From Text to Voice: A Reproducible and Verifiable Framework for Evaluating Tool Calling LLM Agents

Anonymous authors
Paper under double-blind review

Abstract

Voice agents increasingly require reliable tool use from speech, whereas prominent tool-calling benchmarks remain text-based. We study whether verified text benchmarks can be converted into controlled audio-based tool calling evaluations without re-annotating the tool schema and gold labels. Our dataset-agnostic framework uses text-to-speech, speaker variation, and environmental noise to create paired text–audio instances while preserving the original dataset annotations. Based on extensive evaluation of 7 omni-modal models on audio-converted versions of Confetti and When2Call, our framework demonstrates that the performance is strongly model- and task-dependent: Gemini-3.1-Flash-Live obtains the highest Confetti score (70.4), whereas GPT-Realtime-1.5 performs best on When2Call (71.9). On Confetti, the text-to-voice gap ranges from 1.8 points for Qwen3-Omni to 4.8 points for GPT-Realtime-1.5. A targeted analysis of failure cases demonstrates that degradations most often reflect misunderstandings of argument values in the speech. Considering real-world deployment scenarios, we further report text-only results, an ambiguity-based reformulation stress test, and a reference-free LLM-as-judge protocol validated against human preferences. Notably, we find that open-source Qwen3 judges with at least 8B parameters exceed 80% agreement with proprietary judges, supporting privacy-preserving evaluation. Overall, our framework provides a verifiable and reproducible first-stage diagnostic that complements purpose-built audio corpora.

1 Introduction

The rapid advancement of Large Language Models (LLMs) has enabled sophisticated tool calling and function execution capabilities (Schick et al., 2023; Qin et al., 2023; 2024; Qu et al., 2025), in which models identify when to use external functions, determine appropriate parameters, and integrate results into coherent responses. Tool calling has become a cornerstone of modern Agentic AI systems with benchmarks like Confetti (Alkhouli et al., 2025), When2Call (Ross et al., 2025), and BFCL (Patil et al., 2025), have already absorbed the substantial cost of validating tool schemas, curating gold labels, and designing scoring protocols.

Yet in customer support, where interactions often occur via voice (Rzepka et al., 2022; Laskar et al., 2025), these text benchmarks cannot be applied directly. Practitioners deploying voice agents face a concrete architectural choice: a *cascade* pipeline that runs an Asynchronous Speech Recognition (ASR) system followed by a text LLM (Peng et al., 2025), or an *end-to-end omni-modal LLM* that processes speech directly and emits tool calls without an intermediate transcript (Jiang et al., 2025). Cascades usually leverage strong text-only LLMs, expose transcripts for debugging, and make failures easier to localize across ASR, tool selection, and argument generation (Lin et al., 2025). However, they can propagate ASR errors, lose acoustic cues, and increase latency (Yang et al., 2023). End-to-end omni-modal models avoid the transcription bottleneck, but their tool-calling reliability under realistic audio conditions remains insufficiently characterized.

Existing audio function-calling benchmarks (Salesforce AI Research and Berkeley, 2025; Jain et al., 2025) mainly follow a dataset-construction pattern: build a new audio corpus, define new tools, annotate new labels, and report a leaderboard. Although useful, such benchmarks are often restricted to the domains and

tool schemas chosen during construction. This is less helpful for organizations that already have validated domain-specific text benchmarks and need to know how those tasks behave when deployed over voice.

To this end, we study whether existing text-based tool-calling benchmarks can be systematically converted into speech benchmarks while preserving their gold labels and tool schemas. This enables paired text–audio evaluation, measures the degradation introduced by speech, and provides an initial diagnostic for comparing the cascade architecture with end-to-end omni-modal models. More specifically, we introduce a dataset-agnostic framework that converts any text-based tool-calling benchmark into a controlled audio evaluation using off-the-shelf TTS models (Tan et al., 2021), while preserving the original tool schemas, gold labels, and evaluation protocols. Rather than proposing another standalone audio benchmark, our contribution is a reusable recipe that enterprises can apply to proprietary text-based tool-calling data. We treat TTS-generated speech as a controlled approximation of voice-based interaction, which enables us to isolate text–audio and cascade–omni performance gaps. Compared with building audio benchmarks from scratch, this design has three advantages: it preserves gold labels and tool schema, creates paired text–audio instances for failure analysis, and can be applied to proprietary enterprise tool catalogs and text logs.

Using this framework, we evaluate seven omni-modal models across four providers and multiple model tiers, including GPT-Realtime[1] (4o, Mini, 1.5), Gemini-Flash-Live[2] (2.5 and 3.1), Qwen3-Omni (Xu et al., 2025b), and Phi-4-Multimodal (Abouelenin et al., 2025). We compare these systems with text-only baselines on *Confetti* and *When2Call* under clean and noisy audio with diverse voices. Since the cascade architecture depends on the strength of the downstream text LLM, we further conduct a text-mode scaling analysis of Qwen3 models from 0.6B to 32B parameters. We also include an ambiguity-based query reformulation stress test and an error analysis of the text-to-voice performance gap. Finally, because production data often lacks gold annotations, we assess reference-free LLM-as-judge protocols for tool-calling evaluation using both proprietary and open judges (Gu et al., 2024). Our results show that converting trusted text benchmarks gives a signal on architectural choices and failure modes that fixed audio corpora may not isolate directly. We argue that audio-converted versions of trusted text benchmarks can complement purpose-built audio datasets.

Our major contributions are summarized below:

i. **Benchmark conversion for verifiable audio tool-calling evaluation.** We introduce a dataset-agnostic framework for converting existing text-based tool-calling benchmarks into controlled audio evaluations using TTS, voice variation, and environmental noise. Unlike fully new audio benchmarks, this approach preserves the original tool schemas, gold labels, and verifiable scoring protocol.
ii. **A diagnostic protocol for paired text–audio comparison.** By evaluating matched text and audio versions of the same tool-calling instances, our framework measures voice-input degradation relative to a known text baseline and provides a practical signal for modality-induced failures.
iii. **Model-level evidence for cascade-vs-omni diagnostics.** We show that audio tool-calling performance is model- and task-dependent, not determined by architecture alone. Model rankings change across benchmarks, and the Confetti text-to-voice gap ranges from 1.8 to 4.8 points across various models, motivating model- and task-specific cascade-vs-omni diagnostics.
iv. **An error decomposition enabled by paired benchmark conversion.** Because every audio instance has a paired text version with the same gold label, we isolate cases where a model succeeds on text but fails on the corresponding audio input. This enables a counterfactual error analysis showing that audio-induced failures often preserve the broad tool-call structure but fail on argument values, especially for Gemini-3.1-Flash-Live and GPT-Realtime-1.5.
v. **A reference-free LLM-as-judge protocol for production-style evaluation.** Since production data often lacks gold annotations, we assess reference-free LLM-as-judge protocols using both proprietary and open judges. Proprietary judges show broadly stable model-level rankings, and a human preference study finds no clear preference between GPT-5 and Gemini-2.5-Pro. We also observe that open-source Qwen3 judges with at least 8B parameters reach over 80% agreement with proprietary judges, suggesting a path toward privacy-preserving evaluation with open judges.

As a secondary contribution, our converted datasets and evaluation scripts will be made publicly available.

[1] https://developers.openai.com/api/docs/guides/realtime
[2] https://ai.google.dev/gemini-api/docs/live-api

2 Related Work

Tool Calling using LLMs: Tool calling extends LLM capabilities by enabling interaction with external APIs and computational tools (Mialon et al., 2023). Prior work has explored structured API induction via fine-tuning (Li et al., 2023; Xu et al., 2023; Schick et al., 2023; Patil et al., 2024) and introduced large-scale benchmarks such as ToolBench and API-Bank (Qin et al., 2023; Li et al., 2023), along with synthetic scaling approaches like ToolAlpaca (Tang et al., 2023). Domain-specific benchmarks target areas such as mathematics (Gou et al., 2023) and customer support (Alkhouli et al., 2025; Ross et al., 2025), while τ-Bench provides unified agentic evaluation (Yao et al., 2025). Crucially, these benchmarks have already absorbed high cost in validating tool specifications, curating gold parameter values, and designing scoring protocols. However, they remain primarily text-only and cannot directly evaluate omni-modal models that operate on speech. Our framework closes this gap by transforming existing text-based tool-calling benchmarks into audio evaluations while preserving their validated tool schemas, gold labels, and scoring protocols.

Omni-Modal Large Language Models: Omni-modal LLMs extend text-only systems to unified reasoning across text, vision, and audio (Jiang et al., 2025). Unlike cascade architectures that rely on speech-to-text pipelines, end-to-end omni-modal models process audio natively and can reduce pipeline complexity. Both proprietary (e.g., GPT-4o-Realtime, Gemini-Live) and open-source models (Yao et al., 2024; Zeng et al., 2024; Xu et al., 2025a; Ding et al., 2025; Xu et al., 2025b) support such capabilities. Nevertheless, their reliability for structured tool calling remains unsettled, which we aim to address using our proposed framework.

Audio Understanding Benchmarks: Most of the existing audio benchmarks focus on perception tasks such as ASR (Panayotov et al., 2015; Ardila et al., 2020), captioning (Kim et al., 2019; Drossos et al., 2020), and event classification (Gemmeke et al., 2017; Huang et al., 2024). More recent benchmarks (e.g., VoiceBench, WildSpeech-Bench) assess spoken interaction robustness (Chen et al., 2024; Zhang et al., 2025). These resources are important for speech robustness, but they do not provide the tool-calling evaluation setup, which is the focus of this work.

Tool Calling Audio Benchmarks: Evaluation is critical for LLM deployment (Laskar et al., 2023a; 2024a), and recent work has begun evaluating tool calling from speech, including BFCL-v4's audio tier (Patil et al., 2025; Salesforce AI Research and Berkeley, 2025), as well as benchmarks like VoiceAgentBench (Jain et al., 2025) These benchmarks construct new audio corpora in selected domains and report the model performance. Our work is complementary: instead of constructing another fixed audio corpus, we convert existing text-based tool-calling benchmarks into paired speech evaluations. This design supports deployment-oriented analysis in three ways. First, it enables system-level comparison: practitioners can evaluate specific ASR→text-LLM cascade pipelines and omni-modal models on the same tasks, rather than assuming one architecture is always better. Second, each audio example has a clean-text counterpart with the same gold label, allowing direct measurement of the text-to-audio gap and instance-level error decomposition. Third, the framework is portable to proprietary enterprise tool catalogs and text logs, enabling in-house audio evaluation on annotated datasets. Therefore, existing benchmarks answer the question "how do current SpeechLMs perform on a fixed audio corpus for tool calling?", while we answer "how should an enterprise team decide between cascade and omni on its own data for agentic tasks, and what does the modality shift actually cost?"

3 Methodology

3.1 Text-to-Speech Conversion Pipeline

We implement a systematic TTS conversion pipeline with three commercially available TTS models that can be accessed through API endpoints:

(i) Gemini-2.5-Flash-TTS[3]: Google's efficient, low-latency TTS model.

(ii) Gemini-2.5-Pro-TTS[4]: A higher-capacity Gemini TTS system optimized for speech generation quality.

[3] https://docs.cloud.google.com/text-to-speech/docs/gemini-tts#gemini-2-5-flash-tts
[4] https://docs.cloud.google.com/text-to-speech/docs/gemini-tts#gemini-2-5-pro-tts

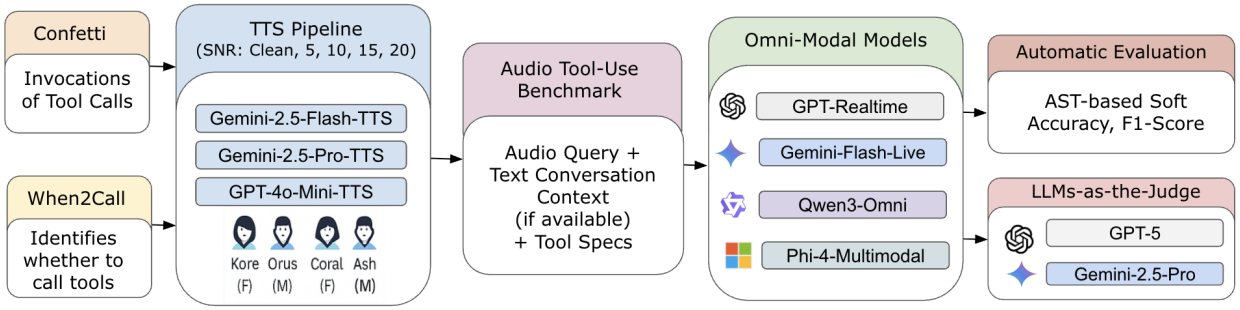


Figure 1: An overview of our methodology for converting text-based tool datasets into audio benchmarks for tool-calling evaluation. The pipeline uses text-to-speech (TTS) models (GPT-4o-Mini-TTS and Gemini-2.5-TTS) to generate diverse audio queries with different voices and genders, which are then processed by omni-modal LLMs and evaluated via automatic evaluation or LLM Judge.

(iii) GPT-4o-Mini-TTS[5]: A compact OpenAI TTS model used as a cross-provider alternative.

By comparing Gemini-2.5-Flash-TTS with the higher-capacity Gemini-2.5-Pro-TTS, we examine whether increased model scale and associated cost translate into measurable gains in synthesis quality and robustness under diverse noisy environments. Additionally, GPT-4o-Mini-TTS serves as a cross-provider alternative to Gemini-2.5-Flash-TTS, enabling us to examine whether compact models from different providers exhibit performance variation under comparable efficiency constraints.

Speaker Variation: To assess model robustness across speaker characteristics, we select distinct voices from each TTS provider for both genders. For Gemini voices, *Kore* represents female voice, while *Orus* represents male. For GPT-4o-Mini, *Ash* represents male voice and *Coral* represents female.

Environmental Noise Injection: To simulate controlled deployment-like acoustic conditions, we inject environmental noise from the DEMAND dataset (Thiemann et al., 2013), which contains multi-channel recordings of acoustic noise across diverse environments. We sample noise segments from settings such as cars, buses, traffic, cafés, kitchens, hallways, meeting rooms, and living rooms, and mix them with clean speech. To evaluate robustness under varying noise levels, we construct separate subsets by adding noise at 5, 10, 15, and 20 dB signal-to-noise ratios (SNRs). Higher SNR values correspond to cleaner audio, reflecting a larger proportion of clean signal energy relative to noise.

3.2 TTS Audio as a Controlled First-Stage Evaluation

Our benchmark is designed as an early-stage evaluation tool for companies making decisions about deploying voice-enabled LLM agents. This is especially useful for teams that already have annotated text transcripts and tool logs: our TTS conversion pipeline provides a low-friction way to test whether a candidate voice-agent stack is likely to preserve tool-calling behavior when the input changes from text to speech. We therefore treat TTS audio as an *optimistic deployment proxy* and first-stage evaluation strategy, rather than a replacement for spontaneous real calls. The key design choice is to evaluate each task in two aligned conditions. In the *text-only* condition, the original text input is given directly to the LLM, measuring the model's downstream ability to use the correct tool when speech processing is not involved. In the *direct-audio* condition, the TTS version of the same input is given to an omni-modal model, measuring whether the model can perform the same tool-calling task from speech. Comparing the text and audio versions of the same examples allows us to estimate the performance loss caused by changing the input modality from text to speech while keeping the task, tool schema, and gold labels fixed. This comparison thereby provides a practical signal for whether an existing text benchmark can support controlled audio evaluation and offers an initial indication of how direct omni-modal inference compares with an ASR→text cascade.. The claim is intentionally conservative: if controlled TTS audio already exposes substantial failures, spontaneous real-world speech is likely to pose an even harder setting.

[5] https://platform.openai.com/docs/models/GPT-4o-Mini-TTS

3.3 Preserving Verifiability through Benchmark Conversion

Building an audio tool-calling benchmark from scratch is costly because it requires defining tool schemas, annotating the correct tool calls and arguments, deciding when a tool call is required, and designing reliable evaluation protocols. These steps are especially challenging for speech input, where evaluation must account for acoustic variation, speaker differences, and possible transcription variability in cascade systems (Sofer et al., 2025). At the same time, text-based tool-calling resources are often more readily available and have already absorbed much of this cost. For instance, public benchmarks such as ToolBench, Confetti, When2Call, and BFCL already provide validated tool specifications, gold parameter values, decision labels, and established scorers (Qin et al., 2023; Alkhouli et al., 2025; Ross et al., 2025; Patil et al., 2025); many enterprise teams similarly maintain text transcripts, tool logs, API schemas, and text-based evaluation pipelines. Our conversion pipeline reuses these resources rather than rebuilding them from scratch.

This design has two practical benefits. First, it preserves verifiability: the converted audio benchmark inherits the source benchmark's tool schemas, gold labels, and evaluation protocol. Second, it enables a controlled text-to-speech comparison: each audio input is generated from an existing text instance, allowing us to measure how performance changes when the same task is presented as speech. The conversion is not merely a cheaper data-generation strategy; it preserves the source benchmark's tool schemas, gold labels, and scoring protocol while testing the effect of changing only the input modality. As the source benchmark improves through better labels, harder splits, or broader tool coverage, the converted audio benchmark can improve with it. We therefore view audio conversion as complementary to purpose-built audio datasets. Speech-native corpora are valuable for studying naturally spoken interactions, while conversion is useful for testing whether existing domain-specific text-based tool-calling tasks remain reliable under voice input.

3.4 Datasets

We employ two publicly available text-based tool-calling benchmarks that are representative of customer-support and conversational agent scenarios for our audio conversion pipeline. While we select two datasets in this paper, our audio construction pipeline is dataset-agnostic: any text-based tool-calling benchmark can be integrated into our framework (see Figure 1) without any architectural modification.

(i) Confetti (Alkhouli et al., 2025): Confetti is a high-quality tool-calling dataset covering diverse real-world conversational settings. Each instance in this dataset includes: (i) a natural language user query requiring tool invocation, (ii) a multi-turn conversational context, and (iii) tool/API documentation with function signatures and parameter specifications. Since our evaluation focuses on tool execution correctness, we consider only those instances that require an explicit tool call, resulting in a filtered subset of 313 examples.

(ii) When2Call (Ross et al., 2025): This dataset evaluates the model's ability to determine whether a tool call is necessary, as opposed to directly responding to the user or requesting clarification. Each instance consists of: (i) a user utterance that may or may not require tool invocation, and (ii) the corresponding API/tool specifications. We use the non-MCQ subset of the dataset, which contains 300 instances.

These benchmarks provide diverse evaluation perspectives. Confetti measures function selection and parameter extraction accuracy, whereas When2Call assesses the decision-making capability required to determine whether to call a tool or not.

3.5 Omni-Modal Models

We primarily evaluate seven omni-modal models from four providers, spanning multiple model tiers.

GPT-4o-Realtime: GPT-4o-Realtime[6] is designed for real-time voice-to-voice interaction. It processes audio input natively and generates streaming speech responses alongside direct tool calling from voice input.

GPT-Realtime-1.5: This Realtime[7] model is an updated version of GPT-4o-Realtime. We evaluate it through the same streaming interface that we used for GPT-4o-Realtime.

[6] https://developers.openai.com/api/docs/models/gpt-4o-realtime-preview
[7] https://developers.openai.com/api/docs/models/gpt-realtime-1.5

GPT-Realtime-Mini: We evaluate this compact OpenAI Realtime[8] variant to measure whether a lower-cost model condition preserves audio tool-calling behavior.

Gemini-2.5-Flash-Live: We use the Gemini-2.5-Flash-Native-Audio model via Google's Live API[9]. It is a real-time multimodal model optimized for low-latency interactive applications, supporting native audio input, streaming speech generation, and structured function calling.

Gemini-3.1-Flash-Live: This is the latest omni-modal model from Google[10], which we evaluate through the same Live API interface we used for Gemini-2.5-Flash-Live.

Qwen3-Omni-30B-A3B-Instruct: Qwen3-Omni (Xu et al., 2025b) is an end-to-end open omni-modal model built by leveraging the Thinker–Talker (Xu et al., 2025a) and the Mixture-of-Experts architecture (Shazeer et al., 2017). The Thinker module handles reasoning and text generation, while the Talker module enables streaming speech generation. The model supports multimodal inputs (text, image, audio, video), streaming interaction with low latency, and function calling.

Phi-4-Multimodal: Microsoft's Phi-4-Multimodal-Instruct (Abouelenin et al., 2025) is a compact open-source omni-modal model (5.6B parameters) supporting text, image, and audio inputs. We include it as an additional compact open-source baseline for audio tool-calling behavior.

4 Experiments

In this section, we first demonstrate the implementation details, followed by the evaluation settings. Finally, we discuss the experimental results.

4.1 Implementation

All audio inputs are stored in `.WAV` format and converted to 16 kHz mono 16-bit PCM before streaming to model endpoints through WebSocket connections. We implement the OpenAI models (`GPT-4o-Realtime`, `GPT-Realtime-1.5`, and `GPT-Realtime-Mini`) via the OpenAI Realtime API using WebSocket streaming. The Gemini models (`Gemini-2.5-Flash-Live` and `Gemini-3.1-Flash-Live`) are implemented via the Google GenAI Live API[11]. For `Qwen3-Omni`, we use the *Qwen3-Omni-30B-A3B-Instruct* checkpoint and run inference using HuggingFace (Wolf et al., 2020). For `Phi-4-Multimodal`, we also use the *Phi-4-multimodal-instruct* checkpoint from HuggingFace and run inference using it. For all the text-only LLMs, we use the Qwen-3 series (Yang et al., 2025) models and run inference using vLLM (Kwon et al., 2023). We use default decoding parameters for all models, except that temperature is set to 0.6 when the parameter is supported. The prompts used for response generation are provided in Appendix A.2.

4.2 Evaluation Settings

In When2Call, we assess whether the model response is a tool call or not by comparing the model-generated response against the ground truth using a parsing script (Laskar et al., 2024b) and report the F1-score.

In Confetti, we follow the standard evaluation protocol from the original paper (Alkhouli et al., 2025) to assess whether the model can accurately invoke the functions with the correct parameters. Specifically, we compare the model predicted function calls with the ground truth using an AST-based soft accuracy metric where the predicted function name and non-string parameter values are scored using exact match, while string parameter values are scored using AlignScore (Zha et al., 2023)). Given the difficulty in evaluating the tool calling outputs generated by LLMs in real-world settings due to the unavailability of annotated reference labels, we also conduct a reference-free evaluation in Confetti using the following two LLMs-as-judge models (Zheng et al., 2023): (i) GPT-5 (OpenAI, 2025), (ii) Gemini-2.5-Pro (Comanici et al., 2025).

[8] `https://developers.openai.com/api/docs/models/gpt-realtime-mini`
[9] `https://docs.cloud.google.com/vertex-ai/generative-ai/docs/models/gemini/2-5-flash-live-api`
[10] `https://ai.google.dev/gemini-api/docs/models/gemini-3.1-flash-live-preview`
[11] `https://ai.google.dev/gemini-api/docs/live-api`

	Clean		SNR 5		SNR 10		SNR 15		SNR 20	
TTS Model	**UTMOS ↑**	**WER ↓**	**UTMOS ↑**	**WER ↓**	**UTMOS ↑**	**WER ↓**	**UTMOS ↑**	**WER ↓**	**UTMOS ↑**	**WER ↓**
Gemini-2.5-Flash	**3.51**	**4.75**	2.81	7.15	2.88	7.35	2.96	7.34	3.07	7.33
Gemini-2.5-Pro	**3.44**	**4.93**	2.75	8.19	2.79	8.15	2.85	8.18	2.99	8.17
GPT-4o-Mini	**3.48**	**3.38**	2.84	4.34	2.87	4.17	2.92	4.25	3.00	4.19

Table 1: TTS quality across the *Confetti* and *When2Call* datasets based on average across all voice types and genders. Higher UTMOS indicates better perceptual quality, while lower WER indicates better recognition accuracy. Bold denotes the best scores.

TTS Model	**Voice**	**GPT-4o Realtime**	**GPT-1.5 Realtime**	**GPT-Mini Realtime**	**Gemini-2.5 Flash-Live**	**Gemini-3.1 Flash-Live**	**Qwen3-Omni 30B-A3B**	**Phi-4 MM**
Gemini-2.5-Flash	Kore (F)	51.0	59.4	40.4	29.5	69.6	61.6	24.3
Gemini-2.5-Flash	Orus (M)	50.3	56.8	39.5	25.5	68.5	59.3	23.3
Gemini-2.5-Pro	Kore (F)	47.7	58.8	39.7	21.4	69.9	60.0	23.5
Gemini-2.5-Pro	Orus (M)	46.7	54.0	40.3	23.1	72.7	59.3	21.4
GPT-4o-Mini	Coral (F)	55.9	62.1	45.5	20.0	71.2	60.5	22.4
GPT-4o-Mini	Ash (M)	55.7	64.0	42.9	19.2	70.2	61.9	24.7
Average		51.2	59.2	41.4	23.1	**70.4**	60.4	23.3

Table 2: AST soft accuracy of omni-modal models on *Confetti* across six TTS configurations.

4.3 Results and Discussion

We first report TTS quality evaluation results and then present omni-modal model performance on Confetti and When2Call datasets.

4.3.1 TTS Performance

To evaluate synthetic speech quality, we use the UTokyo-SaruLab MOS Prediction System (UTMOSv2) (Baba et al., 2024), which predicts the mean opinion score (MOS) for speech naturalness. Higher UTMOS scores indicate more natural synthetic speech. To assess intelligibility, we decode the synthetic audio using Whisper large-v3 (Radford et al., 2023) and compute word error rate (WER), where lower WER indicates better intelligibility. We evaluate the TTS quality and report the results in Table 1.

Across clean and noisy conditions, GPT-4o-Mini obtains the lowest WER among the evaluated TTS models while maintaining competitive naturalness scores. Under clean conditions, Gemini-2.5-Flash-TTS obtains the highest UTMOS score, although the margin over GPT-4o-Mini is small. As expected, audio quality decreases under noise, with larger drops at lower SNR values. The relative ordering of TTS models is nevertheless stable across SNR levels, and GPT-4o-Mini maintains strong intelligibility under degraded audio conditions.

We use WER only as an audio-quality diagnostic and do not filter examples by ASR output; all generated samples are retained in the benchmark. To verify that the synthesized audio remains faithful to the source utterances beyond automatic metrics, we additionally conduct a human intelligibility check on 300 randomly sampled clips, split evenly between clean and noisy audio, using two domain experts having expertise in Natural Language Processing (NLP) and Speech Processing. Human evaluators mark 97.7% of clean samples and 94.3% of noisy samples as content-faithful, indicating that the benchmark largely preserves the intended query semantics while still exposing models to controlled audio degradation.

4.3.2 Omni-Modal Model Performance

Performance across TTS configurations: Tables 2 and 3 present tool-calling performance across six TTS configurations (three TTS systems × two voice types). Performance on Confetti is measured with AST-based soft accuracy, while When2Call is evaluated in terms of F1-score. The results show substantial model- and task-level variation. On Confetti (Table 2), Gemini-3.1-Flash-Live obtains the highest average accuracy (70.4%), followed by Qwen3-Omni (60.4%), GPT-Realtime-1.5 (59.2%), GPT-4o-Realtime (51.2%), GPT-Realtime-Mini (41.4%), Phi-4-Multimodal (23.3%), and Gemini-2.5-Flash-Live (23.1%). On When2Call (Table 3), the relative ranking for the top performing models changes: GPT-Realtime-1.5 leads at 71.9%,

TTS Model	Voice	GPT-4o Realtime	GPT-1.5 Realtime	GPT-Mini Realtime	Gemini-2.5 Flash-Live	Gemini-3.1 Flash-Live	Qwen3 Omni	Phi-4 MM
Gemini-2.5-Flash	Kore (F)	68.9	70.2	74.5	60.8	65.7	58.3	54.0
Gemini-2.5-Flash	Orus (M)	70.7	71.0	65.4	57.0	60.0	61.2	53.4
Gemini-2.5-Pro	Kore (F)	70.3	72.6	68.7	51.0	62.7	60.6	52.6
Gemini-2.5-Pro	Orus (M)	70.1	66.1	63.3	57.0	62.8	60.3	50.6
GPT-4o-Mini	Coral (F)	69.3	76.8	69.7	58.9	62.8	61.3	54.7
GPT-4o-Mini	Ash (M)	71.4	74.6	70.2	56.6	66.7	61.0	56.1
Average		70.1	**71.9**	68.6	56.9	63.4	60.4	53.6

Table 3: F1 score of omni-modal models on *When2Call* across six TTS configurations.

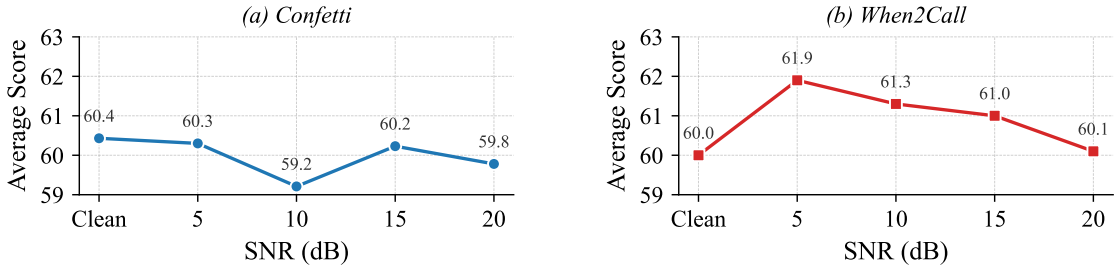


Figure 2: Average performance of Qwen3-Omni across SNR levels, aggregated over all TTS models and voices.

followed by GPT-Realtime-4o (70.1%) and GPT-Realtime-Mini (68.6%). Gemini-3.1-Flash-Live, which achieved the best results in Confetti, ranked 4th in When2call (63.4%), followed by Qwen3-Omni (60.4%). Similar to Confetti, Gemini-2.5-Flash-Live and Phi-4-Multimodal performed the worst, achieving F1-scores of 56.9% and 53.6%, respectively. The shifts in the overall rankings of different models indicate that audio tool-calling performance cannot be characterized by architecture alone or by evaluation on a single dataset; model choice, dataset, and task formulation all affect outcomes.

Robustness under noisy conditions: We further investigate the robustness to environmental noise. For this, we use the Qwen3-Omni model, which achieves balanced performance across both datasets. Based on the results presented in Figure 2, we observe that its accuracy remains relatively stable across SNR levels, with only modest degradation as noise increases. This result suggests that, under the controlled TTS conditions, moderate environmental noise is not the primary source of performance loss for this model.

4.3.3 Cascade vs. Omni Performance Comparison

We compare three input settings that correspond to common deployment choices: *clean text*, where the original text input is given directly to the LLM; *direct voice*, where the TTS audio input is given directly to an omni-modal model to measure native speech-based tool calling; and the *ASR-to-text cascade*, where generated audio is first transcribed and then passed to a text-based LLM for tool calling. Overall, clean-text performance estimates the model's tool-calling ceiling, direct voice measures how well that capability transfers to speech input, and the cascade setting tests whether transcription plus text-based reasoning is sufficient. For transcription, we use the GPT-4o-Transcribe[12] model.

Table 4 reports the results on Confetti for the following representative models: Gemini-3.1-Flash-Live, GPT-Realtime-1.5, and Qwen3-Omni-30B-A3B-Instruct. We restrict our model selection to only real-time omni-modal models because voice agents ensure interactive deployment settings by processing speech with low latency and execute function calls during the live conversation. From Table 4, we find that the text-to-voice gap varies substantially across models: Gemini-3.1-Flash-Live loses 2.6 points from clean text to direct voice (73.0 to 70.4), Qwen3-Omni loses 1.8 points (62.2 to 60.4), and GPT-Realtime-1.5 loses 4.8 points (64.0 to 59.2). This shows that the cost of moving from text to speech is model-dependent rather than a fixed property of omni-modal inference. The cascade results further show that neither architecture uniformly dominates. For Gemini-3.1-Flash-Live, the cascade slightly outperforms direct voice by 0.9 points (71.3 vs. 70.4), suggesting

[12] https://developers.openai.com/api/docs/models/gpt-4o-transcribe

Model	Clean Text	Direct Voice	Cascade (ASR→text)	Δ_{TV}	Δ_{TC}	Δ_{CV}
Gemini-3.1-Flash-Live	**73.0**	70.4	71.3	2.6	1.7	0.9
GPT-Realtime-1.5	64.0	59.2	58.8	4.8	5.2	-0.4
Qwen3-Omni	62.2	60.4	58.9	1.8	3.3	-1.5

Table 4: Performance on Confetti across clean text-only, direct voice, and ASR→text cascade settings. Δ_{TV} measures the text-to-voice gap, Δ_{TC} measures the text-to-cascade gap, and Δ_{CV} compares cascade against direct voice (Clean Text − Direct Voice, Clean Text − Cascade, and Cascade − Direct Voice, respectively).

Model	Failure pairs	Decision	Argument value	Argument schema	Tool selection
Gemini-3.1-Flash-Live	229	25.8%	57.2%	5.2%	11.8%
GPT-Realtime-1.5	254	37.4%	54.3%	2.8%	5.5%
Qwen3-Omni	329	30.4%	39.5%	14.6%	15.5%

Table 5: Error decomposition on the *Confetti* dataset. Each row pools the six TTS configurations and includes paired failure cases where the model exactly matches the gold tool call in text mode but fails on the corresponding audio input. *Decision* errors indicate incorrect high-level tool-use behavior; *argument value* errors indicate incorrect or incomplete field values despite the correct tool/field structure; *argument schema* errors indicate missing, invalid, or misassigned arguments; and *tool selection* errors indicate invocation of the wrong API.

that ASR followed by a strong text-mode model can sometimes recover more of the clean-text performance. In contrast, direct voice performs slightly better than the cascade for GPT-Realtime-1.5 by 0.4 points and for Qwen3-Omni by 1.5 points. These differences are modest but important: the better deployment choice depends on the specific model and task, not only on whether the system is implemented as a cascade or an end-to-end omni-modal model. Overall, this comparison provides an initial diagnostic signal for teams to evaluate candidate cascade and omni-modal systems on the same underlying tasks, tool schemas, and gold labels, before conducting broader deployment studies involving natural speech, latency, and cost constraints.

4.3.4 Error Analysis

Function-call error decomposition on Confetti: A practical advantage of our conversion design is that every audio input has a paired text counterpart with the same gold label. This allows us to isolate *paired failure cases*: instances where a model produces the correct tool call in text mode but fails on the corresponding audio input. We analyze these cases on *Confetti* using four mutually exclusive error categories: *i. decision errors*, where the model fails to make the correct high-level tool-use decision; *ii. tool-selection errors*, where it issues a tool call but selects the wrong API; *iii. argument-schema errors*, where it selects the correct tool but omits required arguments, as well as adds invalid arguments; and *iv. argument-value errors*, where it selects the correct tool and argument fields but fills one or more fields with incorrect, incomplete, or misheard content. We pool these paired failures across the six TTS configurations.

Table 5 shows that audio-induced failures are not explained by a single error type. For all models, the largest category is argument-value errors, which accounts for 57.2% failures for Gemini-3.1-Flash-Live and 54.3% for GPT-Realtime-1.5. These cases suggest that the models often preserve the broad tool-call structure but fail to recognize the exact argument content from speech. Qwen3-Omni shows a more distributed error profile: argument-value errors remain the largest category at 39.5%, but argument-schema errors (14.6%) and tool-selection errors (15.5%) are more frequent than other models. Decision errors are also substantial across all three systems, ranging from 25.8% for Gemini-3.1-Flash-Live to 37.4% for GPT-Realtime-1.5. This suggests that the text-to-audio gap is not only caused by incorrect argument values; in many cases, audio input changes the model's high-level tool-use behavior. Overall, the results indicate that improving speech recognition alone may not fully close the modality gap. Robust audio tool calling also requires preserving the model's downstream decisions about whether to call a tool, which tool to call, and how to call the tool.

Tool-call vs. no-tool-call errors on When2Call: In When2Call, the core challenge is not argument generation but deciding whether a tool call is needed at all. We therefore analyze the confusion matrices in Figure 3, which separate errors on *tool-call* and *no-tool-call* instances. The results show that most models are more reliable when a tool call is required, but struggle in no-tool-call cases. In particular, Qwen3-Omni

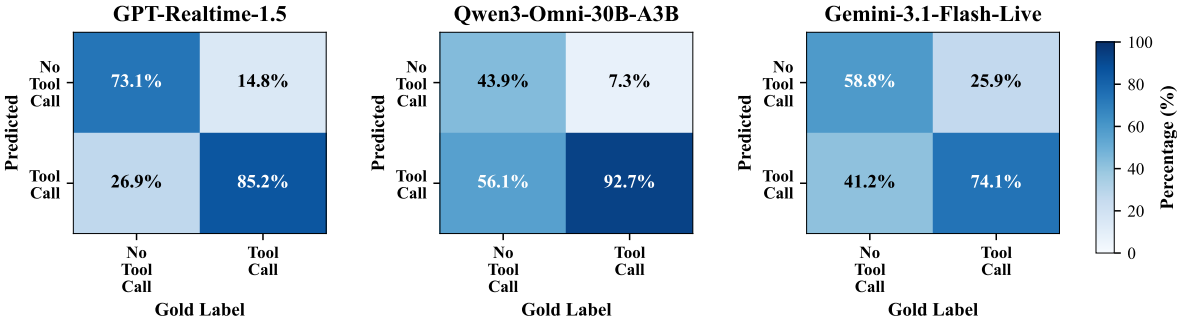


Figure 3: Error Analysis on the *When2Call* benchmark computed over 6 TTS voice variants.

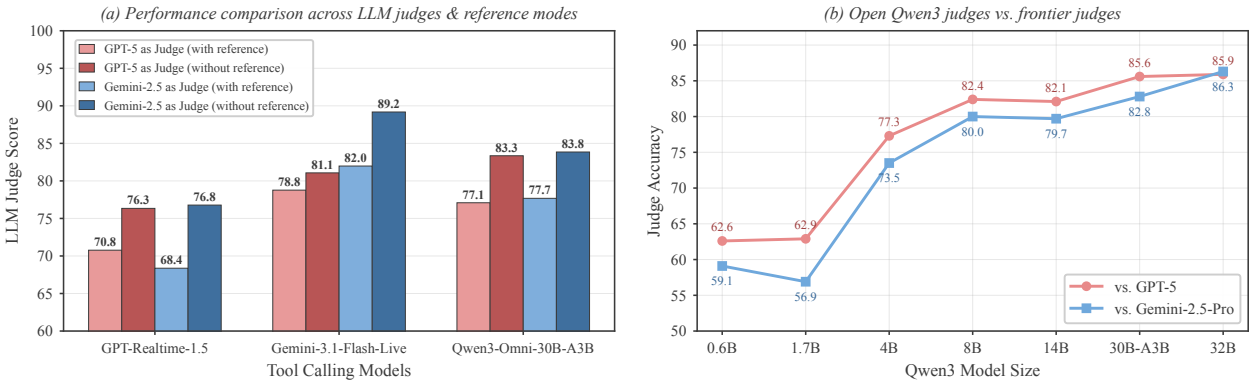


Figure 4: LLM-as-judge evaluation on *Confetti*. (a) Proprietary LLM judge scores in reference-wise and reference-free settings (see Appendix A.6 for the detailed breakdown). (b) Judgment agreement between open judges (Qwen3) against proprietary reference judgments.

correctly identifies tool-call cases in 92.7% of examples, but correctly identifies no-tool-call cases in only 43.9%. GPT-Realtime-1.5 is more balanced, with 85.2% accuracy on tool-call cases and 73.1% on no-tool-call cases. Gemini-3.1-Flash-Live shows a similar asymmetry, performing better on tool-call cases (74.1%) than no-tool-call (58.8%) cases. These results suggest that voice-agent failures are not limited to executing tools; models also need to know when *not* to call a tool.

4.3.5 Evaluation using LLM Judges

Because production deployments often lack complete gold annotations for customer-specific data, we evaluate whether LLM judges can provide a practical reference-free alternative for assessing tool-calling outputs. We focus on the *Confetti* dataset and use GPT-5 and Gemini-2.5-Pro as proprietary LLM judges. We evaluate each output in two settings: a *reference-aware* setting, where the judge sees the gold tool call, and a *reference-free* setting, where the judge needs to assess without seeing the gold tool call labels. This comparison tests whether reference-free judging preserves similar model-level conclusions to reference-aware judging (Appendix A.3 contains the sample prompt for the LLM judge).

Reference-aware vs. reference-free judging: Figure 4a shows that model rankings are broadly stable across judges and reference modes: Gemini-3.1-Flash-Live receives the highest scores overall, followed by Qwen3-Omni and GPT-Realtime-1.5. Reference-free scores are consistently higher than reference-aware scores for all three models, suggesting that judges may penalize minor deviations from the gold label when references are shown, even if the output is functionally reasonable (see Appendix A.4 for an example). To verify this statistically, we apply McNemar's test (McNemar, 1947) to paired judge decisions on the same model responses: the reference effect is **statistically significant** for both judges on GPT-Realtime-1.5 and Gemini-3.1-Flash-Live ($p < 0.001$), confirming that the higher reference-free scores reflect a systematic shift

in judge decisions rather than sampling noise. We further find that the two judges disagree significantly on Gemini-3.1-Flash-Live responses in both settings ($p < 0.001$). Moreover, we observe some judge-family specific preference bias: Gemini-2.5-Pro tends to assign higher scores to Gemini-3.1-Flash-Live. Overall, reference-free judging provides a useful production-oriented signal when gold labels are unavailable. However, results should be interpreted with multiple judges rather than a single one.

Human validation of LLM judges: To assess whether any specific proprietary judge should be preferred, we sample 200 cases where GPT-5 and Gemini-2.5-Pro disagree and conduct a pairwise preference study with two NLP experts. Gemini-2.5-Pro judgments are preferred in 52% of cases, while GPT-5 judgments are preferred in 48%. This near-even split suggests that neither judge is clearly preferred by humans.

Open-source judges for privacy-preserving evaluation: To address data privacy and cost concerns associated with proprietary judge APIs, we also benchmark Qwen3 models (Yang et al., 2025) of varying sizes as open-source evaluation judges on *Confetti*. Figure 4b shows that Qwen3 judges with at least 8B parameters achieve strong agreement (80%) with GPT-5 and Gemini-2.5-Pro, with the larger 32B variant exceeding 85% judgment agreement. This indicates that open judges can provide a viable privacy-preserving evaluation option for organizations that cannot send proprietary customer data to the closed judge APIs.

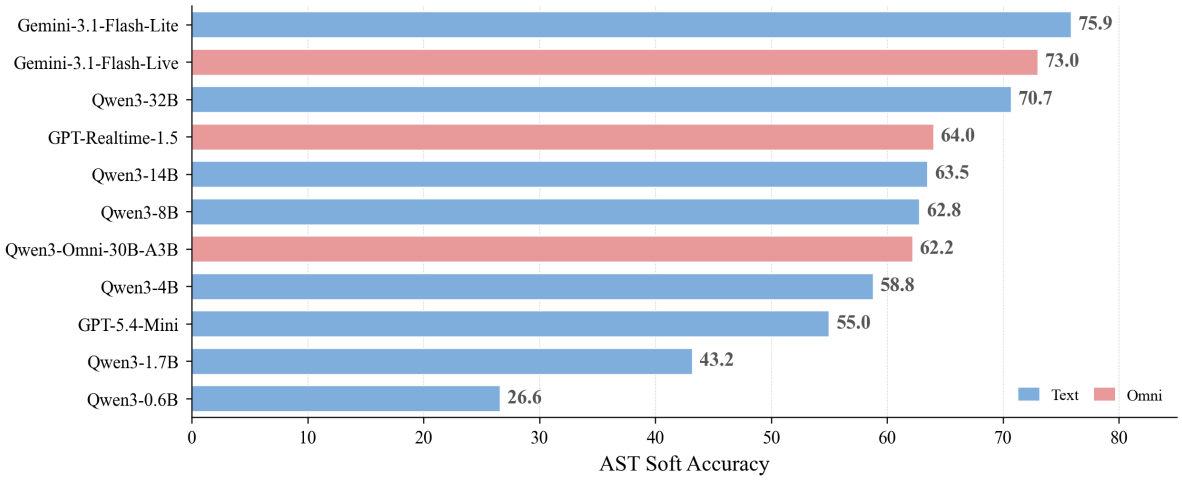
4.3.6 Text-Only Scaling Analysis

The cascade results (Table 4) suggest that text-based tool calling can remain competitive with direct voice models when the audio is first transcribed. This motivates a more deployment-oriented question: if a company uses an ASR→text-LLM cascade, how strong does the text-only LLM need to be? To answer this, we evaluate text-mode tool-calling performance on *Confetti* across various model families and sizes, including cost-effective Qwen3 models. Figure 5a shows that the accuracy generally increases with model size within the Qwen3 family, rising from 26.6% for Qwen3-0.6B to 70.7% for Qwen3-32B. Qwen3-Omni-30B-A3B obtains 62.2% in text mode, which is comparable to the Qwen3-8B (62.8%) and Qwen3-14B (63.5%), suggesting that many smaller text-only models may already provide competitive cascade backends. Among the non-Qwen models, Gemini-3.1-Flash-Lite achieves the highest score (75.9%), followed by Gemini-3.1-Flash-Live (73.0%) and GPT-Realtime-1.5 in text mode (64.0%). These results show that clean-text competence varies substantially across model families and sizes. For deployment, this means that audio degradation should be interpreted relative to each model's own text baseline, and that cascades with strong or cost-effective text-only LLMs may be a competitive alternative to direct omni-modal models.

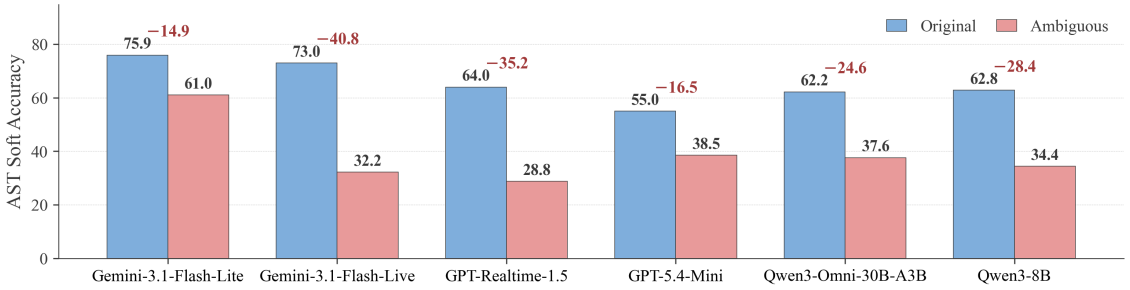
Robustness to query reformulation. Beyond the original datasets, we additionally define stress-test slices that practitioners can add when making deployment choices. These slices target customer-support failure modes that may be underrepresented in clean benchmarks and should be treated as a deployment checklist rather than a replacement for the main evaluation. Here, we test whether tool-calling accuracy is robust to ambiguous user phrasing. By introducing vagueness, we generate reformulations of the queries in *Confetti* using GPT-5 and Gemini-2.5-Pro in equal proportion (see Appendix A.5 for the sample prompt). We then evaluate the performance for the resulting queries in the text-only setting. Figure 5b shows that ambiguous queries cause large drops across all models. Qwen3-Omni falls from 62.2% to 37.6% (−24.6 points), while Gemini-3.1-Flash-Lite drops from 75.9% to 61.0% (−14.9 points). These results suggest that query formulation can be a major source of performance variance even before introducing audio.

4.3.7 Practical Implications for Deployment

Our results suggest that voice-agent architecture should be selected through model- and task-specific evaluation rather than by assuming that cascade or omni-modal systems are uniformly better. Direct omni-modal models are attractive when the text-to-voice gap is small, whereas cascades remain safer when teams need vendor flexibility, or when direct audio introduces large performance degradation. Clean-text performance provides the tool-calling ceiling, and the large drop under ambiguous reformulations shows that handling underspecified requests is as important as improving the audio stack. In practice, teams should use our framework as a first-stage benchmark and choose an omni-modal stack only when it improves accuracy, latency, and cost in the target environment; otherwise, a cascade architecture remains the safer choice.



(a) Text-only performance across model families and sizes.



(b) Ambiguous-query reformulation stress test.

Figure 5: Text-only tool-calling analysis on *Confetti*. (a) AST soft accuracy across model families and sizes. (b) AST soft accuracy under ambiguous-query reformulation stress tests.

5 Conclusion

We introduced a verifiability-preserving framework for converting text-based tool-calling benchmarks into controlled voice evaluations. By preserving tool schemas and gold labels, the framework enables reproducible first-stage comparisons to make cascade-vs-omni deployment decisions. Across two benchmarks, three TTS systems, multiple voices, and several LLMs, we find that performance is strongly model- and task-dependent: Gemini-3.1-Flash-Live leads on Confetti, while GPT-Realtime-1.5 leads on When2Call. On Confetti, the text-to-voice gap ranges from 1.8 points for Qwen3-Omni to 4.8 points for GPT-Realtime-1.5, showing that audio degradation is not a fixed property of omni-modal inference. Moreover, our analysis reveals that paired text–audio conversion provides diagnostic value beyond leaderboard scores. Text-only scaling further shows that cost-effective text LLMs can be competitive cascade backends, while ambiguity-based reformulation demonstrates that underspecified user requests can degrade performance even before audio is introduced. Finally, reference-free LLM-as-judge evaluation provides a practical option for production settings without gold labels, and open judges with sufficient scale show strong agreement with proprietary judges.

This study has limitations. We evaluate two datasets and a finite set of current models; broader domains and additional text benchmarks such as BFCL (Patil et al., 2025), τ-bench (Yao et al., 2025; Barres et al., 2025), as well as MCP benchmarks (Wang et al., 2025; Luo et al., 2025; Guo et al., 2026) remain important extensions. Our audio is TTS-generated, so it should be viewed as a controlled first-stage proxy rather than a replacement for spontaneous calls with natural conversational artifacts. Future work should extend the framework to naturally spoken customer-support data (Laskar et al., 2023b; Balaji et al., 2026), richer stress-test suites, alongside cost–latency evaluation.

References

Abdelrahman Abouelenin, Atabak Ashfaq, Adam Atkinson, Hany Awadalla, Nguyen Bach, Jianmin Bao, Alon Benhaim, Martin Cai, Vishrav Chaudhary, Congcong Chen, et al. Phi-4-mini technical report: Compact yet powerful multimodal language models via mixture-of-loras. *arXiv preprint arXiv:2503.01743*, 2025.

Tamer Alkhouli, Katerina Margatina, James Gung, Raphael Shu, Claudia Zaghi, Monica Sunkara, and Yi Zhang. CONFETTI: Conversational function-calling evaluation through turn-level interactions. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 7993–8006, Vienna, Austria, July 2025.

Rosana Ardila, Megan Branson, Kelly Davis, Michael Kohler, Josh Meyer, Michael Henretty, Reuben Morais, Lindsay Saunders, Francis Tyers, and Gregor Weber. Common voice: A massively-multilingual speech corpus. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 4218–4222, Marseille, France, May 2020.

Kaito Baba, Wataru Nakata, Yuki Saito, and Hiroshi Saruwatari. The T05 system for the VoiceMOS Challenge 2024: Transfer learning from deep image classifier to naturalness MOS prediction of high-quality synthetic speech. In *IEEE Spoken Language Technology Workshop (SLT)*, 2024.

Sumanth Balaji, Piyush Mishra, Aashraya Sachdeva, and Suraj Agrawal. Beyond ivr: Benchmarking customer support llm agents for business-adherence. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 5: Industry Track)*, pp. 193–208, 2026.

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.

Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T Tan, and Haizhou Li. Voicebench: Benchmarking llm-based voice assistants. *arXiv preprint arXiv:2410.17196*, 2024.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Ding Ding, Zeqian Ju, Yichong Leng, Songxiang Liu, Tong Liu, Zeyu Shang, Kai Shen, Wei Song, Xu Tan, Heyi Tang, et al. Kimi-audio technical report. *arXiv preprint arXiv:2504.18425*, 2025.

Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. Clotho: An audio captioning dataset. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 736–740, 2020.

Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 776–780, 2017.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Minlie Huang, Nan Duan, and Weizhu Chen. Tora: A tool-integrated reasoning agent for mathematical problem solving. *arXiv preprint arXiv:2309.17452*, 2023.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, et al. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*, 2024.

Zikang Guo, Benfeng Xu, Chiwei Zhu, Wentao Hong, Xiaorui Wang, and Zhendong Mao. Mcp-agentbench: Evaluating real-world language agent performance with mcp-mediated tools. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 30888–30896, 2026.

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

Dhruv Jain, Harshit Shukla, Gautam Rajeev, Ashish Kulkarni, Chandra Khatri, and Shubham Agarwal. Voiceagentbench: Are voice assistants ready for agentic tasks? *arXiv preprint arXiv:2510.07978*, 2025.

Shixin Jiang, Jiafeng Liang, Jiyuan Wang, Xuan Dong, Heng Chang, Weijiang Yu, Jinhua Du, Ming Liu, and Bing Qin. From specific-mllms to omni-mllms: a survey on mllms aligned with multi-modalities. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 8617–8652, 2025.

Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 119–132, 2019.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Md Tahmid Rahman Laskar, M Saiful Bari, Mizanur Rahman, Md Amran Hossen Bhuiyan, Shafiq Joty, and Jimmy Huang. A systematic study and comprehensive evaluation of ChatGPT on benchmark datasets. In *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 431–469, Toronto, Canada, July 2023a.

Md Tahmid Rahman Laskar, Cheng Chen, Xue-yong Fu, Mahsa Azizi, Shashi Bhushan, and Simon Corston-Oliver. Ai coach assist: An automated approach for call recommendation in contact centers for agent coaching. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pp. 599–607, 2023b.

Md Tahmid Rahman Laskar, Sawsan Alqahtani, M Saiful Bari, Mizanur Rahman, Mohammad Abdullah Matin Khan, Haidar Khan, Israt Jahan, Amran Bhuiyan, Chee Wei Tan, Md Rizwan Parvez, et al. A systematic survey and critical review on evaluating large language models: Challenges, limitations, and recommendations. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 13785–13816, 2024a.

Md Tahmid Rahman Laskar, Elena Khasanova, Xue-Yong Fu, Cheng Chen, and Shashi Bhushan Tn. Query-OPT: Optimizing inference of large language models via multi-query instructions in meeting summarization. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 1140–1151, 2024b. doi: 10.18653/v1/2024.emnlp-industry.86.

Md Tahmid Rahman Laskar, Julien Bouvier Tremblay, Xue-Yong Fu, Cheng Chen, and Shashi Bhushan Tn. Ai knowledge assist: An automated approach for the creation of knowledge bases for conversational ai agents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 1856–1866, 2025.

Minghao Li, Yingxiu Zhao, Bowen Yu, Feifan Song, Hangyu Li, Haiyang Yu, Zhoujun Li, Fei Huang, and Yongbin Li. API-bank: A comprehensive benchmark for tool-augmented LLMs. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 3102–3116, December 2023.

Yen-Ting Lin, Zhehuai Chen, Piotr Zelasko, Zhen Wan, Xuesong Yang, Zih-Ching Chen, Krishna C Puvvada, Ke Hu, Szu-Wei Fu, Jun Wei Chiu, Jagadeesh Balam, Boris Ginsburg, Yu-Chiang Frank Wang, and Chao-Han Huck Yang. NeKo: Cross-modality post-recognition error correction with tasks-guided mixture-of-experts language model. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pp. 222–236, July 2025.

Ziyang Luo, Zhiqi Shen, Wenzhuo Yang, Zirui Zhao, Prathyusha Jwalapuram, Amrita Saha, Doyen Sahoo, Silvio Savarese, Caiming Xiong, and Junnan Li. Mcp-universe: Benchmarking large language models with real-world model context protocol servers. *arXiv preprint arXiv:2508.14704*, 2025.

Quinn McNemar. Note on the sampling error of the difference between correlated proportions or percentages. *Psychometrika*, 12(2):153–157, 1947.

Grégoire Mialon, Roberto Dessì, Maria Lomeli, Christoforos Nalmpantis, Ram Pasunuru, Roberta Raileanu, Baptiste Rozière, Timo Schick, Jane Dwivedi-Yu, Asli Celikyilmaz, et al. Augmented language models: A survey. *arXiv preprint arXiv:2302.07842*, 2023.

OpenAI. Gpt-5 system card. `https://openai.com/index/gpt-5-system-card/`, 2025. Last Accessed: May 11, 2026.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: An ASR corpus based on public domain audio books. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210, 2015.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37:126544–126565, 2024.

Shishir G Patil, Huanzhi Mao, Fanjia Yan, Charlie Cheng-Jie Ji, Vishnu Suresh, Ion Stoica, and Joseph E Gonzalez. The berkeley function calling leaderboard (bfcl): From tool use to agentic evaluation of large language models. In *Forty-second International Conference on Machine Learning*, 2025.

Jing Peng, Yucheng Wang, Bohan Li, Yiwei Guo, Hankun Wang, Yangui Fang, Yu Xi, Haoyu Li, Xu Li, Ke Zhang, et al. A survey on speech large language models for understanding. *IEEE Journal of Selected Topics in Signal Processing*, 2025.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world APIs. *arXiv preprint arXiv:2307.16789*, 2023.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, et al. Tool learning with foundation models. *ACM Computing Surveys*, 57(4):1–40, 2024.

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. Tool learning with large language models: A survey. *Frontiers of Computer Science*, 19(8):198343, 2025.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pp. 28492–28518. PMLR, 2023.

Hayley Ross, Ameya Sunil Mahabaleshwarkar, and Yoshi Suhara. When2Call: When (not) to call tools. In *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)*, 2025.

Christine Rzepka, Benedikt Berger, and Thomas Hess. Voice assistant vs. chatbot–examining the fit between conversational agents' interaction modalities and information search tasks. *Information Systems Frontiers*, 24(3):839–856, 2022.

Salesforce AI Research and Berkeley. BFCL Audio: A benchmark for audio-native function calling. `https://www.salesforce.com/blog/bfcl-audio-benchmark/`, 2025. Last Accessed: May 11, 2026.

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.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Amit Sofer, Yoav Goldman, and Shlomo E Chazan. Pull it together: Reducing the modality gap in contrastive learning. In *Proc. Interspeech 2025*, pp. 196–200, 2025.

Xu Tan, Tao Qin, Frank Soong, and Tie-Yan Liu. A survey on neural speech synthesis. *arXiv preprint arXiv:2106.15561*, 2021.

Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, and Le Sun. Toolalpaca: Generalized tool learning for language models with 3000 simulated cases. *arXiv preprint arXiv:2306.05301*, 2023.

Joachim Thiemann, Nobutaka Ito, and Emmanuel Vincent. Demand: a collection of multi-channel recordings of acoustic noise in diverse environments. In *21st International Congress on Acoustics (ICA 2013)*, 2013.

Zhenting Wang, Qi Chang, Hemani Patel, Shashank Biju, Cheng-En Wu, Quan Liu, Aolin Ding, Alireza Rezazadeh, Ankit Shah, Yujia Bao, et al. Mcp-bench: Benchmarking tool-using llm agents with complex real-world tasks via mcp servers. *arXiv preprint arXiv:2508.20453*, 2025.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

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. Qwen2.5-omni technical report. *arXiv preprint arXiv:2503.20215*, 2025a.

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

Qiantong Xu, Fenglu Hong, Bo Li, Changran Hu, Zhengyu Chen, and Jian Zhang. On the tool manipulation capability of open-source large language models. *arXiv preprint arXiv:2305.16504*, 2023.

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

Chao-Han Huck Yang, Yile Gu, Yi-Chieh Liu, Shalini Ghosh, Ivan Bulyko, and Andreas Stolcke. Generative speech recognition error correction with large language models and task-activating prompting. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 1–8, 2023.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik R Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, et al. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*, 2024.

Aohan Zeng, Zhengxiao Du, Mingdao Liu, Kedong Wang, Shengmin Jiang, Lei Zhao, Yuxiao Dong, and Jie Tang. Glm-4-voice: Towards intelligent and human-like end-to-end spoken chatbot, 2024. URL https://arxiv.org/abs/2412.02612.

Yuheng Zha, Yichi Yang, Ruichen Li, and Zhiting Hu. AlignScore: Evaluating factual consistency with a unified alignment function. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11328–11348, July 2023.

Jian Zhang, Linhao Zhang, Bokai Lei, Chuhan Wu, Wei Jia, and Xiao Zhou. Wildspeech-bench: Benchmarking audio llms in natural speech conversation. *arXiv preprint arXiv:2506.21875*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging LLM-as-a-judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

A Appendix

A.1 Ethical Considerations

We maintain the licensing requirements accordingly while using different tools (OpenAI and Gemini models, HuggingFace, etc.). The human evaluation was conducted by co-authors, and so no additional compensation was required.

A.2 Sample Prompts for Omni-Modal LLMs

Our sample prompt in the Confetti dataset is provided below.

Prompt: Confetti

You are a helpful assistant.

You will be given a conversation context in text format and an audio input from the user.

You are also provided with a list of tools that you can leverage to answer the user query.

If a tool is appropriate, return a tool call with clear JSON arguments.

If not, answer in plain text.

Conversation Context: {Conversation Context}

List of Tools: {List of Tools}

Audio Input: {Audio Query}

Our sample prompt in the When2Call dataset is provided below.

Prompt: When2Call

You are a helpful assistant.

You will receive an audio input from the user and a list of available tools.

Your task is to respond by taking the most appropriate action from the following:

1. TOOL_CALL: If the user's request can be fulfilled by calling one of the available tools, call the tool with appropriate arguments.

2. FOLLOW_UP_QUESTION: If you need more information to fulfill the request or determine which tool to call, ask a clarifying follow-up question. Respond naturally based on the user's audio input.

3. CANNOT_ANSWER: If the user's request cannot be answered based on the information available, just inform the user that you do not know the answer.

4. DIRECT_ANSWER: If you know the answer to user's question, just answer it directly.

List of Tools: {List of Tools}

Audio Input: {Audio Query}

A.3 Sample Prompts for LLM Judge

LLM Judge Prompt (Without Reference): Confetti

You are an expert LLM judge and your goal is to evaluate whether an AI agent generated response in a customer-agent conversation is accurate to address the customer's concern. An AI Agent response can be accurate based on the following two criteria:

(i) It has accurately identified the right tool from the list of available tools to address customer's concern in a given conversation.
(ii) It has accurately identified that none of the tools are relevant to address customer's concern and so it did not make any tool calls.

You are first provided with the conversation, a list of available tools, and the AI agent generated response. You should response with 1 if the AI agent is correct and 0 if the AI agent is incorrect. Do not provide any additional information except '1' or '0' in your response.

The conversation, the list of available tools, and the AI agent generated response are given below.

Conversation: {Conversation}

Available Tools: {Tools}

AI Agent Response: {Ai_Agent_Response}

A.4 LLM Judge Evaluation Analysis

Example 1: Comparing Judgments in Reference-Wise vs Reference-Free Settings

- **Conversation:** The user asks for information about the capital of Jamaica.
- **Model Response:** "The capital of Jamaica is Kingston. It's a vibrant city known for its rich culture, music, and history. Here are some key highlights about Kingston:..." (followed by a detailed and accurate description of Kingston).
- **Gold Answer:**
  [{'RAG_query': {'query': ['capital of Jamaica']}}]
- **Reasoning:** In this case, the model provided a comprehensive and factually correct answer directly to the user's question. However, in the reference-wise setting, it was judged "`incorrect`" by both LLM judges because the expected action in the `gold_answer` was a `RAG_query` tool call to retrieve the information, rather than providing it directly. The model's response was correct and helpful, but it did not follow the prescribed tool-use path in the gold label. Nonetheless, in the reference-free setting, it was judged as "`correct`" by the judges. This case explains the reason behind models achieving higher accuracy in the reference-free setting (which we also find in some other instances in the dataset).

Example 2: Comparing Gemini-2.5-Pro Judgments with GPT-5

- **Conversation:** The user asks to find a hotel in Portland.
- **Model Response:** "Sure, I can help with that. Could you please provide the dates of your stay and any specific preferences you have, like star rating or amenities?"
- **Gold Answer:**
  [{'BookHotel_search_hotel': {'location': ['Portland']}}]

TTS Model	Voice	GPT-Realtime-1.5			Gemini-3.1-Flash-Live			Qwen3-Omni		
		GPT-5	Gemini-2.5-Pro	Avg.	GPT-5	Gemini-2.5-Pro	Avg.	GPT-5	Gemini-2.5-Pro	Avg.
With reference										
Gemini-2.5-Flash	Kore	71.43	69.10	70.27	77.08	79.73	78.41	78.74	78.07	78.41
Gemini-2.5-Flash	Orus	70.53	66.56	68.55	77.48	81.46	79.47	76.49	77.81	77.15
Gemini-2.5-Pro	Kore	71.29	67.99	69.64	78.22	82.51	80.37	77.89	78.22	78.06
Gemini-2.5-Pro	Orus	64.12	63.12	63.62	81.40	82.39	81.90	75.75	76.08	75.92
GPT-4o-Mini	Coral	73.80	71.25	72.53	78.91	83.71	81.31	77.00	77.64	77.32
GPT-4o-Mini	Ash	73.48	72.20	72.84	79.55	82.11	80.83	76.68	78.27	77.48
Average	–	**70.77**	**68.37**	**69.57**	**78.77**	**81.98**	**80.38**	**77.09**	**77.68**	**77.39**
Without reference										
Gemini-2.5-Flash	Kore	76.08	79.07	77.58	81.73	87.71	84.72	83.06	84.72	83.89
Gemini-2.5-Flash	Orus	76.49	74.17	75.33	79.80	86.09	82.95	85.43	82.45	83.94
Gemini-2.5-Pro	Kore	75.91	77.23	76.57	81.85	89.11	85.48	81.85	83.50	82.68
Gemini-2.5-Pro	Orus	71.43	71.43	71.43	81.06	91.03	86.05	82.06	83.72	82.89
GPT-4o-Mini	Coral	80.19	80.19	80.19	80.19	91.05	85.62	83.07	84.03	83.55
GPT-4o-Mini	Ash	77.96	78.59	78.28	81.79	90.10	85.95	84.66	84.66	84.66
Average	–	**76.34**	**76.78**	**76.56**	**81.07**	**89.18**	**85.13**	**83.35**	**83.85**	**83.60**

Table 6: LLM-judge results on Confetti across TTS configurations for GPT-Realtime-1.5, Gemini-3.1-Flash-Live, and Qwen3-Omni. GPT-5 and Gemini-2.5-Pro are used as judges under reference-aware and reference-free settings. The Avg. columns report the mean of GPT-5 and Gemini-2.5-Pro scores for each model. Higher values indicate better.

- **Reasoning:** Here, the user asked to find a hotel, and the model responded with a clarifying question to gather more details (dates, preferences) before searching. While this is a reasonable conversational step to refine the search, the `gold_answer` expected an immediate `BookHotel_search_hotel` tool call with the available information (`location=['Portland']`). The model's response was a sensible interaction, but it did not execute the specific tool call expected by the gold standard at that turn, leading to a "`incorrect`" judgment in the reference-wise setting. However, similar to the prior example, the Gemini-2.5-Pro considers the response as "`correct`" in the reference-free setting. Although in this example, the GPT-5 model considers this as "`incorrect`" in the reference-free setting. Our human annotators prefer Gemini-2.5-Pro judgment in such a case.

A.5 Sample Prompts for Ambiguous Query Generation

LLM Judge Prompt (Without Reference): Confetti

You will receive the conversation context and the original final query asked by a user. You need to rewrite the final user query in an ambiguous form while preserving the user's intent (use the conversation context for grounding).

For query reformulation, you can replace specific named entities, dates, numbers, and parameters with pronouns or generic descriptors (e.g., 'San Diego Airport' -> 'this airport', '2025-01-15' -> 'that day'). However, the query should still be answerable when read together with the conversation context.

[Conversation Context]

[Original Query]

A.6 Additional details in model performance

We provide a detailed breakdown of model performance based on LLM Judge evaluation in Table 6.