

VoxMind: An End-to-End Agentic Spoken Dialogue System

Anonymous ACL submission

Abstract

Recent end-to-end spoken dialogue models enable natural interaction. However, as user demands become increasingly complex, models that rely solely on conversational abilities often struggle to cope. Incorporating agentic capabilities is therefore essential: by enabling tool use, these models can extend their knowledge boundaries and better solve real-world tasks. Yet, existing research has largely concentrated on core perception and generation, with comparatively limited exploration of such tool-augmented extensions. To bridge this gap, we present **VoxMind**, an integrated framework designed to **equip end-to-end spoken dialogue models with comprehensive agentic abilities**. Leveraging our curated 470-hour **AgentChat** dataset, we incorporate a "Think-before-Speak" mechanism, enabling the model to internalize structured reasoning as a critical prerequisite for planning and response generation. Furthermore, to mitigate latency bottlenecks caused by large-scale tool integration, we propose a **Multi-Agent Dynamic Tool Management architecture**. By asynchronously delegating retrieval tasks to an auxiliary agent aligned with the main model's reasoning trajectory, this system effectively **decouples inference latency from toolset size**. Experimental results confirm that VoxMind achieves significant improvements in agent performance: compared with strong baselines, the task completion rate increases from 34.88% to 74.57%, outperforming Gemini-2.5-Pro on spoken agent tasks while preserving general conversational quality.

1 Introduction

End-to-end spoken dialogue models (Zhang et al., 2023; Xie and Wu, 2024; Chen et al., 2024a; KimiTeam et al., 2025; Wu et al., 2025; Xu et al., 2025c) have emerged as a paradigm shift in speech-based human-computer interaction, as they directly model paralinguistic information and generate expressive spoken responses within a unified speech

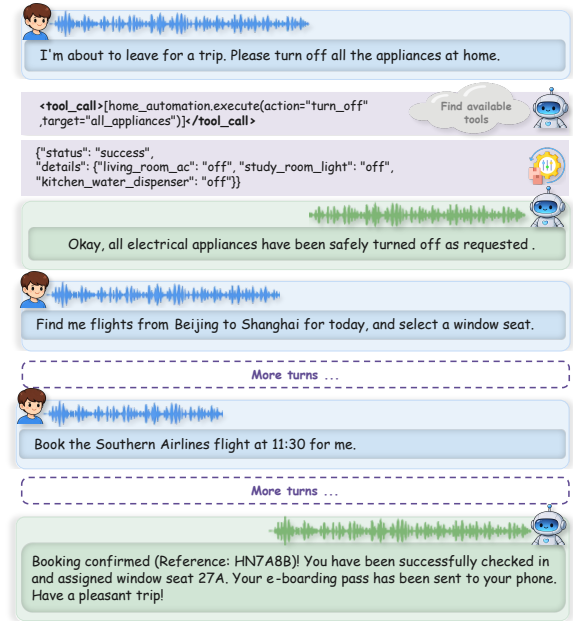


Figure 1: VoxMind can dynamically perceive the interaction context, autonomously determine when to invoke external tools, and drive the generation of subsequent responses based on the tool execution results.

modality (Ji et al., 2024). These models have achieved rapid progress in perception and generation, substantially improving naturalness and responsiveness in conversational settings (Li et al., 2025a; Xu et al., 2025b; Long et al., 2025; Li et al., 2025b). Nevertheless, most existing systems remain primarily optimized for reactive conversation, exhibiting limited capacity to handle complex, goal-oriented tasks that require reasoning, planning, and external knowledge access.

Research on text-based agents has shown that mature tool-calling and planning mechanisms can substantially enhance large language models in handling real-time knowledge access and complex reasoning (Schick et al., 2023; Luo et al., 2025). In contrast, end-to-end spoken agents remain relatively underexplored and face a set of closely

related challenges. At the conceptual level, the speech domain still lacks a unified and widely accepted definition of what constitutes an end-to-end spoken agent, leaving both model design and evaluation without a clear standard. From a capability perspective, end-to-end spoken dialogue models generally lag behind pure text-based models in fine-grained semantic understanding and structured action formulation, such as interpreting tool semantics and generating well-formed tool invocations with appropriate parameters. This limitation directly constrains their ability to support robust planning and long-horizon decision making. The situation is further compounded by the scarcity of speech data explicitly annotated with agentic behaviors, including structured reasoning traces and tool interaction supervision. Moreover, spoken inputs inherently require substantially more tokens to encode rich acoustic information than text, and when combined with large-scale tool descriptions, this results in significant computational overhead, leading to increased inference latency and hindering practical deployment.

To bridge these gaps, we first formulate a rigorous definition of End-to-End Spoken Agents, establishing a unified standard for agentic behaviors in the speech domain. Guided by this formulation, we propose **VoxMind**, a unified framework that integrates autonomous reasoning, tool utilization, and natural spoken interaction, as illustrated in Fig1. To enhance planning capabilities in complex scenarios, VoxMind adopts a "**Think-before-Speak**" mechanism, enabling the model to perform explicit internal reasoning prior to response generation.

To support reasoning-aware training, we construct the **AgentChat** dataset, a large-scale spoken corpus annotated with structured reasoning trajectories and tool interaction labels. Training on AgentChat enables VoxMind to internalize cognitive planning processes and generate structured reasoning and tool invocations directly from spoken context.

In addition, to enable scalable tool usage with low latency, VoxMind incorporates a **Dynamic Tool Management** mechanism based on a multi-agent design. The system maintains a compact, reasoning-conditioned local tool space that is dynamically updated with candidate tools selected from a global pool, thereby avoiding repeated processing of the entire tool library. This design effectively decouples inference efficiency from toolset scale, enabling responsive decision making in tool-

rich environments.

In summary, our main contributions are as follows:

- We formulate a formal definition for End-to-End Spoken Agents, bridging a critical theoretical gap in the field. Building on this foundation, we propose **VoxMind**, a unified model that incorporates a "Think-before-Speak" paradigm to effectively execute these complex reasoning and tool-use tasks.
- We construct **AgentChat**, a speech dataset explicitly annotated with reasoning trajectories, tool interactions, and complex planning paths. This resource alleviates the scarcity of agentic supervision in spoken contexts, facilitating the development of reasoning-aware speech agent.¹
- We design a **Multi-Agent Dynamic Tool Management architecture** that employs an asynchronous parallel execution strategy. This mechanism decouples inference latency from the size of the tool library, ensuring consistent performance and accuracy as the toolset expands.

2 Related Work

The reliance of pre-trained large language models (LLMs) on static training data limits their adaptability to dynamic scenarios (Qu et al., 2024). The autonomous agent paradigm mitigates this by enabling models to interface with external tools (Masterman et al., 2024). While reasoning frameworks are well-established in the text domain (Yao et al., 2022; Qin et al., 2023; Hong et al., 2023), the extension to end-to-end voice interaction remains nascent. Recent works, including Stream RAG (Arora et al., 2025), TARL (Tan et al., 2025), and Qwen3-Omni (Xu et al., 2025c), demonstrate preliminary agent capabilities. However, these efforts lack systematic exploration, primarily limiting models to isolated functionalities such as information retrieval or basic tool use. Consequently, solving complex problems necessitates a comprehensive system architecture, as simple functional extensions are insufficient.

¹We plan to release all datasets upon acceptance of the paper.

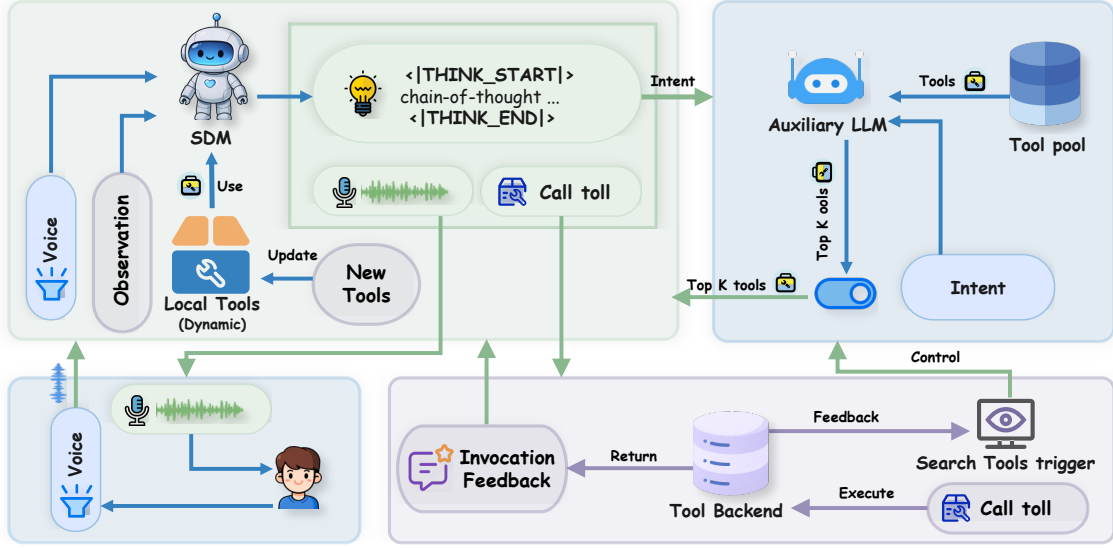


Figure 2: Overall architecture of the VoxMind. Given spoken user input, the speech-centric agent first generates an explicit reasoning trajectory in a "think-before-speak" manner. Conditioned on this reasoning output, the speech model generates a response, while an auxiliary language model operates in parallel to propose candidate tools from a global pool. The selected action and the proposed tool set jointly determine dynamic updates to the agent’s local tool space, enabling scalable tool usage without increasing response latency.

3 Methodology

3.1 Unified Definition of End-to-End Spoken Agents

We define an **End-to-End Spoken Agent** as an autonomous system that transcends reactive speech generation to possess cognitive and executable capabilities. To facilitate complex problem-solving in spoken scenarios, we formulate the agent \mathcal{A} as a unified framework consisting of four essential dimensions.

Profile Definition. A comprehensive agent profile must encompass both semantic roles and acoustic identities. We decompose this definition \mathcal{P} into two dimensions to balance consistency with adaptability: **Static Definition (Consistency):** This specifies the agent’s inherent attributes, denoted as P_{static} , such as timbre, gender, age, accent, and semantic persona (e.g., customer service agent, educational expert). These features are pre-defined to maintain a cohesive persona throughout interactions, ensuring the user perceives a stable conversational partner. **Dynamic Adaptive Definition (Autonomy):** This encompasses the agent’s self-definition $P_{dynamic}$, derived from real-time environmental interaction and self-reflection. Attributes such as emotional tone, speaking rate, rhythm, and prosody are not hard-coded; rather, they are dynamically determined by the agent in

response to the context c (e.g., sensing user urgency). This mechanism reflects the agent’s situational awareness and autonomy, formalized as $\mathcal{P} = (P_{static}, P_{dynamic}(c))$.

Memory Mechanism. To overcome the base model’s inherent statelessness, a robust memory mechanism needs to be introduced to persist interactions across time. This mechanism enforces a **dual-channel architecture** throughout all storage levels, maintaining both **Semantic Memory** (\mathcal{M}_{sem}) and **Acoustic Memory** (\mathcal{M}_{acous}) to capture what was said and how it was said. **Short-Term Memory (\mathcal{M}_{ST}):** Functioning as working memory, this module buffers the immediate multi-modal context. It simultaneously retains semantic content and paralinguistic acoustic features (e.g., emotion, pitch), enabling the agent to maintain situational awareness in real-time fluid interactions. **Long-Term Memory:** This component archives persistent knowledge accumulated over extended periods. It stores not only historical facts and user preferences (Semantic) but also recurring vocal patterns and prosodic habits (Acoustic), ensuring long-term consistency in the agent’s interaction style.

Planning Capability. To solve complex real-world problems, the agent cannot rely solely on reactive behavior (i.e., reflexive response). While end-to-end models typically perform a direct map-

ping from input to output ($x \rightarrow y$), this formulation is often insufficient for complex planning tasks. Thus, an effective agent requires an intermediate reasoning stage z , transforming the interaction paradigm into $x \rightarrow z \rightarrow y$. Here, $x \in \mathcal{X}$ denotes the multimodal input, $z \in \mathcal{Z}$ represents the intermediate reasoning process (e.g., chain-of-thought, task decomposition, or latent logic generation), and $y \in \mathcal{Y}$ corresponds to the final executed action or spoken response. This intermediate step z enables the agent to deliberate and formulate a structured plan prior to execution.

Action Execution. Planning alone remains theoretical without execution. Therefore, this principle centers on **Tool Utilization**, where the execution process is governed by two sequential decision-making stages. **Decision:** The agent evaluates the current context to determine if external assistance is necessary to fulfill the plan. **Selection And Invocation:** Upon confirming the need for tools, the agent identifies the optimal tool t^* from the available API set \mathcal{T} and generates the precise parameters required for invocation.

3.2 VoxMind

To construct a comprehensive spoken dialogue agent, we propose the VoxMind architecture, as shown in Fig2. The system state at time step t is defined as:

$$\mathcal{S}_t = (\mathbf{O}_t, \mathcal{H}_t, \mathcal{A}_t) \quad (1)$$

where \mathbf{O}_t denotes the set of observable events at time t , comprising the current user input \mathbf{X}_t and structured feedback \mathbf{O}_t^{env} returned by tools or the environment (i.e., $\mathbf{O}_t = \{\mathbf{X}_t, \mathbf{O}_t^{env}\}$). \mathcal{H}_t represents the accumulated interaction history, and \mathcal{A}_t denotes the agent’s action space, consisting of verbal responses \mathcal{V} and a dynamically retrieved subset of callable tools $\mathcal{T}_t^{local} \subset \mathcal{T}^{all}$.

The core objective of VoxMind is to learn a hierarchical policy that maps the system state \mathcal{S}_t to an optimal action $\mathbf{a}_t \in \mathcal{A}_t$ via an explicit "think-before-speak" mechanism. Specifically, before producing speech output or invoking tools, the agent generates an explicit **Chain-of-Thought (CoT) reasoning trajectory**:

$$\mathbf{c}_t \sim \pi_\theta^{\text{think}}(\mathbf{c} \mid \mathbf{o}_t, \mathcal{H}_{t-1}, \mathcal{T}_t^{local}). \quad (2)$$

This trajectory captures intent understanding, contextual analysis, and task planning, ensuring that

the reasoning step is completed prior to any action execution.

Conditioned on the sampled reasoning trajectory, the agent selects its next action based on the current observation, interaction history, and locally accessible tools:

$$\mathbf{a}_t \sim \pi_\theta^{\text{act}}(\mathbf{a} \mid \mathbf{c}_t, \mathbf{o}_t, \mathcal{H}_{t-1}, \mathcal{T}_t^{local}). \quad (3)$$

The resulting action corresponds either to a verbal response or the invocation of an external tool. This ensures that all observable behaviors are grounded in explicit reasoning while remaining consistent with the current context and tool availability.

To effectively decouple inference latency from toolset size for scalable tool usage, VoxMind employs a **parallel dynamic tool update mechanism** driven by an auxiliary language model. After the reasoning trajectory \mathbf{c}_t is generated, the system executes two processes in parallel: the agent samples its next action conditioned on the current local tool set, while an auxiliary model proposes candidate tools from the global pool:

$$(\mathbf{a}_t, \mathcal{T}_t^{cand}) \sim (\pi_\theta^{\text{act}}(\cdot \mid \mathbf{c}_t, \mathcal{T}_t^{local}), \pi_{\text{LLM}}(\mathbf{c}_t, \mathcal{T}^{all})) \quad (4)$$

The sampled action explicitly determines the state transition of the tool space. When the agent emits the retrieval action $\mathbf{a}_t = a_{\text{retrieve}}$, indicating that the current local tool set is insufficient to accomplish the task, the candidate tools proposed by the auxiliary model are incorporated to form the tool space for the next decision step:

$$\mathcal{T}_{t+1}^{local} = \mathcal{T}_t^{local} \cup \mathcal{T}_t^{cand}. \quad (5)$$

Otherwise, the local tool set remains unchanged, i.e., $\mathcal{T}_{t+1}^{local} = \mathcal{T}_t^{local}$.

Conditioned on the updated tool availability, the agent then performs its next decision at time step $t+1$ to obtain the final executable action:

$$\mathbf{a}_{t+1} \sim \pi_\theta^{\text{act}}(\mathbf{a} \mid \mathbf{c}_{t+1}, \mathbf{o}_{t+1}, \mathcal{H}_t, \mathcal{T}_{t+1}^{local}). \quad (6)$$

This design enables scalable tool usage by explicitly triggering tool expansion upon detected insufficiency with minimal inference overhead.

3.3 AgentChat

Basic Interaction Data Construction. To train a robust intelligent agent, we constructed the AgentChat dataset, which comprises two distinct corpora (as shown in Table 1): a **tool-interaction**

AgentChat	Instances	Avg. Turns	Duration (H)
Tool Interaction Data			
tool-ace-audio	5582	1.0672	26.6220
apigen-mt-audio	791	7.4355	43.2587
Self-built (Tool)	8432	1.3455	39.1885
General Dialogue Data			
ai2_arc-challenge	1167	1.0000	12.3334
ai2_arc-easy	1164	1.0000	10.8154
gsm8k	1746	1.0000	18.4730
sciq	998	1.0000	9.4903
Self-built (Normal)	26406	1.1201	309.8364

Table 1: Composition of the AgentChat dataset. AgentChat consists of **Tool Interaction Data** (comprising the *tool-ace-audio*, *apigen-mt-audio*, and *Self-built (Tool)* subsets) and **General Dialogue Data** (comprising the *ai2_arc-challenge*, *ai2_arc-easy*, *gsm8k*, *sciq*, and *Self-built (Normal)* subsets).

corpus and a **general-conversation corpus**. The construction process involves rigorous text collection, cleaning, and speech synthesis; each stage is detailed below.

The tool-interaction corpus is derived from existing benchmark datasets, including ToolACE (Liu et al., 2024) and APIGen-MT (Prabhakar et al., 2025). We first perform coarse rule-based filtering to remove content unsuitable for speech synthesis, such as HTML tags, Markdown markers, and code snippets. Subsequently, fine-grained filtering is carried out using the Qwen-plus² language model, polishing the text to ensure a natural conversational style while removing data inappropriate for speech scenarios. To further enrich the tool data, we also employed the language model to generate a set of task-specific dialogues based on the established tool descriptions from ToolACE (Liu et al., 2024).

The general-conversation corpus integrates publicly available datasets (SciQ (Welbl et al., 2017), GSM8K (Cobbe et al., 2021), ARC (Clark et al., 2018)) as well as data derived from common knowledge found in secondary school textbooks. We selected subsets suitable for speech synthesis, resulting in a domain-balanced collection.

All cleaned text is converted to speech using CosyVoice2. To increase speaker diversity and acoustic naturalness, we utilized over 600 prompt-based timbres from SeedTTS (Anastassiou et al., 2024) during synthesis, producing a stylistically diverse and high-fidelity speech corpus.

Chain-of-Thought Construction. To construct intermediate reasoning trajectories for training, we adopt a reverse conditional generation approach.

Specifically, given a task input Q and the corresponding final output A , the model generates a reasoning chain R that logically bridges them. This process is formulated as sampling from the conditional distribution:

$$R \sim p_{\text{LM}}(R | Q, A). \quad (7)$$

To ensure quality, we implement an iterative filtering mechanism based on scoring. Each candidate reasoning chain R is assigned a quality score $S(R) \in [0, 10]$. Only chains satisfying a predefined threshold $\tau = 7$ are retained:

$$\mathcal{R}_{\text{retain}} = \{R | S(R) \geq \tau\}. \quad (8)$$

For chains falling below the threshold, the system regenerates the reasoning chain up to $T = 3$ times:

$$R_{i+1} \sim p_{\text{LM}}(R_i | Q, A), \quad (9)$$

$$i \in \{i' | i' \leq 2, S(R_{i'}) < \tau\}.$$

Candidates that fail to meet the threshold after three attempts are discarded.

Finally, each retained chain undergoes textual refinement. A large language model polishes the reasoning text to improve conciseness and standardize the format. Guided by instructions \mathcal{I} , this process strictly preserves the core logical flow:

$$R' = \text{LLM}_{\text{refine}}(R | \mathcal{I}). \quad (10)$$

The resulting dataset of refined chains, R' , provides clean and structured trajectory data for effective agent training.

Core Agent Competencies. Our method equips the agent with a suite of core capabilities through targeted training on the AgentChat dataset, as illustrated in Fig3. The design encompasses the following key functions: **single-task processing**, enabling the agent to accurately understand user intent and invoke appropriate tools for independent tasks; **task decomposition**, allowing it to break down complex requests into manageable subtasks; **parallel processing**, which enhances efficiency by identifying independent subtasks of the same type and generating parallel execution plans; **proactive seeking**, empowering the agent to initiate external searches or requests when existing tools are inadequate, thus adapting to open-world scenarios; **result feedback**, which enables dynamic adjustment of subsequent actions based on tool execution outcomes; **contextual planning**, leveraging historical interaction context to maintain coherence in multi-turn dialogues. See Appendix A for details on the dataset composition.

²<https://bailian.console.aliyun.com>

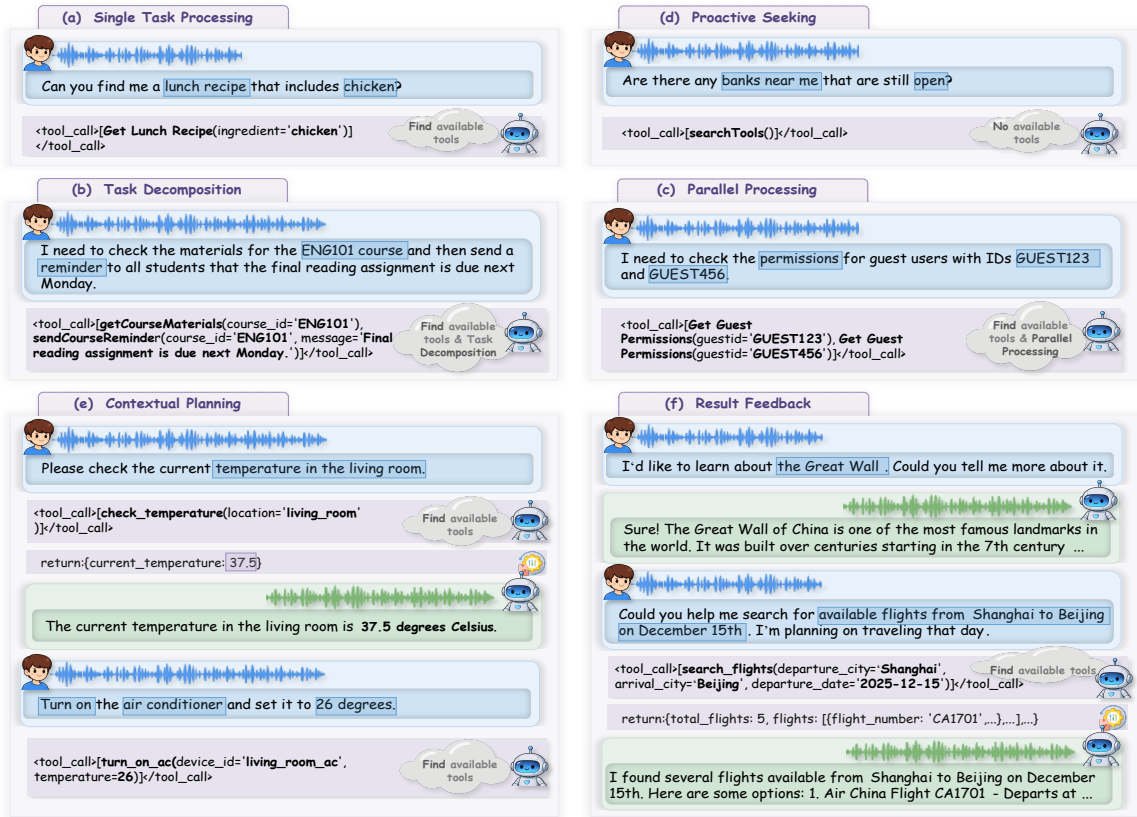


Figure 3: Dialogues demonstrating the agent’s six core capabilities.

4 Experiments

4.1 Experimental Setup

Datasets. Given the lack of open-source agent interaction data in speech environments, we train our model on the AgentChat dataset. We reserve a **disjoint subset** as an independent test set to evaluate the agent’s core capabilities. Additionally, we construct an **out-of-domain** dataset using Gemini-2.5-Pro to investigate model performance across **expanding tool scales** in real-world scenarios. To explore the impact of data proportioning on agent training effectiveness, we further generate two datasets with **distinct ratio configurations** (1:1 and 1:0.5), where the ratio denotes the time proportion between speech-oriented agent interaction data and general dialogue data. Complete dataset statistics and composition details are provided in Appendix G.

Baselines. For extensive comparison, we select a suite of competitive models, including leading closed-source models like Gemini-2.5-pro³, Gemini-2.5-flash, and GPT-4o-audio⁴, as well as

³<https://ai.google.dev/gemini-api>

⁴<https://platform.openai.com/docs/models/gpt-4o-audiopreview>

open-source ones such as Qwen2.5-Omni(Xu et al., 2025a), Kimi-Audio(KimiTeam et al., 2025), and Qwen3+Whisper(Yang et al., 2025; Radford et al., 2022). StepAudio2(Wu et al., 2025), also an open-source model, serves as the foundation for fine-tuning.

Evaluation Setup. Our evaluation covers three complementary aspects. We first assess six core agent capabilities illustrated in Fig.3: single-task processing, task decomposition, parallel processing, proactive seeking, result feedback, and contextual planning. These capabilities are quantified using four task-level metrics: **TS** (Tool Selection accuracy), evaluating correct tool selection from the local tool set; **PF** (Parameter Filling accuracy), measuring structured parameter instantiation from context; **TU** (Tool Usage accuracy), assessing the agent’s ability to detect tool insufficiency and trigger retrieval; and **FC** (Feedback Completeness), evaluating accurate perception and summarization of environment feedback. All evaluations are conducted using **Gemini-2.5-Flash** as an expert evaluator by **verifying model outputs against predefined ground-truth answers**, rather than subjective scoring. To improve robustness and reduce

Model	Single Task Processing		Task Decomposition		Parallel Processing		Contextual Planning		Proactive Seeking	Result Feedback	Overall
	TS \uparrow	PF \uparrow	TS \uparrow	PF \uparrow	TS \uparrow	PF \uparrow	TS \uparrow	PF \uparrow	TU \uparrow	FC \uparrow	
Closed-source models (direct inference with prompt)											
Gemini-2.5-pro	90.98	<u>75.19</u>	<u>82.54</u>	52.38	<u>88.57</u>	69.52	<u>84.25</u>	61.64	<u>26.87</u>	<u>4.16</u>	<u>71.51</u>
Gemini-2.5-flash	<u>92.48</u>	77.44	<u>61.90</u>	31.22	<u>86.67</u>	<u>68.25</u>	86.99	65.75	<u>31.34</u>	<u>4.10</u>	<u>68.40</u>
GPT-4o-audio	85.71	70.68	23.81	15.87	84.76	<u>61.90</u>	71.23	49.32	0.00	4.22	54.77
Open-source models (direct inference with prompt)											
<i>Cascaded models</i>											
Qwen3-8B+Whisper	<u>94.99</u>	68.42	<u>82.54</u>	<u>41.27</u>	85.71	46.67	<u>84.25</u>	47.72	7.46	4.05	64.00
<i>End-to-end models</i>											
Kimi-Audio	78.45	56.89	48.15	22.75	79.05	55.24	76.03	46.80	13.64	3.62	54.94
Qwen2.5-Omni	78.70	35.84	38.62	3.17	65.40	28.57	65.75	26.03	0.00	2.82	39.85
StepAudio2	78.70	48.87	60.32	26.98	53.33	33.33	4.34	1.60	3.12	1.91	34.88
Ours											
VoxMind	98.50	<u>72.18</u>	95.24	<u>38.10</u>	89.52	61.59	<u>80.82</u>	<u>62.33</u>	68.66	3.94	74.57

Table 2: We evaluate model performance using four metrics: TS (Tool Selection accuracy), PF (Parameter Filling accuracy), TU (Tool Usage accuracy), and FC (Feedback Completeness).

Metric		w/o think (1:1)	w/o think (1:0.5)	w/ think (1:1)	w/ think (1:0.5)
Single Task Processing	TS \uparrow	88.72	90.23	<u>90.98</u>	98.50
	PF \uparrow	70.68	<u>71.68</u>	68.42	72.18
Task Decomposition	TS \uparrow	95.24	93.65	<u>94.71</u>	95.24
	PF \uparrow	<u>39.68</u>	36.51	44.44	38.10
Parallel Processing	TS \uparrow	80.00	80.00	<u>80.95</u>	89.52
	PF \uparrow	45.71	<u>59.05</u>	51.43	61.59
Contextual Planning	TS \uparrow	86.99	<u>86.30</u>	84.93	80.82
	PF \uparrow	<u>73.29</u>	75.34	65.75	62.33
Proactive Seeking	TU \uparrow	31.34	37.31	<u>59.70</u>	68.66
Result Feedback	FC \uparrow	3.83	3.98	3.92	<u>3.94</u>
Overall		68.83	70.97	<u>71.97</u>	74.57

Table 3: Ablation Study. Investigate the impact of deep reasoning on agent performance. The metrics are preserved across different training strategies.

evaluator variance, **each model output is evaluated three times and the final score is obtained by averaging**, with detailed evaluation prompts and criteria provided in Appendix F.

We additionally evaluate **VoxMind** on the **VoiceBench** (Chen et al., 2024b) benchmark to verify that general conversational ability is preserved under agentic training.

Finally, to analyze the impact of dynamic tool management, we conduct controlled experiments on a **Gemini-generated cross-domain dataset**, comparing configurations **with and without the auxiliary tool management agent** while varying the number of available tools and measuring **task accuracy and relative inference latency**.

Training details. The experiments were configured with the following key parameters: we adopted 2 pieces of H20-NVLink GPU for model training. The batch size was set to 1, with gradient accumulation steps of 8 to compensate for the small batch size. The learning rate was initialized at $1e-5$, and a cosine learning rate scheduler was employed during training. Other regularization and

optimization settings included a weight decay of 0.01, a maximum gradient norm clipping of 1.0, and the AdamW optimizer. For efficient large-scale model training, we enabled DeepSpeed with the ZeRO-3 strategy, bfloat16 precision, and gradient checkpointing. Further implementation details can be found in Appendix G.

4.2 Results and Analysis

Evaluation of Core Capabilities of Agents. As shown in Table 2, VoxMind achieves SOTA performance with an overall score of 74.57. Compared to the base model StepAudio2 (34.88), VoxMind demonstrates a substantial relative improvement of 113.79%. It also significantly outperforms the top-tier open-source end-to-end model, Kimi-Audio (54.94), and the cascaded system Qwen3-8B + Whisper (64.00), while surpassing even the leading closed-source model, Gemini-2.5-pro (71.51).

Among open-source baselines, the cascaded system (Qwen3-8B + Whisper) notably outperforms end-to-end alternatives such as Kimi-Audio. This suggests that cascaded systems leveraging text-based LLMs maintain an advantage when paralinguistic information and latency are not primary constraints. Additionally, a broader comparison reveals that closed-source models generally outperform open-source baselines, highlighting a discernible gap between community-driven and proprietary models.

Collectively, these results indicate that current end-to-end speech large models still exhibit suboptimal performance on agentic tasks. This highlights the importance of our proposed VoxMind, which serves as a vital contribution to the open-source community.

Ablation Study. Table 3 analyzes the impact of

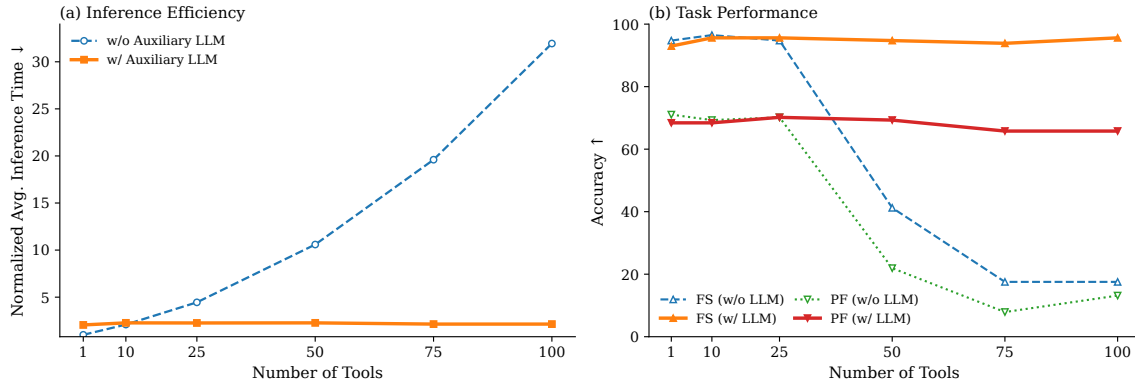


Figure 4: Comparison of inference efficiency and task accuracy with and without the auxiliary LLM across varying tool pool sizes. The auxiliary LLM enables efficient tool-space pruning, significantly reducing inference overhead while maintaining performance.

VoiceBench	Step-Audio-2 (Base)	w/o think (1:0.5)	w/o think (1:1)	w/ think (1:1)	w/ think (1:0.5)
AlpacaEval	4.19	3.38	3.77	<u>4.08</u>	3.98
CommonEval	3.12	3.43	3.75	4.03	<u>3.94</u>
WildVoice	3.36	3.02	3.42	3.79	<u>3.69</u>
SD-QA (USA) / Panda	55.15	49.73	48.28	<u>51.90</u>	49.73
SD-QA (USA) / GPT	52.80	38.34	39.24	44.48	<u>44.85</u>
MMSU	50.82	36.88	47.69	<u>51.61</u>	53.04
OBQA	68.13	56.70	<u>68.79</u>	65.49	71.87
BBH	58.53	50.66	50.25	<u>56.31</u>	54.69
IFEval	39.64	20.74	<u>23.61</u>	17.40	18.83
AdvBench	92.88	87.69	84.62	<u>95.58</u>	100.00
Overall	<u>64.15</u>	54.80	59.72	63.62	64.21

Table 4: Performance comparison on the **VoiceBench** general conversation task between the base model, models with and without deep thinking training, and models trained with different data ratios.

training strategies and data ratios on agent task performance. Without chain-of-thought reasoning (w/o think), increasing the proportion of agent interaction data (shifting from 1:1 to 1:0.5) yields only marginal gains, improving the score from 68.83 to 70.97. This indicates that the direct speech-to-answer paradigm faces a performance bottleneck. Furthermore, Table 4 reveals that this approach comes at the cost of general speech capability, with the VoiceBench score regressing significantly from 59.72 to 54.80.

Conversely, models incorporating deep reasoning (w/ think) demonstrate superior robustness and performance. The w/ think (1:0.5) configuration achieves a peak agent task score of 74.57 (+2.6 points over the 1:1 baseline) and elevates the general evaluation to 64.21, outperforming both the 1:1 variant and the base model (64.15). Notably, while w/o think models suffer substantial regressions on VoiceBench (dropping between 4.43 and 9.35 points), the w/ think variants exhibit negligible

degradation (maximum 0.53 points). This confirms that reasoning capabilities effectively preserve general knowledge while enhancing specialized performance.

These findings suggest that explicit chain-of-thought reasoning is critical for stabilizing training and mitigating the trade-off between domain specialization and general speech proficiency.

Dynamic Tool Management Analysis. As shown in Fig 4(a), when configured with a single tool, VoxMind’s inference time is marginally higher than that of the single-agent approach. However, as the number of tools increases, the single-agent’s inference time exhibits exponential growth, rendering it entirely unsuitable for real-world scenarios involving numerous tools. In contrast, VoxMind maintains stable inference times through its auxiliary agent-based tool management mechanism, achieving true decoupling between tool quantity and inference duration. Experimental results in Fig 4(b) further validate this advantage: as tool scale expands, single-agent inference performance degrades significantly, whereas the VoxMind model consistently sustains stable performance.

5 Conclusion

In this work, we establish a comprehensive definition and theoretical standard for End-to-End Spoken Agents. Building on this foundation, we propose VoxMind, an end-to-end spoken agent capable of intrinsic reasoning and tool use. Experimental results demonstrate that VoxMind significantly outperforms strong baselines on complex agentic tasks, providing a robust theoretical and technical framework for the field.

547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568

569

570
571
572
573
574
575
576
577

578
579
580
581
582
583
584
585

586
587
588
589
590
591
592

593
594
595
596

597
598

Limitations

Despite the advancements presented, two aspects warrant further discussion. First, the core "Think-before-Speak" mechanism, while pivotal for enabling complex reasoning, inherently introduces an inference latency trade-off. The generation of internal reasoning trajectories precedes the final verbal response, inevitably incurring a computational overhead compared to shallow reactive models. We regard this as a necessary trade-off for correctness, yet minimizing this latency remains an objective for future research. Second, regarding dataset construction, the AgentChat dataset relies on synthesizing mature text-based reasoning corpora. Although we implemented rigorous filtering to ensure audio-text alignment, the semantic structure may still reflect the precision of written language rather than the spontaneity and disfluencies characteristic of authentic daily speech. Future iterations will focus on constructing datasets natively rooted in spoken scenarios to better capture the nuances of acoustic pragmatics.

References

Philip Anastassiou, Jiawei Chen, Jitong Chen, Yuanzhe Chen, Zhuo Chen, Ziyi Chen, Jian Cong, Lelai Deng, Chuang Ding, Lu Gao, Mingqing Gong, Peisong Huang, Qingqing Huang, Zhiying Huang, Yuanyuan Huo, Dongya Jia, Chumin Li, Feiya Li, Hui Li, and 27 others. 2024. [Seed-tts: A family of high-quality versatile speech generation models](#). *ArXiv*, abs/2406.02430.

Siddhant Arora, Haidar Khan, Kai Sun, Xin Dong, Sajal Choudhary, Seungwhan Moon, Xinyuan Zhang, Adithya Sagar, Surya Teja Appini, Kaushik Patnaik, Sanat Sharma, Shinji Watanabe, Anuj Kumar, Ahmed Aly, Yue Liu, Florian Metze, and Zhaojiang Lin. 2025. [Stream rag: Instant and accurate spoken dialogue systems with streaming tool usage](#). *ArXiv*, abs/2510.02044.

Wenxi Chen, Ziyang Ma, Ruiqi Yan, Yuzhe Liang, Xiquan Li, Ruiyang Xu, Zhikang Niu, Yanqiao Zhu, Yifan Yang, Zhaxun Liu, Kai Yu, Yuxuan Hu, Jinyu Li, Yan Lu, Shujie Liu, and Xie Chen. 2024a. [Slam-omni: Timbre-controllable voice interaction system with single-stage training](#). In *Annual Meeting of the Association for Computational Linguistics*.

Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T. Tan, and Haizhou Li. 2024b. [Voicebench: Benchmarking llm-based voice assistants](#). *ArXiv*, abs/2410.17196.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind

Tafjord. 2018. [Think you have solved question answering? try arc, the ai2 reasoning challenge](#). *ArXiv*, abs/1803.05457. 599
600
601

Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. [Training verifiers to solve math word problems](#). *ArXiv*, abs/2110.14168. 602
603
604
605
606
607

Sirui Hong, Mingchen Zhuge, Jonathan Chen, Xiawu Zheng, Yuheng Cheng, Ceyao Zhang, Jinlin Wang, Zili Wang, Steven Ka Shing Yau, Zi Hen Lin, Liyang Zhou, Chenyu Ran, Lingfeng Xiao, Chenglin Wu, and Jürgen Schmidhuber. 2023. [Metagpt: Meta programming for a multi-agent collaborative framework](#). In *International Conference on Learning Representations*. 608
609
610
611
612
613
614
615

Shengpeng Ji, Yifu Chen, Minghui Fang, Jialong Zuo, Jingyu Lu, Hanting Wang, Ziyue Jiang, Long Zhou, Shujie Liu, Xize Cheng, and 1 others. 2024. [Wavchat: A survey of spoken dialogue models](#). *arXiv preprint arXiv:2411.13577*. 616
617
618
619
620

KimiTeam, Ding Ding, Zeqian Ju, Yichong Leng, Songxiang Liu, Tong Liu, Zeyu Shang, Kai Shen, Wei Song, Xu Tan, Heyi Tang, Zhengtao Wang, Chu Wei, Yifei Xin, Xinran Xu, Jian-Xiu Yu, Yutao Zhang, Xinyu Zhou, Y. Charles, and 21 others. 2025. [Kimi-audio technical report](#). *ArXiv*, abs/2504.18425. 621
622
623
624
625
626

Gang Li, Jizhong Liu, Heinrich Dinkel, Yadong Niu, Junbo Zhang, and Jian Luan. 2025a. [Reinforcement learning outperforms supervised fine-tuning: A case study on audio question answering](#). *arXiv preprint arXiv:2503.11197*. 627
628
629
630
631

Tianpeng Li, Jun Liu, Tao Zhang, Yuanbo Fang, Da Pan, Mingrui Wang, Zheng Liang, Zehuan Li, Mingan Lin, Guosheng Dong, Jianhua Xu, Haoze Sun, Zenan Zhou, and Weipeng Chen. 2025b. [Baichuan-audio: A unified framework for end-to-end speech interaction](#). *ArXiv*, abs/2502.17239. 632
633
634
635
636
637

Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, Zezhong Wang, Yuxian Wang, Wu Ning, Yutai Hou, Bin Wang, Chuhan Wu, Xinzhi Wang, Yong Liu, Yasheng Wang, and 8 others. 2024. [Toolace: Winning the points of llm function calling](#). *ArXiv*, abs/2409.00920. 638
639
640
641
642
643
644

Zuwei Long, Yunhang Shen, Chaoyou Fu, Heting Gao, Lijiang Li, Peixian Chen, Mengdan Zhang, Hang Shao, Jian Li, Jinlong Peng, Haoyu Cao, Ke Li, Rongrong Ji, and Xing Sun. 2025. [Vita-audio: Fast interleaved cross-modal token generation for efficient large speech-language model](#). *ArXiv*, abs/2505.03739. 645
646
647
648
649
650
651

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, Rongcheng Tu, Xiaoming Luo, Wei Ju, Zhiping Xiao, Yifan Wang, Mengxue Xiao, 652
653
654
655

656	Chenwu Liu, Jingyang Yuan, Shichang Zhang, and 7 others. 2025. Large language model agent: A survey on methodology, applications and challenges . <i>ArXiv</i> , abs/2503.21460.	Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, Bin Zhang, Xiong Wang, Yunfei Chu, and Junyang Lin. 2025a. Qwen2.5-omni technical report . <i>ArXiv</i> , abs/2503.20215.	710 711 712 713 714
660	Tula Masterman, Sandi Besen, Mason Sawtell, and Alex Chao. 2024. The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey . <i>ArXiv</i> , abs/2404.11584.	Jin Xu, Zhifang Guo, Jinzheng He, Hangrui Hu, Ting He, Shuai Bai, Keqin Chen, Jialin Wang, Yang Fan, Kai Dang, and 1 others. 2025b. Qwen2. 5-omni technical report . <i>arXiv preprint arXiv:2503.20215</i> .	715 716 717 718
664	Akshara Prabhakar, Zuxin Liu, Weiran Yao, Jianguo Zhang, Ming Zhu, Shiyu Wang, Zhiwei Liu, Tulika Manoj Awalganekar, Haolin Chen, Thai Hoang, Juan Carlos Niebles, Shelby Heinecke, Huan Wang, Silvio Savarese, and Caiming Xiong. 2025. Apigenmt: Agentic pipeline for multi-turn data generation via simulated agent-human interplay . <i>ArXiv</i> , abs/2504.03601.	Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xianzhong Shi, Ting He, Xinfu Zhu, Yuanjun Lv, Yongqi Wang, Dake Guo, He Wang, Linhan Ma, Pei Zhang, Xinyu Zhang, Hongkun Hao, Zishan Guo, and 19 others. 2025c. Qwen3-omni technical report . <i>ArXiv</i> , abs/2509.17765.	719 720 721 722 723 724 725
672	Yujia Qin, Shi Liang, Yining Ye, Kunlun Zhu, Lan Yan, Ya-Ting Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Runchu Tian, Ruobing Xie, Jie Zhou, Marc H. Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. 2023. Toolllm: Facilitating large language models to master 16000+ real-world apis . <i>ArXiv</i> , abs/2307.16789.	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, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report . <i>ArXiv</i> , abs/2505.09388.	726 727 728 729 730 731 732
679	Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Jirong Wen. 2024. Tool learning with large language models: a survey . <i>Frontiers of Computer Science</i> , 19.	Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models . <i>ArXiv</i> , abs/2210.03629.	733 734 735 736
683	Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision . In <i>International Conference on Machine Learning</i> .	Dong Zhang, Shimin Li, Xin Zhang, Jun Zhan, Pengyu Wang, Yaqian Zhou, and Xipeng Qiu. 2023. Speechgpt: Empowering large language models with intrinsic cross-modal conversational abilities . In <i>Conference on Empirical Methods in Natural Language Processing</i> .	737 738 739 740 741 742
688	Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools . <i>ArXiv</i> , abs/2302.04761.	A Detailed Composition of the Dataset	743
693	Weiting Tan, Xinghua Qu, Ming Tu, Meng Ge, Andy T. Liu, Philipp Koehn, and Lu Lu. 2025. Process-supervised reinforcement learning for interactive multimodal tool-use agents . <i>ArXiv</i> , abs/2509.14480.	A.1 AgentChat Data Details	744
697	Johannes Welbl, Nelson F. Liu, and Matt Gardner. 2017. Crowdsourcing multiple choice science questions . <i>ArXiv</i> , abs/1707.06209.	Table 5 outlines the detailed specifications of the AgentChat-Tool subset. This component integrates established benchmarks, specifically ToolACE (Liu et al., 2024) and APIGen-MT (Prabhakar et al., 2025), with a suite of custom-synthesized datasets designed to enhance specific agentic capabilities. The custom entries address distinct operational phases: tool-select and multi-tool-select focus on single and multi-tool selection accuracy, respectively, while para-filled and parallel-call target precise argument populating and parallel tool execution. Furthermore, searchTool simulates scenarios where the agent proactively requests new tools (active inquiry), and observation is curating to train the model in interpreting and reacting to environmental feedback. Collectively, this subset comprises 14,805 samples with a total audio duration of approximately 109 hours.	745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762
700	Boyong Wu, Chao Yan, Chen Hu, Cheng Yi, Chengli Feng, Fei Tian, Feiyu Shen, Gang Yu, Haoyang Zhang, Jingbei Li, Mingrui Chen, Peng Liu, Wang You, Xiangyu Tony Zhang, Xingyuan Li, Xue jun Yang, Ya lu Deng, Yechang Huang, Yuxin Li, and 82 others. 2025. Step-audio 2 technical report . <i>ArXiv</i> , abs/2507.16632.		
707	Zhifei Xie and Changqiao Wu. 2024. Mini-omni2: Towards open-source gpt-4o with vision, speech and duplex capabilities . <i>ArXiv</i> , abs/2410.11190.		

AgentChat-Tool	Samples	Tool Categories	Avg. Turns	Duration (H)
tool-select	1,237	5,038	1	1.9225
multi-tool-select	1,486	7,240	1	5.1605
para-filled	1,409	3,626	1	4.4508
parallel-call	1,144	4,767	1	2.5235
searchTool	467	2,164	1	0.6172
tool-ace-audio	5,582	10,892	1.0672	26.6220
apigen-mt	791	3,980	7.4355	43.2587
observation	2,465	8,423	2	22.4615
obs_searchtools	224	855	3	2.0525
All	14,805	-	-	109.0692

Table 5: AgentChat-Tool data specifications.



Figure 5: Word clouds of AgentChat data: (Left) tool-interaction data; (Right) general conversational data.

Table 6 presents the composition of the AgentChat-Normal subset, which serves as the foundation for general dialogue and reasoning capabilities. This collection aggregates diverse public benchmarks, including ARC (Challenge and Easy)(Clark et al., 2018), GSM8K(Cobbe et al., 2021), and SciQ (Welbl et al., 2017), to ensure broad coverage of scientific and logical reasoning tasks. Additionally, the dataset includes conversation and course subsets, which are derived from general dialogue corpora and secondary school textbooks, respectively, to maintain domain balance and conversational fluency. The general conversation subset represents the largest portion of our data, totaling 38,681 samples with a cumulative duration of approximately 361 hours.

A.2 Data Word Cloud

Fig 5 presents word clouds of interaction data between users and agents. The left panel displays the word cloud for tool interaction data, while the right panel shows the word cloud for general dialogue data.

A.3 Tool Interaction Data Training Example

As shown in Fig6. This example illustrates a complete spoken interaction between a user and a voice-based assistant. The user query is provided in audio form, followed by a sequence of tool calls and observations used to retrieve relevant information. The assistant then integrates the retrieved results and produces a final spoken response.

AgentChat-Normal	Samples	Tool Categories	Avg. Turns	Duration (H)
ai2_arc-challenge	1,167	3,125	1.0000	12.3334
ai2_arc-easy	1,164	4,819	1.0000	10.8154
conversation	11,259	14,335	1.0000	125.4635
course	19,152	14,357	1.0000	141.9130
gsm8k	1,746	4,395	1.0000	18.4730
multi-conversation	3,171	9,755	2.0000	42.3503
sciq	998	2,707	1.0000	9.4903
who-conversation	24	118	1.0000	0.1096
All	38,681	–	–	360.9485

Table 6: AgentChat-Normal data specifications.

B CoT Construction System Prompt

B.1 CoT Construction For Tool

Fig. 7 illustrates the system prompt used for constructing Chain-of-Thought with tool interaction. The prompt specifies the structure and constraints of the generated reasoning, including:

- **Inputs:** the user query and the corresponding gold tool call from the previous interaction round.
- **Reasoning scope:** a strictly causal, step-by-step explanation starting from the user query, without back-solving from the answer.
- **Tool grounding:** explicit justification of tool selection and parameter instantiation.
- **Constraints:** prohibition of unstated assumptions, bounded reasoning length, and natural language steps.
- **Output format:** a single-line JSON object containing only the reasoning text.

B.2 CoT Construction For General Dialogue

Fig. 8 presents the system prompt used for constructing Chain-of-Thought for general dialogue data. The prompt defines the structure and constraints of the reasoning process, including:

- **Inputs:** the user query and the corresponding gold response.
- **Reasoning scope:** a strictly causal, step-by-step reasoning process starting from the user query.
- **Content focus:** semantic reasoning leading to the response, excluding stylistic or rhetorical considerations.

- **Constraints:** no unstated assumptions or external knowledge, bounded reasoning length, and training-only usage.

- **Output format:** a single-line JSON object containing only the reasoning text.

C CoT Quality Evaluation System Prompt

C.1 CoT Quality Evaluation of Tool Interaction Data

Fig. 9 illustrates the prompt used to evaluate the quality of Chain-of-Thought for tool-based spoken interactions. Given the user query, the corresponding gold tool call, and a candidate Chain-of-Thought, the evaluator assigns a strict score on a 0–10 scale.

The evaluation emphasizes alignment between reasoning and tool usage. In particular, it checks whether the Chain-of-Thought follows a coherent, step-by-step causal structure, correctly explains the selection of the tool and the origin of each parameter, and remains fully consistent with the gold tool call. Additional criteria penalize hallucinated assumptions or invented information, and reward clarity and readability of the reasoning process.

The final score aggregates all criteria into a single numeric value, providing a unified measure of logical soundness, tool grounding, and reasoning quality for tool interaction data.

C.2 CoT Quality Evaluation for General Dialogue Data

As shown in Fig10 .The generated Chain-of-Thought is assessed via a rigorous 0–10 point scoring system that evaluates correctness (0–4) by checking for logical derivation, factual accuracy,

Table 7: Detailed composition of training data at different mixing ratios.

Category	Ratio 1:1 Distribution		Ratio 1:0.5 Distribution	
	Samples	Duration (H)	Samples	Duration (H)
ai2_arc-challenge	334	3.57	173	1.88
ai2_arc-easy	338	3.15	181	1.68
apigen-mt	791	43.26	791	43.26
conversation	3,391	37.65	1,673	18.52
course	5,890	43.58	2,975	21.97
dialog	5,582	26.62	5,582	26.62
gsm8k	543	5.73	271	2.84
multi-conversation	944	12.50	469	6.15
multi-tool-select	1,486	5.16	1,486	5.16
obs	2,465	22.46	2,465	22.46
obs-searchTools	224	2.05	224	2.05
para-filled	1,409	4.45	1,409	4.45
parallel-call	1,144	2.52	1,144	2.52
sciq	293	2.83	155	1.50
tool-gap	467	0.62	467	0.62
tool-select	1,237	1.92	1,237	1.92
who-conversation	13	0.07	5	0.02
Total	26,551	218.14	19,607	163.62

Table 8: Comprising additional modalities and safety training data.

Dataset	Total Dialogs	Tool-free	User Modality (Turns)		Assistant Modality (Turns)		Duration (H)
			Text	Audio	Text	Audio	
No-Tool	2,500	2,500	0	2,717	2,717	0	5.09
Security	556	556	556	0	556	0	0.00
Text	2,500	0	2,713	0	2,713	0	0.00
Total	5,556	3,056	3,269	2,717	5,986	0	5.09

859 and absence of hallucinations; relevance (0–2) by
860 ensuring the reasoning stays tightly focused on the
861 user query; step quality and clarity (0–2) by veri-
862 fying that steps are structured and easy to follow
863 without logical jumps; completeness (0–1) by con-
864 firming all necessary steps are present to justify
865 the final answer; and brevity (0–1) by ensuring the
866 response remains concise and free of unnecessary
867 verbosity.

868 D Tool Usage Necessity Check

869 Fig. 11 shows the system prompt used to assess
870 whether a user query genuinely requires invoking
871 external tools. This check is applied during data

872 cleaning to distinguish queries that depend on exter-
873 nal information or computation from those that can
874 be answered purely through the model’s internal
875 reasoning.

876 The evaluator assigns a score on a 0–4 scale,
877 reflecting increasing degrees of tool dependency.
878 Higher scores indicate that answering the query
879 requires capabilities beyond a standalone language
880 model, such as access to real-time information, pri-
881 vate or external data sources, precise numerical
882 computation, or interaction with external environ-
883 ments. Lower scores correspond to queries that
884 rely on general world knowledge, conceptual under-
885 standing, creative generation, or logical reasoning
886 without external inputs.

887	By explicitly quantifying tool necessity, this step	This procedure ensures rigorous, reproducible	932
888	helps filter out spurious or unnecessary tool use	evaluation and avoids direct string comparison, pro-	933
889	and ensures that tool-invoking examples in the	viding a precise measure of a speech agent’s tool-	934
890	dataset correspond to queries where external tools	call capabilities.	935
891	are meaningfully required.		
892	E Polishing and cleaning of COT	G Training configuration and data	936
		composition	937
893	As shown in Fig12, the Chain-of-Thought com-	G.1 Training Environment Setup	938
894	pression prompt is designed to condense original	All models were trained using the PyTorch frame-	939
895	reasoning into a concise, strictly causal statement	work on NVIDIA GPUs. The development environ-	940
896	within a defined word limit, requiring the model	ment was configured with Python 3.10 , utilizing	941
897	to preserve the logical flow, explicitly justify tool	PyTorch 2.6.0 with CUDA 12.4 .	942
898	selection and parameter sources based on the user’s	G.2 Training Data Composition	943
899	intent and the Gold Tool Call, and output the re-	Core Data Distribution (Ratio 1:1 vs. 1:0.5)	944
900	sult in a strict JSON format without introducing	Table 7 details the composition of our main instruc-	945
901	unsupported assumptions.	tion tuning dataset. We explored two data mixing	946
902	F Evaluation of Core Competencies	strategies to analyze the balance between general	947
903	Figure 13 illustrates the strict evaluation procedure	conversational capabilities and specific agentic tool	948
904	of Gemini-2.5-flash for end-to-end speech agents.	usage:	949
905	The process follows a <i>tool extraction + correctness</i>	• Ratio 1:1 Distribution: This serves as the	950
906	<i>evaluation</i> paradigm, consisting of the following	baseline, utilizing the full extent of our col-	951
907	steps:	lected dataset across all categories.	952
908	1. Tool Extraction: Extract all tool calls from	• Ratio 1:0.5 Distribution: In this setting, we	953
909	both the target and model outputs (including	applied a downsampling strategy to general	954
910	only tool names and parameter name-value	conversational and knowledge-intensive tasks	955
911	pairs), ignoring textual content, formatting,	(e.g., <i>conversation</i> , <i>course</i> , <i>gsm8k</i> , <i>sciq</i>) by	956
912	spaces, quotes, and line breaks. No correct-	approximately 50%. Crucially, categories re-	957
913	ness judgment is performed at this stage.	lated to tool usage, API generation, and obser-	958
914	2. Tool Selection Evaluation: Compare ex-	vation reasoning (e.g., <i>apigen-mt</i> , <i>tool-select</i> ,	959
915	tracted tool names (case-sensitive, ignoring	<i>obs</i> , <i>parallel-call</i>) were preserved at their orig-	960
916	order and leading/trailing spaces). Tool occur-	inal volume to maintain high proficiency in	961
917	rence counts must match exactly; otherwise,	agentic tasks.	962
918	evaluation stops immediately and both tool	Supplementary Data To further enhance the	963
919	selection and parameter filling are marked in-	model’s robustness across modalities and align-	964
920	correct.	ment with safety standards, we incorporated addi-	965
921	3. Parameter Filling Evaluation: Performed	tional datasets as shown in Table 8. This includes:	966
922	only if tool selection is correct. Parameter	• No-Tool: A cross-modal dataset consisting of	967
923	names ignore case and spaces, while param-	2,717 turns where user audio input is paired	968
924	eter values must match exactly. Numeric equiv-	with assistant text output (5.09 hours of au-	969
925	alence and quoting differences are allowed,	dio), designed to improve audio understanding	970
926	and argument order does not affect the evalua-	without triggering tool calls.	971
927	tion.	• Security: A pure text dataset focused on	972
928	4. Output Format: Evaluation results are	safety and reasoning chains.	973
929	strictly returned in JSON format, containing	• Text: Additional standard text-only dialogues	974
930	only two boolean fields: "func-select-correct":	to stabilize language generation performance.	975
931	truelfalse, "param-fill-correct": truelfalse.		

Full Agent–Tool Interaction Example

id: dialog-781

Available Tools

name: headset.user_reviews_analysis

description: Analyzes user reviews for gaming headsets to extract sentiment and common themes.

parameters: headset_model (string), review_sources (array: Amazon, BestBuy, Newegg, Walmart, eBay)

name: headset.feature_availability

description: Checks availability of specific features across regions.

parameters: feature (string), regions (array: North America, Europe, Asia, South America, Australia)

Conversation

role: User

type: audio

content: I'm considering purchasing a new gaming headset for my streaming setup. Could you check whether noise cancellation is available in gaming headsets in Europe and Asia? If so, summarize user sentiment for the HyperX Cloud II from Amazon and eBay.

audio-path: 53ef182d-1a2a-4bc5-9ee8-f2e319bdb329.wav

role: Assistant

type: tool

content: [{ name: headset.feature_availability, arguments: { feature: noise_cancellation, regions: [Europe, Asia] } }]

think: (...)

role: Observation

type: observation

content: [{name: headset.feature_availability, results: {availability: [{region: Europe, is_available: true}], {region: Asia, is_available: true}}}]

role: Assistant

type: tool

content: [{ name: headset.user_reviews_analysis, arguments: { headset_model: HyperX Cloud II, review_sources: [Amazon, eBay] } }]

think: (...)

role: Observation

type: observation

content: [{name: headset.user_reviews_analysis, results: {sentiment_score: 4.5, common_themes: ["comfortable fit", "excellent sound quality", "value for money", "effective noise cancellation"]}}]

role: Assistant

type: audio

content: Reviews indicate highly positive sentiment toward HyperX Cloud II, highlighting comfort, strong sound quality, good value, and effective noise cancellation, making it suitable for streaming.

audio-path: d760bdba-88e7-4bf2-8801-188e9c27b0ad.wav

think: (...)

Figure 6: Tool Interaction Data Training Example.

Chain-of-Thought Construction Prompt for Tool-Based Spoken Interactions

You are a senior reasoning expert.

Goal:

Given the User Query and the Gold Tool Call, write a clear, strictly causal, step-by-step Chain-of-Thought.

Requirements:

- 1) Start from the User Query; do not back-solve from the answer.
- 2) Explain why the selected tool is needed and appropriate.
- 3) For every parameter in the tool call, explain its source and any transformation.
- 4) Do not introduce assumptions not stated in the query.
- 5) Produce 5-12 steps in natural language; keep within \leq THINK_MAX_WORDS words.
- 6) Do NOT output the tool call; output only the reasoning.
- 7) searchTools() indicates that the current tool list cannot meet the task and that additional tools are needed.

Output format:

Output strictly one-line JSON: {"think": "..."}.
No extra text.

User Query:

{(user_q or ").strip() }

Gold Tool Call:

{(assistant_a or ").strip() }

Figure 7: Prompt for constructing chain-of-thought reasoning data for tool interaction.

Chain-of-Thought Construction Prompt for General Dialogue

You are a senior reasoning expert.

Goal:

Given the User Query and the Gold Response, write a clear, strictly causal, step-by-step Chain-of-Thought that explains how the response is derived.

Requirements:

- 1) Start reasoning strictly from the User Query; do not back-solve from the final answer.
- 2) Ensure that each step follows causally from the previous one.
- 3) Do not introduce assumptions or external knowledge not implied by the query.
- 4) Focus on semantic reasoning rather than stylistic or rhetorical choices.
- 5) Produce 5-12 reasoning steps in natural language; keep within \leq THINK_MAX_WORDS words.
- 6) The Chain-of-Thought is used for training purposes only and will not be shown to end users.
- 7) Do NOT output the final answer; output only the reasoning process.

Output format:

Output strictly one-line JSON: {"think": "..."}.
No extra text.

User Query:

{(user_q or ").strip() }

Gold Response:

{(assistant_a or ").strip() }

Figure 8: Prompt for constructing chain-of-thought reasoning data for general dialogue interaction.

Chain-of-thought quality evaluation prompt for tool-based spoken interactions

You are a senior Chain-of-Thought quality evaluator.

Task:

Evaluate the quality of a candidate Chain-of-Thought based on the following inputs:

- 1) User Query
- 2) Gold Tool Call
- 3) Candidate Chain-of-Thought

Scoring criteria (very strict):

[1] Logical soundness (0-3):

Is the reasoning stepwise, coherent, and causally connected?
Are any key reasoning steps missing?

[2] Consistency with the tool call (0-3):

Does the Chain-of-Thought explain why the tool is selected and the source of each tool parameter?
Is it fully consistent with the Gold Tool Call?

[3] No hallucination (0-2):

Does the reasoning avoid inventing facts, assumptions, or parameters not present in the User Query or Gold Tool Call?

[4] Clarity (0-2):

Is the reasoning clear, well-structured, and easy to follow as a step-by-step explanation?

Final score:

The final score is the sum of all criteria, ranging from 0 to 10.

Output format requirements:

Output JSON only; no explanations.

Fields:

```
{"score": <0-10 integer or float>, "reason": "<brief 1-2 sentence justification>"}
```

Inputs:

User Query: {(user_q or ").strip()}}

Gold Tool Call: {(gold_tool_call or ").strip()}}

Candidate Chain-of-Thought: {(candidate_cot or ").strip()}}

Figure 9: Prompt specification for evaluating the quality of Chain-of-Thought in tool-based spoken interactions.

Chain-of-thought quality evaluation prompt for general dialogue

You are a strict Chain-of-Thought quality evaluator.

Task:

Evaluate the quality of a generated Chain-of-Thought (CoT) based on the following inputs:

- 1) User Question
- 2) Model Final Answer
- 3) Generated Chain-of-Thought

Scoring criteria:

[1] Correctness (0-4):

Does the Chain-of-Thought logically derive the final answer?

Is each reasoning step factually correct?

Is the reasoning free from hallucinations or fabricated assumptions?

Scoring guidelines:

4 = fully correct

3 = mostly correct with minor issues

2 = contains some errors but the final answer is still reachable

1 = reasoning is incorrect but coincidentally leads to the correct answer

0 = wrong or nonsensical reasoning

[2] Relevance (0-2):

Does the Chain-of-Thought remain tightly focused on the user question?

Does it avoid unrelated tangents?

Scoring guidelines:

2 = fully relevant

1 = partially relevant

0 = mostly irrelevant

[3] Step quality and clarity (0-2):

Are the reasoning steps clear, structured, and easy to follow?

Are there any unjustified jumps in logic?

Scoring guidelines:

2 = very clear

1 = acceptable clarity

0 = unclear or disorganized

[4] Completeness (0-1):

Does the Chain-of-Thought cover all necessary steps to justify the final answer?

Scoring guidelines:

1 = complete

0 = missing key steps

[5] Brevity and conciseness (0-1):

Is the Chain-of-Thought concise and free of unnecessary verbosity?

Scoring guidelines:

1 = concise

0 = overly long or verbose

Final score:

The total score is the sum of all criteria, ranging from 0 to 10.

Output format requirements (strict):

Output JSON only; no additional text.

Required fields:

```
{
  "correctness": X,
  "relevance": X,
  "clarity": X,
  "completeness": X,
  "brevity": X,
  "total_score": X,
  "keep": "yes" or "no",
  "explanation": "short explanation"
}
```

Inputs:

User Question: {(user_q or "").strip()}

Model Final Answer: {(model_final_answer or "").strip()}

Generated Chain-of-Thought: {(candidate_cot or "").strip()}

Figure 10: Prompt specification for evaluating the quality of Chain-of-Thought in general dialogue settings.

Tool-necessity evaluation prompt for data cleaning

You are a strict evaluator whose job is to determine whether a user query requires calling an external tool (e.g., search engines, calculators, databases, APIs), or can be answered fully by a language model alone.

Inputs:

User Question

Task:

Judge whether a tool call is **necessary** for answering the given user question.

A tool is considered **necessary** only when the question requires:

- 1) Real-time information (e.g., weather, stock prices, news, events, schedules)
- 2) External knowledge not contained in general training data (e.g., private databases, personal files, proprietary datasets)
- 3) Precise numerical computation beyond mental math (e.g., long arithmetic, complex mathematical evaluation)
- 4) Retrieval of specific, unmemorized facts (e.g., obscure identifiers, URLs, tables, large codebases)
- 5) Interaction with an external environment (e.g., search engines, APIs, calculators, file operations)

A tool is **not necessary** when:

- The question asks for explanations, definitions, or conceptual reasoning
- The answer can be inferred using general world knowledge
- The question is creative in nature (e.g., writing, storytelling, opinions, reasoning, code explanation)
- The question requires logical reasoning but no external data
- The question involves math that can be computed manually

Scoring criteria (0-4):

Tool necessity (0-4):

- 4 = Tool is absolutely required; the question cannot be answered without external information
- 3 = Tool is likely required; the language model cannot reliably know the required information
- 2 = Tool is possibly required; the language model might still answer without a tool
- 1 = Tool is probably not required
- 0 = Tool is clearly not required; the question can be answered purely through reasoning

Output format requirements (strict):

Output JSON only.

Required field:

```
{ "tool_necessity": X }.
```

Figure 11: Prompt specification for evaluating tool necessity during data cleaning.

Chain-of-thought compression prompt for tool-based data cleaning

You are a senior reasoning expert.

Goal:

Compress the original Chain-of-Thought into a short but strictly causal reasoning statement, while preserving the original step-by-step logic and using the Gold Tool Call as ground truth.

Requirements:

- 1) Preserve the logical flow of the original reasoning while condensing it into at most {num} English words.
- 2) Start from the user's intent as reflected in the original reasoning.
- 3) Explicitly mention the selected tool and explain why it is appropriate.
- 4) For each parameter in the tool call, briefly indicate its source from the user request or the tool-call text.
- 5) Do not introduce assumptions not supported by the original reasoning.
- 6) Output only the compressed reasoning.
- 7) Output strictly one-line JSON: {"think": "..."} with no extra text.

Inputs:

Original Chain-of-Thought:
{orig_think}

Gold Tool Call (raw, including tags):
{raw_tool_call}

Figure 12: Prompt specification for compressing original Chain-of-Thought annotations in tool-based data cleaning.

Gemini-2.5-flash evaluates core capabilities of end-to-end speech agents

You are a strict evaluation engine. DO NOT compare raw strings.
You MUST first extract tool calls, then evaluate.

MANDATORY PROCEDURE (DO NOT SKIP):

Step 0: Tool Extraction (internal reasoning only)

- From BOTH Target and Output, extract a list of tool calls.
- Each tool call consists ONLY of:
 - (1) tool name
 - (2) parameter name-value pairs
- Ignore all non-tool text.
- Ignore formatting, spacing, quotes, and line breaks.
- DO NOT judge correctness during this step.

Evaluation Order:

1. Tool Selection (ONLY based on extracted tool names)
 - Compare tool names AFTER extraction, not raw text.
 - Tool name = full string before '('.
 - Tool names are case-sensitive; ignore leading/trailing spaces.
 - Tool occurrence counts must match exactly (order does NOT matter).
 - If ANY mismatch exists:
 - * func_select_correct = false
 - * param_fill_correct = false
 - * STOP evaluation immediately.
2. Parameter Filling (ONLY if Tool Selection is correct)
 - Compare parameters ONLY within matched tools.
 - Parameter names ignore case and spaces.
 - Parameter values must match exactly (case-sensitive).
 - Ignore ALL quoting differences:
 - q='Taylor Swift' ≡ q="Taylor Swift" ≡ q=Taylor Swift
 - Numeric equivalence:
 - 42 ≡ 42.0
 - Argument order does NOT matter.

STRICT OUTPUT FORMAT:

Return ONLY the following JSON.

No explanation, no markdown, no extra text:

```
{"func_select_correct": true|false, "param_fill_correct": true|false}
```

Target:

```
{t}
```

Output:

```
{o}
```

Figure 13: Prompt specification for strict, extraction-based evaluation of tool-call correctness.