{L}_{\text{act}}$ is binary cross-entropy for the word-onset action. We down-weight the no-word class in $\mathcal{L}_{\text{act}}$ by 0.5.

Optimization	Before	After
Audio encoder step	33 ms	5.3 ms
Thinker micro-turn	∼200 ms	76.5 ms
Talker frame	22 ms	6.2 ms
First-audio latency	∼370 ms	160–270 ms

Table 3: Latency improvements from our real-time inference optimizations. This exlcudes the median 240ms turn-taking latency

To enable classifier-free guidance, with probability 0.2 we jointly drop the text, lookahead, and thinker-conditioning streams, masking the action loss for those examples. We additionally drop the lookahead stream alone with probability 0.5, since future words may not yet be available at serving time. Fine-tuning reproduces the partial lookahead seen at inference.

A.4 Real-Time Inference

Full-duplex inference imposes a strict latency constraint: every 480 ms micro-turn must encode the newly arrived audio, update the thinker context, and decode the next interaction decision before the following chunk arrives. Since both the thinker and talker operate on small sequential steps, inference is largely dominated by kernel-launch and serving overhead rather than arithmetic. We therefore optimize the serving stack to minimize repeated computation and per-step overhead.

Persistent Conversation State. We serve each conversation as a single resumable request rather than creating a new request for every micro-turn. This keeps the paged KV cache resident throughout the conversation, so each new 480 ms chunk requires only a 7-token incremental prefill, which takes approximately 7.5 ms and remains independent of conversation length.

Incremental Audio Encoding. Re-encoding the complete audio history at every micro-turn introduces increasing computation as a conversation grows. We instead cache the causal encoder state and process only the newly arrived audio. To further reduce the substantial overhead of executing many small encoder operations, we capture the complete incremental encoder step as a CUDA graph using a fixed-size KV-cache ring buffer. This reduces encoder latency from 33 ms to 5.3 ms per micro-turn while closely matching full-prefix encoding.

Graph-Captured Thinker. We similarly use CUDA-graph execution for the thinker to reduce kernel-launch and host-side overhead during autoregressive decoding. Thinker latency decreases from approximately 200 ms per micro-turn in eager execution to 76.5 ms median latency with graph-captured execution, while preserving identical decoded outputs.

Graph-Captured Talker. The talker must generate one Mimi frame every 80 ms. A standard vLLM-Omni execution path requires approximately 22 ms per frame and introduces an additional startup burst at the beginning of each response. We instead capture the complete talker frame, including the temporal transformer, prediction heads, and sequential depth decoding, as a single CUDA graph. This reduces frame generation to 6.2 ms in isolation and 8–13 ms while running concurrently with the thinker. We additionally precompute word-independent startup frames while the user is still speaking, eliminating the startup burst.

Together, these optimizations keep both components comfortably within their real-time budgets. On the path to the first generated audio, a thinker micro-turn requires approximately 60 ms of fixed processing before autoregressive token decoding, and the latency from the thinker's decision to speak to the first audible frame is reduced from approximately 370 ms to 160–270 ms.

B Baseline Details

We compare against seven full-duplex systems that cover the main design choices in current models. We group them first by access (closed source models served through a product or API, and open-weight checkpoints that we run locally) and then, for the open models, by how they achieve full-

Table 4: Thinker training configurations.

	Pre-training	**SFT**	**RL**
Trainable parameters	Adapter + LLM LoRA	LLM LoRA	LLM LoRA
Training steps	14.5K	4,881	150
Peak learning rate	3×10^{-4} / 3×10^{-5}	2×10^{-5}	1×10^{-6}
Global batch size	64	32	8 rollouts/group
Warmup steps	400 / 1,000	150	–
LR schedule	Cosine	Cosine	–
Context / sequence length	4K tokens	32K tokens	Decision windows
GPUs	32 H100	32 H100	16 H100

Table 5: Talker training configurations.

	Pre-training	**SFT**
Trainable parameters	Talker	Talker
Training steps	116K	1,650
Peak learning rate	1×10^{-4}	3×10^{-5}
Global batch size	8	–
Warmup steps	1,000	–
LR schedule	Cosine	–
Context / sequence length	750 frames	750 frames
GPUs	8 H100	–

duplex behavior. Two open models are *dual-stream* models that model user and system audio as parallel token streams on a shared codec clock (Moshi (Défossez et al., 2024), PersonaPlex (Roy et al., 2026)). The other three attach a streaming speech encoder and a speech decoder to a pretrained text LLM and handle simultaneous listening and speaking through interleaving or explicit control tokens (Raon-SpeechChat (Kim et al., 2026), NVIDIA VoiceChat (Balam et al., 2026), MiniCPM-o 4.5 (Cui et al., 2026)). For open models we use the official checkpoints and inference code with the authors' default decoding settings. We do not re-train or fine-tune any baseline.

Gemini Live. Gemini Live (Google DeepMind, 2026) is Google's native-audio real-time model, served through the Gemini Live API. It takes 16 kHz PCM audio and returns 24 kHz audio. We use Gemini 3.8 Live with extended thinking, with all the benchmarks.

GPT-Live. GPT-Live (OpenAI, 2026) is OpenAI's full-duplex voice model, released in July 2026 as the voice mode of ChatGPT. Unlike earlier turn-based voice assistants, it processes input while it is producing output and decides several times per second whether to speak, keep listening, pause, or yield to the user. It also produces short backchannels (e.g., "mhm", "got it") while the user is speaking. GPT-Live uses a backend reasoning model, and we use GPT 5.6-sol as the backend for all the benchmarks, except τ^2-voice which uses GPT6-Astra as the backend.

Moshi. Moshi (Défossez et al., 2024) is the first open real-time full-duplex spoken dialogue model and the backbone of several later systems, including PersonaPlex. It pairs a 7B text language model (Helium) with Mimi, a streaming neural audio codec running at 12.5 Hz. Moshi models the user's speech and its own speech as two parallel token streams, so it listens and speaks at the same time and can produce overlaps, interruptions, and backchannels without an explicit turn-taking module. It also predicts time-aligned text tokens before its audio tokens ("Inner Monologue"). We use the official `kyutai/moshiko-pytorch-bf16` checkpoint and inference code with the default sampling settings (temperature 0.8 for audio and 0.7 for text).

PersonaPlex. PersonaPlex (Roy et al., 2026) is NVIDIA's persona-controllable full-duplex model, built on the Moshi 7B architecture and the Mimi codec. It adds two conditioning inputs to the dual-stream model. The first is a text prompt describing the role, background, and conversation context. The second is a voice prompt, a short speech sample that sets the output voice and speaking style. It was fine-tuned on 1,217 hours of real Fisher English conversations with back-annotated

role prompts, together with about 2,250 hours of synthetic assistant and customer-service dialogues. We use the released `nvidia/personaplex-7b-v1` checkpoint with the official inference code. As the text prompt we use the same system instruction given to all prompt-able baselines, and as the voice prompt we use the default voice. Comparing PersonaPlex with Moshi shows how much fine-tuning on conversation data improves the same architecture.

Raon-SpeechChat. Raon-SpeechChat (Kim et al., 2026) is KRAFTON's open full-duplex speech dialogue model (about 9B parameters). Unlike the Moshi family, it uses a Qwen3 language model backbone. Input speech is encoded with a streaming Voxtral-Mini-4B-Realtime encoder, and speech is generated through a Qwen3-based talker that predicts Mimi codec tokens. It supports simultaneous listening and speaking, with dedicated backchannel tokens and a speak-first or listen-first start mode that is selected at inference time. We use the released `KRAFTON/Raon-SpeechChat-9B` checkpoint with the official demo inference code.

NVIDIA VoiceChat. NVIDIA VoiceChat (NemotronLabs-VoiceChat-11B) (Balam et al., 2026) is an 11B end-to-end full-duplex speech-to-speech model that replaces the cascaded ASR, LLM, and TTS pipeline with a single network. It combines a streaming FastConformer speech encoder (from Nemotron-Speech-Streaming-En-0.6B) with a Nemotron Nano v2 9B hybrid Mamba-Transformer backbone and an NVIDIA TTS decoder. It takes 16 kHz audio and produces 22.05 kHz speech. It was trained on about 550K hours of real and synthetic speech, reports about 450 ms turn-taking latency, and supports tool calls through a separate output channel.

MiniCPM-o 4.5. MiniCPM-o 4.5 (Cui et al., 2026) is a 9B open omni-modal model built from a Whisper-medium audio encoder, a Qwen3-8B language backbone, and a CosyVoice2 speech decoder (plus an unused SigLIP2 vision encoder). Its full-duplex behavior comes from Omni-Flow, which uses time-division multiplexing to interleave the incoming and outgoing streams into short periodic time slices on a shared time axis. A decision step running at 1 Hz lets the model start speaking on its own instead of only reacting to the user. We use the official `openbmb/MiniCPM-o-4_5` checkpoint in full-duplex streaming mode with default decoding settings.

C Riva-Dialog: Data Construction Details

C.1 Shared Data Construction Strategies

Across subsets, we use a few shared strategies to improve data diversity and quality.

Verbalized Sampling. Directly prompting a model to generate conversation openers or scenarios often produces a narrow set of recurring patterns. We therefore use verbalized sampling (Zhang et al., 2025a), asking the generator to propose several distinct candidates together with estimated probabilities and then sampling among them. We use this particularly for conversation seeds and scenario generation, where diversity is important.

Generate-then-Distill. For subsets requiring concise spoken responses, directly prompting the model to answer briefly often compresses many facts into dense responses with poor connective flow. We instead first generate a fluent, fully developed response and then distill it into a shorter spoken form. This preserves the central idea and supporting information while producing responses that are more natural and easier to follow in speech.

C.2 Skill-Specific Data Construction Strategies

Knowledge We construct the knowledge subset from Natural Questions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017), and WebQuestions (Berant et al., 2013), carefully decontaminating against our evaluation sets. Qwen3-30B-A3B converts the source questions into topic cards, which are then used to simulate multi-turn conversations with Qwen3-8B acting as both the user and assistant. Assistant responses are generated for correctness and substance and subsequently distilled into concise spoken form. When gold answers are available, we verify responses against the reference answers and aliases; otherwise, an LLM judge assesses response quality.

Open-Ended Question Answering We construct open-ended QA from WildChat (Zhao et al., 2024) and the Tulu v2 SFT mixture (Ivison et al., 2023). Qwen3-30B-A3B filters prompts that are unsuitable for spoken interaction and converts the remaining examples into topic cards containing the core request and plausible follow-ups. Because these tasks admit many valid conversational trajectories and responses, we use verbalized sampling to encourage diversity during generation. We then simulate multi-turn conversations with Qwen3-8B as both the user and assistant, and use Qwen3-30B-A3B to judge candidate responses for instruction following, helpfulness, and truthfulness. Selected responses are then distilled into concise spoken form.

Safety We construct safety conversations from the Llama Guard safety taxonomy, using verbalized sampling with Qwen-27B to generate diverse scenarios within each harm category. A Qwen-27B user simulator acts as a red-team adversary, while Qwen3-8B plays the defending assistant and is instructed to maintain appropriate safety boundaries throughout the interaction. As a final safeguard, we filter generated assistant responses with Qwen3Guard-Gen-4B and WildGuard before including them in the training set.

Open-Domain Conversation To train natural everyday conversation, we use Multi-Session Chat (Xu et al., 2022), which contains multi-session conversations in which users discuss personal experiences, interests, and events in their lives. We extract a persona from each user and use it to simulate new conversations spanning their interests, experiences, and everyday problems, with Qwen3-8B acting as both the user and assistant simulator. The assistant is instructed to remain grounded in what the user has shared while providing helpful and engaging responses. We complement them with synthetic topics ranging from everyday assistant requests to extended brainstorming, broadening the range and length of open-domain interactions.

Agentic We construct agentic training data in two complementary settings. First, for long-horizon customer-service interactions, we follow an environment-grounded generation pipeline similar in spirit to AReaL-τ^2 (Gao et al., 2026) We define tasks inside mock airline, retail, and telecom environments with executable tools and database state, and roll them out as full multi-turn conversations. Because deriving gold trajectories is expensive in this setting, we instead collect trajectories from strong teacher agents, using Qwen3-30B-A3B, Kimi-K2.6, and GLM-5.2, which we find substantially outperform Qwen3-8B on these long-horizon tasks.

Second, for more general tool use, we follow the task-first, trajectory-second recipe of APIGen-MT (Prabhakar et al., 2025). We first synthesize an executable task grounded in a mock tool environment, then simulate the user-agent interaction turn by turn with tool execution in the loop. Here, Qwen3-8B is sufficient for both user and assistant simulation. APIGen-MT similarly separates task construction from simulated multi-turn interaction, allowing trajectories to evolve from actual tool outputs rather than being generated in a single pass.

For both settings, tool execution is integrated into the spoken trajectory. We execute each generated tool call and record its latency, then generate a short spoken phrase grounded in both the call and its execution time. This teaches the model to continue speaking naturally while a tool call is being committed or executed, rather than blocking the conversation until the result is returned.

C.3 Speech-Native Interaction Augmentation

Interruptions. Interruptions are not inserted uniformly at random, since overlapping speech serves different conversational functions and tends to occur at meaningful points in an ongoing turn. We therefore ground interruption generation in prior work on turn-taking and overlapping speech (Sacks et al., 1974; Schegloff, 2000; Murata, 1994). In particular, Murata (Murata, 1994) distinguishes cooperative interruptions from intrusive ones, with the latter including floor-taking, disagreement, and topic-changing behavior. More broadly, conversation-analysis work treats overlap as part of the organization of turn-taking rather than as arbitrary simultaneous speech (Sacks et al., 1974; Schegloff, 2000).

We instantiate ten interruption types spanning several conversational functions: agreement with added detail, candidate understanding, tangents, specific repair initiation, disagreement, exposed correction, floor-taking, collaborative completion, topic change, and open repair initiation. This

produces both cooperative interruptions, where the user supports or completes the assistant's turn, and more competitive interruptions that redirect or take the floor.

For each augmented conversation, we first sample an interruption type and a coarse position within the assistant response. A planner then identifies a semantically appropriate trigger in the assistant's utterance and generates a short interruption consistent with the sampled behavior. For example, a clarification or correction must be grounded in something the assistant has already said, whereas a floor-taking or topic-change interruption can occur without such a trigger. The assistant turn is truncated at the selected point and the user interruption is inserted, after which the conversation continues with a recovery response from the assistant. We limit each conversation to a small number of interruptions to avoid making overlap unrealistically frequent.

Backchannels. Backchannels are short listener responses such as "mm-hm", "yeah", or "right" that signal attention, understanding, or agreement without taking the conversational floor. They are a common component of natural turn-taking and typically occur in parallel with an ongoing speaker turn rather than constituting a new primary turn (Reece et al., 2023; Ward & Tsukahara, 2000). CANDOR, for example, finds backchannels throughout natural conversation and shows that their frequency depends strongly on the length of the ongoing speaker turn (Reece et al., 2023).

We therefore insert backchannels only at contextually appropriate locations rather than sampling arbitrary timestamps. Given an assistant turn and its recent conversational context, a planner proposes candidate insertion points after specific phrases and assigns each a pragmatic function, including acknowledgment, agreement, surprise, positive feedback, or neutral continuation. We reject candidates that interrupt an unfinished phrase or question, or that occur at the end of the turn where they could instead be interpreted as taking the floor.

To obtain realistic timing and frequency, we subsample the remaining candidates to match statistics measured from CANDOR, including the distribution of backchannels across relative positions within a turn and the expected number of backchannels as a function of turn length. The selected backchannel is synthesized in the user's voice and overlaid on the corresponding assistant turn. Importantly, it does not truncate the assistant response, teaching the model to distinguish brief listener feedback from a genuine interruption.

Speech Repairs. Natural speech frequently contains self-repairs, where a speaker begins an utterance, detects a problem, and revises what was just said. Such repairs are structured rather than arbitrary: prior work describes a strong preference for self-correction in conversation and shows that repairs typically involve an interruption of the ongoing utterance followed by hesitation or an editing expression and the corrected continuation (Schegloff et al., 1977; Levelt, 1983).

We augment user turns with several common forms of self-repair, including corrections of entities or values (e.g., "Tuesday, no, Wednesday"), replacements of a partially produced phrase, false starts followed by restarts, and reformulations that preserve the original intent. Repairs are inserted only where the revised material has a clear local referent, so that the correction can be resolved from the spoken context. In particular, entity and tool-related repairs modify values that matter downstream, such as names, dates, locations, or quantities, requiring the model to disregard the superseded value and act on the corrected one.

We generate the repaired utterance as a single continuous user turn, including appropriate pauses, hesitation markers, or editing expressions around the repair point. This exposes the model to the non-monotonic nature of spoken input, where an earlier interpretation may need to be revised before the user has yielded the floor.

Disfluencies. Spontaneous speech contains frequent disfluencies, including filled pauses, repetitions, false starts, and hesitations, which occur systematically rather than as arbitrary noise (Shriberg, 1996; 2001). They are far more common in conversational than in read speech and can affect downstream processing (Gabrea & O'Shaughnessy, 2000).

We therefore augment user turns with several common disfluency types while preserving the underlying intent of the utterance. We insert filled pauses and hesitation markers at plausible phrase boundaries, introduce occasional repetitions or abandoned fragments before the intended continuation, and vary within-turn pause duration. Unlike speech repairs, these augmentations generally do

not alter the semantic content of the request. Their purpose is to expose the model to the timing and surface irregularities of spontaneous speech, particularly cases where a hesitation or long pause should not be mistaken for an end of turn.

D RIVA-ENTITY: DATA CONSTRUCTION DETAILS

Named entities are particularly challenging for speech models because ASR errors often concentrate on names and other rare structured strings. We therefore construct RIVA-Entity with an entity-aware verification pipeline rather than relying on transcript-level ASR quality alone.

For named entities, we sample entities from Wikipedia together with their accompanying context, and use verbalized sampling to generate natural sentences that mention the entity. For structured entities such as phone numbers, account numbers, and alphanumeric identifiers, we generate synthetic values directly. Each sentence is rendered to speech and transcribed with an ASR model. We then align the hypothesis with the reference and identify whether recognition errors overlap the target entity. Samples are retained only when entity-specific word error remains below a predefined tolerance; otherwise, the TTS sample is discarded and regenerated.

This procedure allows us to tolerate incidental ASR errors elsewhere in the utterance while applying stricter verification to the entity that matters for downstream tool use. The final RIVA-Entity corpus contains approximately 2,000 hours of speech.

E REINFORCEMENT LEARNING DETAILS

E.1 DECISION WINDOWS AND REWARDS

Rather than rolling out complete conversations, we train on short decision windows centered around interaction and tool-use events. Each window consists of a teacher-forced prefix followed by several free micro-turns sampled from the current policy. A micro-turn lasts 480 ms. For windows containing an overlay, u_{on} denotes the micro-turn in which the overlay begins and u_{end} the micro-turn in which the user's speech ends.

We define a *sustained stop* as the first micro-turn beginning at least two consecutive `LISTEN` micro-turns. This distinguishes genuine yielding from the single-`LISTEN` phrase-boundary pauses that occur naturally in SFT responses. Appendix E.2 details the construction and reward of each window.

E.2 WINDOWS AND REWARDS

Rather than rolling out complete conversations, we train on short decision windows centered around interaction and tool-use events. Each window consists of a teacher-forced prefix followed by several free micro-turns sampled from the current policy. A micro-turn lasts 480 ms. For windows containing an overlay, u_{on} denotes the micro-turn in which the overlay begins and u_{end} the micro-turn in which the user's speech ends.

Interrupt windows. The assistant response is forced to a point mid-turn, after which an overlapping user question begins at u_{on}. The reward combines three terms:

$$R = 0.35\, r_{\text{yield}} + 0.15\, r_{\text{quiet}} + 0.5\, r_{\text{take}}.$$

Here, $r_{\text{yield}} \in \{1, 0.75, 0.4\}$ if the sustained stop begins at u_{on}, $u_{\text{on}} + 1$, or $u_{\text{on}} + 2$, respectively, and is zero otherwise. r_{quiet} is the fraction of micro-turns from the sustained stop to u_{end} spent listening. $r_{\text{take}} \in \{1, 0.7, 0.4, 0.2\}$ depending on whether the response begins 1, 2, 3, or more micro-turns after u_{end}. Talking through the interruption receives zero reward.

Hold windows. These use the same construction, but the overlay is a backchannel, DNS5 noise at 0–15 dB SNR, or background speech at -12 to -20 dB. With horizon $H = 10$,

$$R = \begin{cases} 1, & \text{if the model does not yield,} \\ (y - u_{\text{on}})/H, & \text{if it yields at micro-turn } y. \end{cases}$$

A yield is a sustained stop or a `<turn_eos>` within three micro-turns of the overlay onset.

Take-turn windows. The user makes a tool-use request and either finishes normally, pauses once for 1.2–2.0 s, or contains several 0.8–1.8 s internal pauses. The timing reward is zero if the first SPEAK occurs at or before u_{end}, and otherwise takes values $\{1, 0.8, 0.5, 0.2\}$ with increasing response latency. When a tool call is required,

$$R = 0.5\, r_{\text{timing}} + 0.5\, r_{\text{call}},$$

where

$$r_{\text{call}} = 0.6\,\mathbb{1}[\text{correct tool}] + 0.4\,\mathbb{1}[\text{exact arguments}].$$

When no tool call is required, $R = r_{\text{timing}}$.

Identifier windows. Customer-service trajectories are randomly forked shortly before a tool call and rolled out for approximately eight free micro-turns, terminating after the first completed call. The reward is

$$R = \mathbb{1}\left[\text{expected tool} \wedge \text{arguments} = \text{true entity}\right],$$

with arguments compared after normalization. We grade against the true entity in the task database and retain only windows in which the target value is audibly recoverable from the caller's speech.

Retention windows. Long conversational examples are forced up to the final answer and followed by 16 free micro-turns. These receive

$$R = 0,$$

so the environment advantage vanishes and only the KL regularization signal remains. Identifier windows are constructed from customer-service conversations in Riva-Dialog and additional τ^2-style tasks with no overlap with the evaluation tasks. We grade generated arguments against the true entity in the task database rather than the SFT target, since some training trajectories deliberately contain initially misheard values. We retain a window only when the target value is audibly recoverable from the caller's speech, preventing the reward from depending on information removed by simulated channel corruption. Tool-call decoding follows the same streamed execution behavior used at inference time, and identifier rollouts terminate after the first completed call.

E.3 RL Objective

For a decision window with $G = 8$ rollouts and rewards $R_1, \ldots, R_G$, we compute the group-normalized environment advantage

$$A_i^{\text{env}} = \frac{R_i - \bar{R}}{\sigma_R + \varepsilon}, \tag{6}$$

which is broadcast over the free tokens of rollout i. Groups with zero reward variance are dropped unless a teacher signal is available.

We optimize GRPO (Shao et al., 2024) with asymmetric clipping (Yu et al., 2025a) and a bounded KL penalty to the SFT policy:

$$\mathcal{L}(\theta) = -\frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \left[w_t \min(\rho_t A_t,\ \text{clip}\left(\rho_t, 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}}\right) A_t) - \beta \min(k_3(t), 10) \right], \tag{7}$$

where ρ_t is the ratio between the current policy and the policy that generated the rollout,

$$w_t = \min\left(\frac{\pi_{\text{old}}(y_t)}{\pi_{\text{roll}}(y_t)}, 2 \right), \tag{8}$$

and

$$k_3(t) = e^{r_t} - r_t - 1, \qquad r_t = \text{clip}\left(\log \pi_{\text{ref}}(y_t) - \log \pi_\theta(y_t), \pm 5\right). \tag{9}$$

We use $\epsilon_{\text{low}} = 0.2$, $\epsilon_{\text{high}} = 0.28$, and $\beta = 0.05$. The reference policy is the SFT model.

On-Policy Self-Distillation. Exact-match rewards on identifier windows are sparse, so many rollout groups receive uniform reward and provide no environment-gradient signal. We therefore augment these windows with on-policy self-distillation (Zhao et al., 2026). The frozen SFT model receives the exact transcript of the caller's speech as privileged context and scores the student's generated tokens. We add its clipped token-level log-ratio to the environment advantage:

$$A_t = A^{\text{env}} + \lambda\ \text{clip}\left(\log \pi_{\text{T}}(y_t) - \log \pi_\theta(y_t), \pm 8\right), \qquad \lambda = 0.5. \tag{10}$$

We apply this auxiliary signal only to identifier windows; timing windows use GRPO alone.

E.4 Training Details

RL updates only the thinker's LoRA parameters. We use a learning rate of 1×10^{-6} and groups of 8 rollouts. Training proceeds in two stages. Stage 1 trains all decision-window types with GRPO for 100 steps. Stage 2 continues for 50 additional steps and applies on-policy self-distillation to identifier windows. We also introduce longer take-turn requests and add a low-level noise floor to synthesized user audio during this stage, preventing the model from relying on digital silence as the sole cue for turn completion.

F Evaluation Benchmark Details

We evaluate on six public benchmarks grouped into six capability areas, including General Knowledge, Alignment, Multi-Turn Interactivity, Conversational Intelligence, Tool Use and Agentic Tasks. For every benchmark, we follow the official evaluation protocol and scoring scripts.

OpenAudioBench. To evaluate general knowledge under speech input, we use OpenAudioBench (Li et al., 2025), a spoken question-answering benchmark derived from standard text QA datasets. We report accuracy on Llama Questions, Web Questions, and TriviaQA, which cover general-knowledge, entity-centric, and trivia questions, respectively. Following the official protocol, model responses are transcribed and judged against the reference answer by an LLM. It thus measures how well the text LLM's knowledge survives adaptation to speech.

VoiceBench. To evaluate alignment under speech input, we use VoiceBench (Chen et al., 2026), which converts standard text alignment benchmarks into spoken instructions using TTS, with CommonEval recorded by human speakers. We report four subsets: AlpacaEval and CommonEval for open-ended response quality, IFEval for instruction following under verifiable constraints, and AdvBench for refusal behavior on harmful requests. We follow the official scoring protocol, using LLM-based ratings for AlpacaEval and CommonEval, rule-based checks for IFEval, and refusal rate for AdvBench. Together, these benchmarks test whether helpfulness, instruction following, and safety behaviors are preserved when instructions are delivered through speech.

Full-Duplex-Bench v2. To evaluate multi-turn interactivity, we use Full-Duplex-Bench v2 (Lin et al., 2026b), which tests full-duplex systems in live conversations with an automated real-time examiner. The examiner follows staged conversational goals under both *fast* pacing, where it may interrupt or backchannel while the system is speaking, and *slow* pacing, where it waits for an end of turn. Tasks cover daily interactions, correction handling, entity tracking, and safety. An LLM judge scores turn-taking fluency, instruction following, and task success on 1–5 scales. This benchmark measures whether the model can maintain coherent, appropriately timed behavior across overlapping and evolving multi-turn conversations.

Audio MultiChallenge. To evaluate conversational intelligence over extended spoken interactions, we use Audio MultiChallenge (Gosai et al., 2026), which contains 452 natural multi-turn conversations from 47 speakers. It targets four capabilities: inference memory, instruction retention, self-coherence, and voice editing, including spoken repairs and revisions. Responses are evaluated against 1,712 instance-specific binary rubrics using an LLM judge. We report average pass rate (APR), the fraction of conversations satisfying all rubrics, and average rubric score (ARS), which gives partial credit across rubrics. The benchmark tests whether models can reason over conversational history while handling the revisions and disfluencies characteristic of natural speech.

Full-Duplex-Bench v3. To evaluate tool use under realistic spoken interaction, we use Full-Duplex-Bench v3 (Lin et al., 2026a), which contains 100 recordings of real human speech spanning four tool-use domains and includes natural disfluencies such as pauses, false starts, and self-corrections. We report Pass@1 (P@1), which requires the expected tool calls with correct arguments, argument accuracy (Arg. Acc.), which measures semantic correctness of tool arguments, and response quality, which evaluates whether the spoken response correctly communicates the tool result. The benchmark tests whether a full-duplex agent can recover the intended tool parameters from disfluent and self-correcting speech and execute the appropriate actions. Models without a tool-calling interface are marked as "–".

τ-Voice. To evaluate end-to-end agentic task completion in speech, we use τ-Voice (Ray et al., 2026), an extension of τ^2-bench (Barres et al., 2025) to full-duplex voice agents. It contains 278 customer-service tasks across airline, retail, and telecom, where the agent must interact with a simulated caller, follow domain policies, and invoke tools against a task database. The simulator includes diverse accents, background audio, and natural turn-taking behaviors such as interruptions. We utilize GPT4o TTS and GPT4o Transcribe model as the speech front-ends for the user simulator. We report Pass@1 per domain, where a task is solved only if the resulting database state and required outputs match the annotated goal. This benchmark measures how well text-level agentic capabilities transfer to real-time spoken interaction. Models without a tool-calling interface are marked as "–".