FROM SPEECH TO ACTION: UNCERTAINTY-AWARE REINFORCEMENT LEARNING FOR TOOL CALLING IN SPEECH LLMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Voice-driven tool calling extends LLM agents from typed interaction to spoken requests, enabling users to invoke external services through natural speech. However, unlike text-based tool calling, speech input introduces an upstream acoustic stage where noise, speaker variation, and ambiguity can distort the request before tool-use decisions are made. Existing uncertainty-aware tool-use methods mainly target text agents, leaving speech-channel uncertainty in tool routing and structured-call generation underexplored. To address this gap, we introduce a hierarchical diagnostic view that separates observable tool-name routing failures from conditional structured-call failures after correct routing, showing that these failure risks can be predicted from Speech LLM representations. This motivates UC-GRPO, an uncertainty-aware post-training and inference framework for end-to-end Speech LLM tool calling. UC-GRPO extracts routing and conditional decisional uncertainty with lightweight probes over Audio Encoder and Decoder states, uses the probe-derived signals to shape GRPO rewards, and implements selective execution through clarification routing. To enable systematic evaluation under controlled acoustic degradation, we further propose SpeechToolBench, a noise-augmented benchmark combining public tool-calling tasks with practical voice-agent scenarios and routing-decision diagnostics. UC-GRPO achieves 51.91% Acc Norm on SpeechToolBench, improving over the base Speech LLM by 11.46 points and standard GRPO by 5.09 points. Zero-shot evaluation on audio-adapted When2Call further shows that the learned uncertainty-aware behavior transfers to an independent tool-use benchmark.

1 INTRODUCTION

Large language model (LLM)-based agents (Huang et al., 2024; Zhao et al., 2024) extend language models by invoking external tools for information retrieval (Zhang et al., 2024), computation, and interaction with real-world services (Wang et al., 2025b). As speech interfaces become a natural access point for such agents, tool calling (Masterman et al., 2024) is no longer limited to typed instructions: users may expect a model to understand spoken requests and directly decide whether and how to call external APIs. End-to-end speech large language models (Speech LLMs) (Yu et al., 2026), exemplified by Qwen3-Omni (Xu et al., 2025b), make this interaction technically plausible by processing audio within a unified multimodal model. However, speech-driven tool calling creates a hierarchical reliability problem: the model must first map a noisy spoken request to an actionable tool intent and then instantiate a correct structured action. A noisy or ambiguous request can be routed incorrectly and then transformed into an executable but invalid tool call. Unlike ordinary textual errors, such failures may trigger external actions, corrupt downstream states, or expose private information, making uncertainty awareness essential before execution.

For text-based LLMs, uncertainty estimation (Van Amersfoort et al., 2020) has become a common way to identify unreliable generations before they are trusted or executed. Existing methods estimate uncertainty from output distributions, such as token entropy (Song et al., 2026b), perplexity (Huang et al., 2025), and semantic entropy (Kuhn et al., 2023; Nikitin et al., 2024), or from internal representations through hidden-state probing (Zhang et al., 2025a). In tool-use research, recent work (Zhou et al., 2026; Ye et al., 2026) studies how reinforcement learning affects tool-calling calibra-

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

tion and designs uncertainty-aware objectives. These studies motivate uncertainty-aware tool-use decisions.

Unlike text agents, Speech LLMs must infer the user's request from an acoustic signal before tool-use reasoning. Environmental noise, reverberation, and accent variation can affect the content of an utterance (Radford et al., 2023). Pipeline systems can attach confidence scores to ASR transcripts, but end-to-end Speech LLMs do not expose a clean boundary between transcription and decision making. Consequently, a model may confidently execute a tool call even when grounded in an unreliable spoken-input interpretation. The central question is therefore not only whether a tool-calling decision is uncertain, but where that uncertainty appears inside the speech model and how it affects action selection.

In this paper, we frame reliable speech tool calling as an uncertainty-aware decision problem for end-to-end Speech LLMs. Tool-call failure can arise from routing a request to the wrong tool or instantiating invalid arguments after correct routing. We therefore adopt a hierarchical diagnostic view that separates audio-side routing risk from conditional structured-call risk. The two risks are supervised by observable tool-call outcomes. We use uncertainty operationally as predicted task-failure risk, without assuming that tool-name mismatch identifies acoustic misperception. Inspired by this diagnostic view, we propose UC-GRPO, an uncertainty-aware post-training and inference framework. UC-GRPO trains lightweight probes over Audio Encoder and Decoder states, uses the resulting uncertainty to modulate GRPO rewards against risky execution, and routes high-risk executable calls to clarification at inference time.

To support systematic evaluation, we construct SpeechToolBench, a noise-augmented benchmark for speech tool calling. Beyond converting text queries into audio, it combines public tool-calling datasets with practical voice-agent scenarios, typed tool schemas, controlled acoustic perturbations, and routing-decision diagnostics. This enables controlled evaluation of how speech degradation changes tool routing, structured argument generation, and clarification behavior. We also evaluate zero-shot transfer on audio-adapted When2Call. Across these evaluations, UC-GRPO improves tool-calling performance over the base Speech LLM and standard GRPO, showing that internal uncertainty modeling is a viable mechanism for more reliable voice-driven tool use. Our main contributions are summarized as follows:

- We present a hierarchical diagnostic study of uncertainty in Speech LLM tool calling, separating upstream audio-side routing errors from structured-call errors conditioned on correct routing, and showing that these risks are reflected in different internal representations.
- We propose UC-GRPO, an uncertainty-aware GRPO framework that uses routing and conditional decisional probe signals for adaptive reward modulation during post-training and risk-aware execution-versus-clarification routing during inference.
- We construct SpeechToolBench, a noise-augmented benchmark for speech tool calling that includes 1,000 practical tool-use scenarios and supports routing-decision evaluation. UC-GRPO achieves an Acc Norm of 51.91%, improving over the base model by 11.46 points and over standard GRPO by 5.09 points.

2 Related Works

2.1 Uncertainty Quantification in LLMs

Uncertainty quantification (Smith, 2024) in LLMs asks whether a model can indicate when its output should not be trusted. Early methods rely on generation statistics such as entropy or perplexity, while semantic uncertainty methods estimate uncertainty over meaning-level alternatives (Shorinwa et al., 2025; Kossen et al., 2024; Nikitin et al., 2024). Later work estimates confidence in continuous semantic spaces, formulates uncertainty probabilistically, or uses conformal prediction to provide coverage guarantees (Qiu & Miikkulainen, 2024; Song et al., 2026a; Karimi et al., 2026; Vishwakarma et al., 2024). These output-level methods are useful after responses are produced or sampled, but can miss uncertainty encoded before decoding. Representation-based analyses probe hidden states for reliability signals, showing that activations can contain information related to hallucination or error, while also cautioning that such signals may reflect knowledge recall rather than

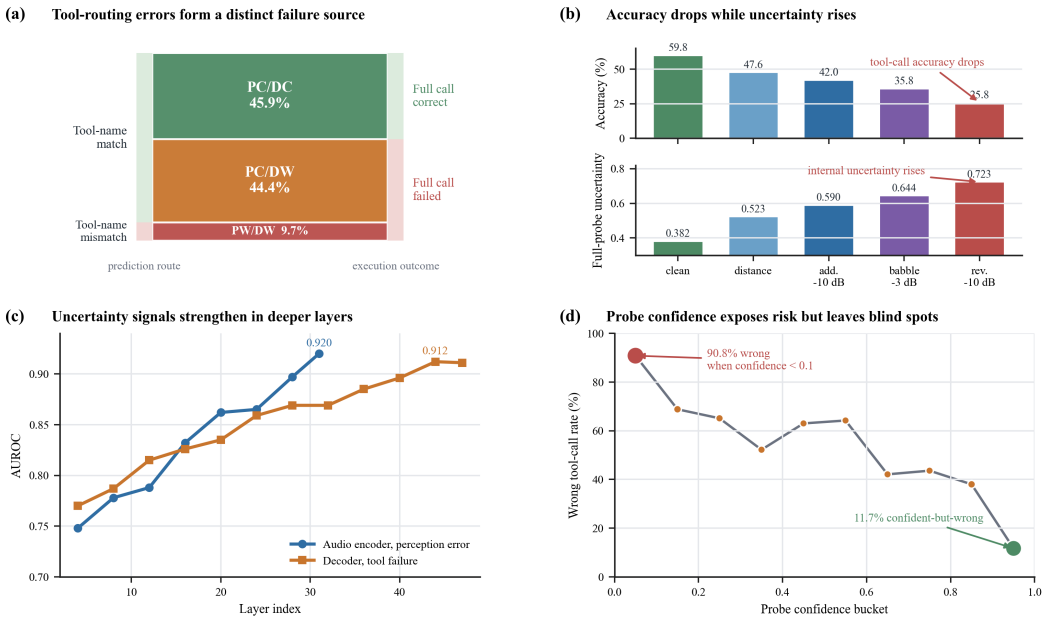


Figure 1: Uncertainty analysis for Speech LLM tool calling. PC/PW denote whether the predicted tool name matches the reference tool, and DC/DW denote whether the full tool call is correct. (a) shows the diagnostic flow from tool-name matching to full-call correctness. (b) compares accuracy and fused-probe uncertainty across acoustic conditions. (c) reports layer-wise AUROC for detecting tool-name mismatches from Audio Encoder states and conditional structured-call failures from Decoder states. (d) groups samples by probe confidence and reports wrong-call rates.

truthfulness and may not generalize uniformly (Orgad et al., 2025; Cheang et al., 2026). Our setting differs because uncertainty can enter before linguistic reasoning, when speech maps to tool intent.

2.2 Tool Calling in Large Language Model Agents

Tool calling (Yang et al., 2026b; Wang et al., 2026) extends LLM agents beyond text generation by enabling retrieval, computation, and interaction with external services. Toolformer (Schick et al., 2023), Gorilla (Patil et al., 2024), and ToolLLM (Qin et al., 2024) establish core techniques for learning when to call APIs, selecting tools, and generating arguments over large API collections. Evaluation has accordingly moved from isolated API calls toward decision-centered settings. BFCL (Patil et al., 2025) measures function-calling correctness across diverse functions and languages, When2Call (Ross et al., 2025) evaluates whether a model should call a tool, clarify, or answer directly, and recent environments test irrelevance, insufficient information, stateful execution, and long-horizon agent behavior (Lu et al., 2025; Yao et al., 2024; Trivedi et al., 2024). This line of work shows that tool use is not only a generation problem but also a reliability problem. TRUST (Zhou et al., 2026) further studies uncertainty-aware post-training for text-based tool agents. Our work addresses the additional failure mode introduced by speech input, where an erroneous tool action may arise before tool reasoning begins.

2.3 Speech Large Language Models

Voice-based LLM systems were traditionally built as ASR-LLM-TTS pipelines (Yang et al., 2026a), where speech is transcribed into text before an LLM performs reasoning or tool calling. This modular design supports speech recognition, spoken dialogue, and tool use, but it also compresses the acoustic signal into a transcript before the main model reasons over the request. End-to-end Speech LLMs and omni-modal systems reduce this boundary by integrating speech perception, language reasoning, and speech generation in a unified model. SpeechGPT (Zhang et al., 2023) and SALMONN (Tang et al., 2024) explore speech and audio instruction following, GPT-4o (Hurst

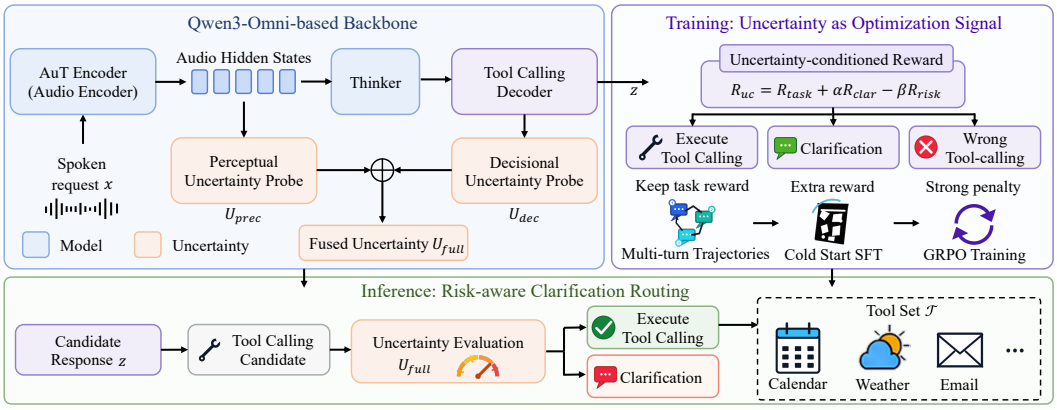


Figure 2: Overview of UC-GRPO. The Speech LLM processes a spoken request through the Audio Encoder and Decoder. Probes estimate spoken-input routing uncertainty and conditional decisional uncertainty from internal representations. UC-GRPO uses fused uncertainty for reward modulation during post-training and for clarification routing during inference.

et al., 2024) supports real-time multimodal interaction, and the Qwen-Omni series advances open omni-modal modeling from Qwen3-Omni (Xu et al., 2025b) to Qwen3.5-Omni (Team, 2026).

However, end-to-end modeling does not remove uncertainty from the acoustic channel. Recent audio-language benchmarks (Cheng et al., 2026; Gao et al., 2025) show that systems remain sensitive to speaker variation, environmental conditions, spoken instructions, and speech-based reasoning scenarios (Yang et al., 2024; Wang et al., 2025a). Unlike pipeline systems that expose transcripts and ASR confidence scores, end-to-end Speech LLMs blend perception, transcription, semantic reasoning, and action selection inside one model, making uncertainty harder to localize. Existing evaluations mainly measure audio understanding, conversation, or reasoning ability. They leave open how speech-channel uncertainty affects executable tool-use decisions, which is the focus of our analysis and method.

3 Uncertainty Analysis in Speech LLMs

Before introducing our training framework, we first examine where uncertainty arises in end-to-end Speech LLM tool calling. The typed structure of function calling provides a natural hierarchy of observable diagnostic variables. A valid tool call requires two ordered commitments. The model must first route the request to the intended function and only then instantiate the argument schema of that function. The predicted tool name is thus a necessary condition for task-level correctness and the earliest observable proxy for whether the spoken request has been mapped to the correct actionable intent. Once this routing decision is fixed, the remaining failures concern argument selection, argument formatting, or whether the available information is sufficient for execution.

Figure 1(a) checks whether these variables expose distinct failure regimes. The zero PW/DC cell follows from full-call correctness requiring the correct tool, rather than independently establishing a causal separation of perception and reasoning. The large PC/DW cell shows the opposite side of the problem. Even when the model selects the correct tool, many calls still fail because the arguments or execution conditions are wrong. We therefore use tool-name matching to track upstream routing uncertainty and analyze structured-call errors within the PC subset to study downstream decision risk. This is a diagnostic factorization of observable outcomes, not a claim that every tool-name mismatch is caused only by acoustic misperception.

Acoustic conditions further reveal why speech tool calling requires a separate uncertainty analysis. As shown in Figure 1(b), tool-call accuracy generally decreases as acoustic degradation becomes stronger, while fused probe uncertainty follows the opposite trend. Reverberant noise causes the sharpest collapse, with accuracy dropping to 25.8% at -10dB and uncertainty reaching its highest value. This pattern associates acoustic degradation with downstream risk. The joint trend in ac-

curacy and uncertainty motivates using internal risk estimates when deciding whether to execute a spoken request. Additional paired clean/noisy diagnostics are provided in Table 9.

We next ask whether this risk is visible only after generation or already encoded inside the model. Figure 1(c) shows that uncertainty signals become stronger in deeper representations. Audio Encoder states increasingly predict tool-name mismatches, reaching an AUROC of 0.920 in the deepest layer. Decoder states show a parallel trend for structured-call failures within correctly routed examples, reaching an AUROC of 0.912. These curves motivate complementary probes for routing and conditional-call prediction. They evaluate different diagnostic targets, so we also provide cross-representation comparisons for each target in Table 7.

Finally, Figure 1(d) examines how probe confidence relates to actual failure. Low-confidence samples are overwhelmingly unreliable, with a wrong-call rate of 90.8% when confidence is below 0.1. High-confidence samples are more reliable, but still contain a non-negligible confident-but-wrong region, with 11.7% wrong calls even when confidence exceeds 0.9. This pattern suggests two conclusions. Internal uncertainty is an informative signal for identifying high-risk speech tool calls, but it is not a complete solution by itself. Parameter-level errors can remain difficult to detect even when the model has selected the correct tool. These observations motivate the design of our framework, which combines internal uncertainty probing with training-time reward shaping and inference-time clarification routing.

4 METHOD

Our framework operationalizes the diagnostic structure in Section 3 for training and inference. Lightweight probes estimate routing and conditional decisional risk from hidden states. During UC-GRPO post-training, these scores modulate rewards to discourage uncertain erroneous execution and encourage annotated clarification. During inference, uncertainty-guided routing chooses between tool invocation and clarification. Thus, uncertainty serves as a decision signal throughout the framework.

4.1 TASK FORMULATION

Given a spoken request x and a tool set $\mathcal{T}$ with typed schemas, the Speech LLM policy π_θ generates an executable tool-call candidate, a clarification response, or a direct answer. An executable candidate is $z = (\hat{t}, \hat{\mathbf{a}})$ with $\hat{t} \in \mathcal{T}$, where $\hat{t}$ is the predicted tool name and $\hat{\mathbf{a}}$ denotes the predicted arguments. Executable examples provide a reference call $z^\star = (t^\star, \mathbf{a}^\star)$, while clarification and direct-answer examples provide annotated decision targets. For executable candidates, we define

$$C_{\text{name}}(z) = \mathbb{I}[\hat{t} = t^\star], \qquad C_{\text{call}}(z) = \mathbb{I}[s_{\text{AST}}(z, z^\star) \geq \delta], \tag{1}$$

where s_{AST} is the structured matching score between the generated and reference calls, and δ is the correctness threshold. Following the diagnostic analysis, we define routing and conditional structured-call failure labels:

$$y_{\text{route}} = 1 - C_{\text{name}}(z), \qquad y_{\text{dec}} = \mathbb{I}[C_{\text{name}}(z) = 1 \wedge C_{\text{call}}(z) = 0]. \tag{2}$$

These labels distinguish routing failure from structured-call failure after correct routing. Clarification and direct-answer rollouts have action-level targets but no tool-call correctness labels. They receive task rewards against their annotated decisions; at inference, probe-based risk determines whether an executable candidate should be deferred.

4.2 INTERNAL UNCERTAINTY PROBING

Output probabilities are available only after generation and can miss uncertainty that is already encoded in intermediate representations. We therefore train probes over hidden states. Let $H^{\text{aud}} = \{h_l^{\text{aud}}\}_{l=1}^{L}$ be Audio Encoder states and $H^{\text{dec}} = \{h_m^{\text{dec}}\}_{m=1}^{M}$ be Decoder states. For an Audio Encoder layer l, we mean-pool temporal states:

$$\bar{h}_l^{\text{aud}} = \frac{1}{T_l} \sum_{j=1}^{T_l} h_{l,j}^{\text{aud}}. \tag{3}$$

The audio-routing probe predicts tool-routing failure, while the decisional probe predicts conditional structured-call failure from the pre-execution action-decision state:

$$U_{\text{route}} = \sigma(w_r^\top \bar{h}_l^{\text{aud}} + b_r), \qquad U_{\text{dec}} = \sigma(w_d^\top h_m^{\text{dec}} + b_d). \tag{4}$$

Both probes are lightweight logistic classifiers trained with binary cross-entropy while the Speech LLM remains fixed. The routing probe uses all executable examples, whereas the decisional probe is trained and evaluated on the correctly routed subset so that it does not absorb routing failures. In UC-GRPO, probe parameters are trained before policy optimization and then kept fixed, but uncertainty scores are computed on the fly from pre-execution states. These scores are detached auxiliary reward signals, while tool correctness is still determined by external structured matching. We further learn a fused uncertainty score

$$U_{\text{full}} = \sigma(w_f^\top [U_{\text{route}}, U_{\text{dec}}] + b_f), \tag{5}$$

which is supervised by executable-call failure and used for both training and inference; generated clarification or direct-answer text is not used as probe supervision. This design preserves separate diagnostic targets while providing a single execution-risk score. Because policy updates change hidden representations, probe parameters remain fixed while their input states and resulting scores evolve with the policy. Post-training discrimination and calibration measurements are provided in Table 6.

4.3 UC-GRPO Training

Standard GRPO compares G rollouts $\{z_i\}_{i=1}^G$ sampled from the current policy for each spoken request. The task reward $R_{\text{task}}(z_i)$ evaluates format, tool selection, and arguments, but does not explicitly account for predicted execution risk.

UC-GRPO uses probe risk to reward appropriate clarification and penalize uncertain erroneous execution. Let $a_i \in \{\text{execute}, \text{clarify}, \text{direct}\}$ denote the action type of rollout z_i, let $U_i = U_{\text{full}}(z_i)$ be its internal uncertainty score, and let $C_{\text{clar}}(z_i)$ indicate that a clarification response matches the annotated decision target. We first define an uncertainty-aligned clarification reward

$$R_{\text{clar}}(z_i) = U_i \mathbb{I}[a_i = \text{clarify}] C_{\text{clar}}(z_i), \tag{6}$$

which assigns larger reward only to appropriate clarification when the model's internal state indicates higher risk. We also define an uncertainty-weighted execution penalty

$$R_{\text{risk}}(z_i) = U_i \mathbb{I}[a_i = \text{execute}] \mathbb{I}[C_{\text{call}}(z_i) = 0], \tag{7}$$

which increases when the model executes an incorrect tool call under high uncertainty. The final uncertainty-conditioned reward is

$$R_{\text{uc}}(z_i) = R_{\text{task}}(z_i) + \alpha R_{\text{clar}}(z_i) - \beta R_{\text{risk}}(z_i), \tag{8}$$

where α and β control the clarification bonus and erroneous-execution penalty. Only annotated clarification receives the bonus, and call-correctness terms apply only to executable rollouts.

We then use R_{uc} as the rollout reward in standard GRPO (Shao et al., 2024). After group-wise reward normalization, let $\hat{A}_i$ denote the normalized advantage of rollout z_i. We further weight each rollout loss by

$$W_i = 1 + \lambda U_i \mathbb{I}[a_i = \text{execute}] \mathbb{I}[C_{\text{call}}(z_i) = 0] \mathbb{I}[\hat{A}_i < 0], \tag{9}$$

so high-uncertainty erroneous executions increase the loss only when their group-relative advantage is negative. This avoids amplifying erroneous rollouts that receive a positive normalized advantage from other reward components. All other GRPO components are unchanged.

4.4 Inference-Time Clarification Routing

During inference, the model first generates a candidate response. If it is already a clarification or direct answer, it is returned directly. If it is executable, U_{full} is computed before tool execution. The final action is

$$\hat{a} = \begin{cases} \text{clarify}, & U_{\text{full}} > \tau, \\ \text{execute}, & U_{\text{full}} \le \tau, \end{cases} \tag{10}$$

where τ is selected on the validation split and kept fixed for test evaluation. This selective-execution rule converts predicted risk into clarification before an external service is invoked.

Table 1: Main results on SpeechToolBench in the audio-input setting.

Method	Training	Acc Norm↑	F1↑	Tool Hall.↓	AST↑	Param↑
Closed-source baselines						
Qwen3.5-Omni-Flash	–	45.13	61.64	55.85	0.615	0.529
Qwen3.5-Omni-Plus	–	43.61	60.73	48.72	0.657	0.598
Gemini-2.5-Flash	–	35.58	54.87	69.15	0.494	0.458
Gemini-2.5-Pro	–	41.98	58.52	61.09	0.608	0.533
GPT-4o-Mini	–	40.77	64.67	62.34	0.589	0.518
Open-source baselines						
Qwen2.5-Omni-7B	–	23.77	37.03	67.27	0.437	0.345
Qwen3-Omni-30B	–	40.45	57.61	55.82	0.611	0.511
Post-trained from Qwen3-Omni-30B						
GRPO	✓	46.82	63.78	52.86	0.643	0.551
UC-GRPO	✓	**51.91**	**66.50**	**47.45**	**0.803**	**0.660**

Table 2: Zero-shot transfer results on audio-adapted When2Call (Ross et al., 2025). Models are evaluated by direct inference without training, fine-tuning, or threshold tuning on When2Call. The noisy audio average is computed over the 5dB, 10dB, and 15dB conditions following the audio-conversion evaluation protocol.

		Noisy audio average			Clean audio		
Method	Training	Acc Norm↑	F1↑	Tool Hall.↓	Acc Norm↑	F1↑	Tool Hall.↓
Closed-source baselines							
Qwen3.5-Omni-Flash	–	70.36	64.85	39.58	72.00	66.40	33.50
Qwen3.5-Omni-Plus	–	**79.57**	**71.83**	**30.24**	**84.33**	**78.73**	**25.08**
Open-source baselines							
Qwen2.5-Omni-7B	–	40.04	49.95	85.44	44.20	53.46	82.11
Qwen3-Omni-30B	–	68.70	64.73	42.69	71.27	68.29	40.65
Post-trained from Qwen3-Omni-30B							
GRPO	✓	70.44	65.26	38.82	73.48	69.62	36.59
UC-GRPO	✓	**71.55**	**66.13**	**37.23**	**76.24**	**71.52**	**31.71**

5 EXPERIMENTS

5.1 SPEECHTOOLBENCH

Existing tool-calling benchmarks are primarily text-based, so they do not test whether a model preserves the intended action when the user request is spoken and acoustically degraded. To evaluate this ability, we propose SpeechToolBench, a benchmark for tool calling from spoken requests. SpeechToolBench combines public tasks and practical voice-agent requests with typed tool schemas, controlled acoustic degradation, and diagnostic labels. It evaluates both overall correctness and failures in tool routing or argument generation.

SpeechToolBench contains 9,090 schema-normalized tool-calling instances, split into 8,090 training, 500 validation, and 500 test instances before acoustic augmentation. It includes 8,090 instances from xLAM (Zhang et al., 2025b), ToolBench (Qin et al., 2024), and Hermes (Teknium et al., 2025), covering simple, multiple, and parallel tool-calling tasks with 2,765, 2,706, and 2,619 instances respectively. We further add 1,000 practical voice-agent scenarios to broaden everyday tool-use coverage; public and practical sources are stratified across all splits. For public text-derived instances, we synthesize speech using five TTS voices with varied gender and accent characteristics, while practical scenarios use human speech recordings. Each clean recording is paired with ten acoustic variants, yielding eleven evaluation conditions, including additive environmental noise, reverberant noise, multi-speaker babble, and distance fading. Appendix B describes the benchmark construction and the distribution of public and human-recorded requests across the source-stratified splits.

For evaluation, we use an independent test split of 500 queries under eleven acoustic conditions, resulting in 5,500 speech-input test examples. The main setting uses direct audio input, while text and ASR-transcript inputs are supported for controlled ablations. We report normalized task accuracy, tool hallucination rate, action-level F1, AST soft accuracy, parameter accuracy, execution rate, and clarification rate. SpeechToolBench also reports routing correctness and conditional decision correctness to separate tool selection from structured-call generation after correct routing. These labels are operational diagnostics rather than causal annotations of acoustic error. Full metric definitions and matching rules are provided in the supplementary material.

5.2 Experimental Details

5.2.1 Baselines

We compare UC-GRPO with closed-source and open-source speech-capable models. On SpeechToolBench, closed-source baselines include Qwen3.5-Omni-Flash, Qwen3.5-Omni-Plus (Team, 2026), Gemini-2.5-Flash, Gemini-2.5-Pro (Comanici et al., 2025), and GPT-4o-Mini (Hurst et al., 2024). Open-source baselines include Qwen2.5-Omni-7B (Xu et al., 2025a) and Qwen3-Omni-30B (Xu et al., 2025b). All baselines use the same tool schemas, reference decisions, and audio inputs, with model-specific wrappers only for tool-call interfaces. For training-based comparison, standard GRPO (Shao et al., 2024) is initialized from the same official Qwen3-Omni-30B checkpoint.

5.2.2 Training Setup

Our method uses Qwen3-Omni-30B as the backbone model. Both GRPO and UC-GRPO are trained on the SpeechToolBench training split with the same cold-start SFT and RL stages. We use full-parameter fine-tuning for post-training. UC-GRPO adds internal probes, uncertainty-conditioned rewards, uncertainty-weighted loss, and uncertainty-guided inference. Hyperparameters, prompts, training data construction, and probe settings are provided in the supplementary material.

5.2.3 Evaluation Setup

We evaluate on SpeechToolBench and on audio-adapted When2Call (Ross et al., 2025). SpeechToolBench uses 5,500 audio examples from 500 held-out queries across eleven acoustic conditions. For When2Call, we follow the audio-conversion evaluation protocol of Laskar et al. (2026), which preserves the original tool schemas and reference decisions while evaluating clean and noisy speech inputs. This setting is zero-shot: no model is trained or fine-tuned on When2Call. We report Acc Norm, F1, tool hallucination rate, AST score, and parameter accuracy where applicable. Metric definitions and implementation details are provided in the supplementary material.

5.3 Experimental Results

5.3.1 Results on SpeechToolBench

Table 1 shows the main results on SpeechToolBench. Standard GRPO improves Acc Norm over the Qwen3-Omni-30B backbone from 40.45% to 46.82%. UC-GRPO further raises Acc Norm to 51.91%, reduces tool hallucination to 47.45%, and improves structured-call quality, with AST increasing from 0.611 to 0.803 and parameter accuracy from 0.511 to 0.660. Relative to standard GRPO, UC-GRPO also improves AST from 0.643 to 0.803 and parameter accuracy from 0.551 to 0.660, while yielding the lowest tool hallucination rate. The AST and parameter gains indicate that UC-GRPO improves executable-call quality rather than only shifting execute-versus-clarify decisions. These results show that uncertainty-aware optimization improves both tool-use decisions and generated arguments under acoustic degradation.

5.3.2 Results on Audio When2Call

Table 2 evaluates transfer beyond SpeechToolBench. Among methods initialized from Qwen3-Omni-30B, UC-GRPO improves over standard GRPO on noisy and clean audio, raising Acc Norm from 70.44% to 71.55% on the noisy average and from 73.48% to 76.24% on clean audio. It also reduces tool hallucination from 38.82% to 37.23% under noise and from 36.59% to 31.71% on

Table 3: Ablation results on SpeechToolBench. All variants are evaluated in the same audio-input setting.

Variant	Acc Norm↑	F1↑	Tool Hall.↓
Base	40.45	57.61	55.82
Standard GRPO	46.82	63.78	52.86
Audio Probe Only	49.27	63.78	48.09
Decoder Probe Only	48.36	61.95	52.12
GRPO + Probe Routing	49.27	61.90	50.55
w/o Probe in Reward	47.48	61.41	52.19
w/o U-Weighted Loss	50.73	63.78	48.47
UC-GRPO	**51.91**	**66.50**	**47.45**

clean speech. The gains of 1.11 points under noise and 2.76 on clean speech support transfer of tool-use reliability across both acoustic settings. UC-GRPO outperforms Qwen3.5-Omni-Flash on noisy average Acc Norm and tool hallucination, suggesting that uncertainty-aware post-training transfers beyond SpeechToolBench to an independent zero-shot setting with different prompts and tool schemas.

5.4 Ablation Study

Table 3 isolates the main UC-GRPO components. The full method achieves the best accuracy, F1, and tool-hallucination rate among these variants.

The single-probe variants support complementary predictive signals. Using only the Audio Encoder probe improves Acc Norm to 49.27% and reduces tool hallucination to 48.09%, supporting the utility of encoder-derived routing risk for spoken tool use. The Decoder-only variant also improves over standard GRPO in Acc Norm, but its weaker tool-hallucination reduction suggests that conditional decisional uncertainty alone cannot fully control upstream spoken-input routing errors. Inference-time routing improves GRPO by 2.45 points (46.82 to 49.27); full UC-GRPO adds another 2.64 points (51.91). Both selective execution and uncertainty-aware training therefore contribute to the total gain. Removing probe uncertainty from the reward causes the largest drop, while removing the uncertainty-weighted loss still degrades all metrics. These results support the joint design of routing and conditional decisional probes, uncertainty-conditioned rewards, uncertainty-weighted optimization, and probe-guided inference. Additional analyses of clarification behavior and threshold sensitivity are provided in Tables 10 and 11.

The ablations also distinguish action-level performance at similar accuracy. Audio-only probing and inference-only routing both achieve 49.27% Acc Norm, but differ in F1 and tool hallucination. Full UC-GRPO improves F1 over inference-only routing from 61.90% to 66.50% and reduces tool hallucination from 50.55% to 47.45%. Removing probe-based rewards lowers Acc Norm to 47.48%, compared with 50.73% when removing uncertainty-based loss weighting, supporting both components in the full configuration. Supplementary training-stability and pre-execution error analyses appear in Tables 5 and 13.

6 Conclusion

This paper presented an uncertainty-aware framework for end-to-end Speech LLM tool calling. Our hierarchical diagnostic view separates audio-side routing errors from structured-call errors conditioned on correct routing. These risks appear in different internal representations, motivating UC-GRPO, which uses Audio Encoder and Decoder probes to shape GRPO rewards and guide selective execution through clarification routing. SpeechToolBench evaluates spoken tool calling under controlled acoustic degradation with routing-decision diagnostics. UC-GRPO improves normalized tool-calling performance over the base Speech LLM and standard GRPO on SpeechToolBench, while zero-shot evaluation on audio-adapted When2Call shows transfer to an independent benchmark, supporting internal uncertainty signals as a practical mechanism for more reliable spoken tool use under the evaluated conditions. Together, these results support incorporating internal risk estimates into spoken tool-use policies through complementary training and inference mechanisms.

REFERENCES

Chi Seng Cheang, Hou Pong Chan, Wenxuan Zhang, and Yang Deng. Do llms really know what they don't know? internal states mainly reflect knowledge recall rather than truthfulness. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 713–730, 2026.

Xize Cheng, Dongjie Fu, Chenyuhao Wen, Shannon Yu, Zehan Wang, Shengpeng Ji, Siddhant Arora, Tao Jin, Shinji Watanabe, and Zhou Zhao. Aha-bench: Benchmarking audio hallucinations in large audio-language models. *Advances in Neural Information Processing Systems*, 38, 2026.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, 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.

Kuofeng Gao, Shu-Tao Xia, Ke Xu, Philip Torr, and Jindong Gu. Benchmarking open-ended audio dialogue understanding for large audio-language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4763–4784, 2025.

Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. Understanding the planning of llm agents: A survey. *arXiv preprint arXiv:2402.02716*, 2024.

Yiming Huang, Junyan Zhang, Zihao Wang, Biquan Bie, Yunzhong Qiu, Xuming Hu, Yi R Fung, and Xinlei He. Reppl: Recalibrating perplexity by uncertainty in semantic propagation and language generation for explainable qa hallucination detection. *arXiv preprint arXiv:2505.15386*, 2025.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

Hamed Karimi, Vaishali Meyappan, and Reza Samavi. Llms uncertainty quantification via adaptive conformal semantic entropy. *arXiv preprint arXiv:2605.04295*, 2026.

Jannik Kossen, Jiatong Han, Muhammed Razzak, Lisa Schut, Shreshth Malik, and Yarin Gal. Semantic entropy probes: Robust and cheap hallucination detection in llms. *arXiv preprint arXiv:2406.15927*, 2024.

Lorenz Kuhn, Yarin Gal, and Sebastian Farquhar. Semantic uncertainty: Linguistic invariances for uncertainty estimation in natural language generation. *arXiv preprint arXiv:2302.09664*, 2023.

Md Tahmid Rahman Laskar, Xue-Yong Fu, Seyyed Saeed Sarfjoo, Quinten McNamara, Jonas Robertson, and Shashi Bhushan TN. From text to voice: A reproducible and verifiable framework for evaluating tool calling llm agents. *arXiv preprint arXiv:2605.15104*, 2026.

Jiarui Lu, Thomas Holleis, Yizhe Zhang, Bernhard Aumayer, Feng Nan, Haoping Bai, Shuang Ma, Shen Ma, Mengyu Li, Guoli Yin, et al. Toolsandbox: A stateful, conversational, interactive evaluation benchmark for llm tool use capabilities. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 1160–1183, 2025.

Tula Masterman, Sandi Besen, Mason Sawtell, and Alex Chao. The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey. *arXiv preprint arXiv:2404.11584*, 2024.

Alexander Nikitin, Jannik Kossen, Yarin Gal, and Pekka Marttinen. Kernel language entropy: Fine-grained uncertainty quantification for llms from semantic similarities. *Advances in Neural Information Processing Systems*, 37:8901–8929, 2024.

Hadas Orgad, Michael Toker, Zorik Gekhman, Roi Reichart, Idan Szpektor, Hadas Kotek, and Yonatan Belinkov. Llms know more than they show: On the intrinsic representation of llm hallucinations. In *International Conference on Learning Representations*, volume 2025, pp. 66880–66913, 2025.

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.

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. In *International Conference on Learning Representations*, volume 2024, pp. 9695–9717, 2024.

Xin Qiu and Risto Miikkulainen. Semantic density: Uncertainty quantification for large language models through confidence measurement in semantic space. *Advances in neural information processing systems*, 37:134507–134533, 2024.

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 et al. 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)*, pp. 3391–3409, 2025.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Ola Shorinwa, Zhiting Mei, Justin Lidard, Allen Z Ren, and Anirudha Majumdar. A survey on uncertainty quantification of large language models: Taxonomy, open research challenges, and future directions. *ACM Computing Surveys*, 58(3):1–38, 2025.

Ralph C Smith. *Uncertainty quantification: theory, implementation, and applications*. SIAM, 2024.

Haoyi Song, Ruihan Ji, Naichen Shi, Fan Lai, and Raed Al Kontar. Inv-entropy: A fully probabilistic framework for uncertainty quantification in language models. *Advances in Neural Information Processing Systems*, 38:107889–107922, 2026a.

Jouwon Song, Sohyeon Kim, and Kyeongbo Kong. Uncertainty-guided graph formulation via mwis for token pruning in lvlms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9510–9519, 2026b.

Changli Tang, Wenyi Yu, Guangzhi Sun, Xianzhao Chen, Tian Tan, Wei Li, Lu Lu, Zejun Ma, and Chao Zhang. Salmonn: Towards generic hearing abilities for large language models. In *International Conference on Learning Representations*, volume 2024, pp. 16607–16629, 2024.

Qwen Team. Qwen3. 5-omni technical report. *arXiv preprint arXiv:2604.15804*, 2026.

Ryan Teknium, Roger Jin, Jai Suphavadeeprasit, Dakota Mahan, Jeffrey Quesnelle, Joe Li, Chen Guang, Shannon Sands, and Karan Malhotra. Hermes 4 technical report. *arXiv preprint arXiv:2508.18255*, 2025.

Harsh Trivedi, Tushar Khot, Mareike Hartmann, Ruskin Manku, Vinty Dong, Edward Li, Shashank Gupta, Ashish Sabharwal, and Niranjan Balasubramanian. Appworld: A controllable world of apps and people for benchmarking interactive coding agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 16022–16076, 2024.

Joost Van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. In *International conference on machine learning*, pp. 9690–9700. PMLR, 2020.

Harit Vishwakarma, Alan Mishler, Thomas Cook, Niccolo Dalmasso, Natraj Raman, and Sumitra Ganesh. Prune'n predict: Optimizing llm decision-making with conformal prediction. *arXiv preprint arXiv:2501.00555*, 2024.

Bin Wang, Xunlong Zou, Geyu Lin, Shuo Sun, Zhuohan Liu, Wenyu Zhang, Zhengyuan Liu, AiTi Aw, and Nancy Chen. Audiobench: A universal benchmark for audio large language models. 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)*, pp. 4297–4316, 2025a.

Haoxin Wang, Xianhan Peng, Huang Cheng, Yizhe Huang, Ming Gong, Chenghan Yang, Yang Liu, and Jiang Lin. Ecom-bench: Can llm agent resolve real-world e-commerce customer support issues? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 276–284, 2025b.

Ziyi Wang, Yuxuan Lu, Yimeng Zhang, Pei Chen, Ziwei Dong, Jing Huang, Jiri Gesi, Xianfeng Tang, Chen Luo, Qun Liu, et al. Trajectory2task: Training robust tool-calling agents with synthesized yet verifiable data for complex user intents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44021–44044, 2026.

Jin Xu, Zhifang Guo, Jinzheng He, et al. Qwen2.5-omni technical report. *arXiv preprint arXiv:2503.20215*, 2025a. doi: 10.48550/arXiv.2503.20215.

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

Jianing Yang, Yusuke Fujita, and Yui Sudo. Duplexcascade: Full-duplex speech-to-speech dialogue with vad-free cascaded asr-llm-tts pipeline and micro-turn optimization. *arXiv preprint arXiv:2603.09180*, 2026a.

Qian Yang, Jin Xu, Wenrui Liu, Yunfei Chu, Ziyue Jiang, Xiaohuan Zhou, Yichong Leng, Yuanjun Lv, Zhou Zhao, Chang Zhou, et al. Air-bench: Benchmarking large audio-language models via generative comprehension. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1979–1998, 2024.

Zuhao Yang, Sudong Wang, Kaichen Zhang, Keming Wu, Sicong Leng, Yifan Zhang, Bo Li, Chengwei Qin, Shijian Lu, Xingxuan Li, et al. Longvt: Incentivizing" thinking with long videos" via native tool calling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 33816–33826, 2026b.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. tau-bench: A benchmark for tool-agent-user interaction in real-world domains. *arXiv preprint arXiv:2406.12045*, 2024.

Zihuiwen Ye, Lukas Aichberger, Michael Kirchhof, Sinead Williamson, Luca Zappella, Yarin Gal, Arno Blaas, and Adam Golinski. Uncertainty quantification for llm function-calling. *arXiv preprint arXiv:2604.22985*, 2026.

Wenyi Yu, Siyin Wang, Xiaoyu Yang, Xianzhao Chen, Xiaohai Tian, Jun Zhang, Guangzhi Sun, Lu Lu, Yuxuan Wang, and Chao Zhang. Salmonn-omni: A standalone speech llm without codec injection for full-duplex conversation. *Advances in Neural Information Processing Systems*, 38: 24617–24643, 2026.

Anqi Zhang, Yulin Chen, Jane Pan, Chen Zhao, Aurojit Panda, Jinyang Li, and He He. Reasoning models know when they're right: Probing hidden states for self-verification. *arXiv preprint arXiv:2504.05419*, 2025a.

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

Jianguo Zhang, Tian Lan, Ming Zhu, Zuxin Liu, Thai Quoc Hoang, Shirley Kokane, Weiran Yao, Juntao Tan, Akshara Prabhakar, Haolin Chen, et al. xlam: A family of large action models to empower ai agent systems. 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)*, pp. 11583–11597, 2025b.

Weinan Zhang, Junwei Liao, Ning Li, Kounianhua Du, and Jianghao Lin. Agentic information retrieval. *arXiv preprint arXiv:2410.09713*, 2024.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: Llm agents are experiential learners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 19632–19642, 2024.

Yijin Zhou, Linqian Zeng, Xiaoya Lu, Wenyuan Xie, Dongrui Liu, Junchi Yan, and Jing Shao. Exploring agentic tool-calling decisions via uncertainty-aligned reinforcement learning. *arXiv preprint arXiv:2606.06976*, 2026.

A EXPERIMENTAL DETAILS

This section provides implementation details for the training and evaluation setup used in the main paper. All post-training experiments use Qwen3-Omni-30B-A3B-Instruct as the backbone model. We perform full fine-tuning. Unless otherwise specified, training is conducted on two NVIDIA B200 GPUs.

The training objective is UC-GRPO. The learning rate is 3×10^{-6}, the group size is 8, and the query batch size is 64, giving 512 sampled rollouts per batch. The model is trained for 3000 total steps, with 100 SFT warm-up steps and 100 learning-rate warm-up steps. Rollout sampling uses temperature 2.0, top-p 0.95, top-k 50, and a maximum generation length of 128 tokens. The UC-GRPO coefficients follow the notation in the main paper: $\alpha = 4.0$, $\beta = 2.5$, and $\lambda = 1.0$. Incorrect or unnecessary clarification is handled by the task reward, so no additional γ term is used outside the reported objective. The probe routing threshold used during inference is 0.5.

The routing and conditional decisional probes are trained before policy optimization and kept fixed during UC-GRPO training. The routing probe uses mean-pooled states from the deepest Audio Encoder layer, and the conditional decisional probe uses the pre-execution action-decision state from the final Decoder layer. Both probes are logistic heads trained with class-balanced binary cross-entropy on the training split, with layer and threshold selection performed on the validation split. The fused probe is a logistic head over $[U_{\text{route}}, U_{\text{dec}}]$ trained to predict executable-call failure. During rollout generation and inference, probe scores are computed online from the current hidden states of each example. The probe scores are used as detached auxiliary signals for reward shaping and inference-time routing; tool-call correctness is always determined by the external structured matching rules described in Section C.

All GRPO implementation details not specific to uncertainty modeling, including optimizer, clipping, KL control, precision, and distributed training configuration, are kept identical between the standard GRPO baseline and UC-GRPO. The only method-specific changes are the probe-derived reward terms, uncertainty-weighted loss factor, and inference-time routing rule.

B SPEECHTOOLBENCH

SpeechToolBench is designed to evaluate tool calling from spoken user requests under controlled acoustic degradation. The benchmark contains 9,090 tool-calling instances, split by original request into 8,090 training, 500 validation, and 500 test instances before acoustic augmentation. After expanding each instance to 11 acoustic conditions, the split contains 88,990 training, 5,500 validation, and 5,500 test speech-input examples. Among the canonical instances, 8,090 are collected from xLAM (Zhang et al., 2025b), ToolBench (Qin et al., 2024), and Hermes (Teknium et al., 2025), covering simple, multiple, and parallel tool-calling tasks. We further add 1,000 practical voice-agent scenarios to improve coverage of everyday tool-use requests beyond curated text benchmarks. The public and practical sources are stratified across splits, so the 8,090/500/500 split should not be interpreted as a source split.

Table 4: Source distribution before acoustic augmentation. Both public and human-recorded scenarios are distributed across train, validation, and test splits.

Source	Train	Val.	Test	Total
Public text-derived	7200	445	445	8090
Human-recorded practical	890	55	55	1000
Total	8090	500	500	9090

Each instance is normalized into a unified schema containing the user query, available tool definitions, reference tool calls, task category, and source dataset. For public text-derived instances, we convert requests into spoken-style utterances and synthesize clean speech with five TTS voices covering different gender and accent characteristics. In contrast, the 1,000 practical voice-agent scenarios are collected as human speech recordings rather than TTS audio. To evaluate robustness to the speech channel, each clean recording is paired with ten controlled perturbations, yielding 11

acoustic conditions in total. These conditions include clean speech, additive environmental noise, reverberation, multi-speaker babble, distance fading, and other noise sources. The same tool schema and reference decision are preserved across acoustic variants so that performance changes can be attributed to the speech input condition.

C Metric Definitions

We report the same metrics as in the main paper: normalized task accuracy, F1, tool hallucination rate, AST soft accuracy, and parameter accuracy. The action space contains three response types: executable tool calls, clarification requests, and direct answers.

Acc Norm measures task success after normalizing heterogeneous responses into a common action space. It is a micro task-accuracy metric rather than class-balanced accuracy:

$$\mathrm{AccNorm} = \frac{1}{N} \sum_{a \in \mathcal{A}} N_{\mathrm{correct}}(a), \tag{11}$$

where $\mathcal{A} = \{\mathrm{execute}, \mathrm{clarify}, \mathrm{direct}\}$ and N is the total number of evaluated examples. The term "Norm" refers to response normalization across action types, not per-class reweighting. A correct execution denotes an executable response whose tool name and arguments match the reference call. A correct clarification denotes a clarification response that matches the annotated decision target for insufficient or ambiguous requests. A correct direct answer denotes a response that matches a reference no-tool decision and satisfies the expected answer type without invoking a tool.

F1 measures macro-F1 over the three action labels after parsing the response as execute, clarify, or direct answer; structured correctness is reported separately by AST and parameter accuracy. Tool hallucination rate is computed over all evaluated examples and measures unsafe or unnecessary tool behavior, including unsupported tool invocation, executable calls when the reference action is clarify or direct answer, wrong tool selection, and executable calls whose arguments contradict the reference decision.

AST soft accuracy is the average normalized tree-similarity score s_{AST} on examples whose reference action is execute. If the model does not produce an executable tool call for such an example, its AST score is set to zero. Clarification and direct-answer reference examples are excluded because they do not have reference tool-call JSON objects. For binary executable-call correctness in diagnostic labels, we use

$$C_{\mathrm{call}}(z) = \mathbb{I}[s_{\mathrm{AST}}(z, z^\star) \geq \delta], \tag{12}$$

where z is the generated call, $z^\star$ is the reference call, and $\delta = 1.0$ is used for exact executable-call correctness unless otherwise specified. Parameter accuracy uses the same reference-execute denominator and is computed after schema normalization as the fraction of required reference parameters whose normalized values match the generated values; non-executable responses to reference-execute examples receive zero parameter accuracy. For multiple or parallel calls, tool calls are first matched by tool name and then by maximum bipartite AST similarity before parameter scores are averaged.

For routing-decision diagnostics, we additionally use tool-name correctness:

$$C_{\mathrm{name}}(z) = \mathbb{I}[\hat{t} = t^\star], \tag{13}$$

where $\hat{t}$ is the predicted tool name and $t^\star$ is the reference tool name. Conditional decision correctness is evaluated only on correctly routed examples and measures whether the complete structured call is correct after the intended tool has been selected.

D Additional Validation Experiments

This section provides additional validation experiments that support the main claims. We focus on questions that are most relevant to the reliability of UC-GRPO: whether fixed probes remain meaningful after full-parameter RL, whether different hidden states encode different risks, whether clarification behavior is driven by higher uncertainty, and whether UC-GRPO reduces errors across uncertainty regions.

D.1 Multi-Seed Stability

Table 5 reports mean and standard deviation across independently trained checkpoints. UC-GRPO remains consistently above standard GRPO across seeds, indicating that the main improvement is not due to a single favorable training run.

Table 5: Multi-seed results on SpeechToolBench. We report mean and standard deviation across independently trained checkpoints.

Method	Seeds	Acc Norm↑	F1↑	Tool Hall.↓
Base	–	40.45	57.61	55.82
GRPO	3	46.82 ± 0.62	63.78 ± 0.47	52.86 ± 0.53
UC-GRPO	3	$\mathbf{51.91 \pm 0.45}$	$\mathbf{66.50 \pm 0.49}$	$\mathbf{47.45 \pm 0.37}$

D.2 Uncertainty Quality after Post-Training

UC-GRPO trains routing and conditional decisional probes before policy optimization and keeps only the probe parameters fixed during full-parameter RL. The uncertainty scores themselves are not fixed or cached. During rollout generation and inference, each score is recomputed online from the current hidden states of the updated Speech LLM. We evaluate whether the fused risk score remains informative after post-training by comparing uncertainty quality across UC-GRPO and ablated variants. Table 6 reports calibration and ranking metrics for predicting executable-call failure. UC-GRPO achieves the best ECE and AUPRC and substantially improves AUROC over variants that remove probe-based reward shaping, supporting a useful fused risk signal after policy optimization. Calibration remains imperfect, and the full method does not achieve the best Brier score or AUROC in this table.

Table 6: Uncertainty quality of the fused risk score after post-training. Rows are UC-GRPO ablations; GRPO+Routing applies probe routing only at inference.

Method	ECE↓	Brier↓	AUROC↑	AUPRC↑
Full	**0.2247**	0.2677	0.8137	**0.8195**
GRPO+Routing	0.3043	**0.2603**	0.7756	0.7647
w/o Probe Reward	0.2760	0.2735	0.7608	0.7737
w/o U-Loss	0.3005	0.2630	0.8142	0.7736

D.3 Hierarchical Representation Evidence

We further evaluate whether Audio Encoder and Decoder states are associated with different failure types. Table 7 reports cross-representation diagnostic AUROC complementing Figure 1(c) in the main paper. Audio Encoder states are most predictive of tool-name routing failures, while Decoder states are most predictive of conditional structured-call failures. These matched diagnostic probes should not be confused with the fused any-failure score in Table 6, which solves a harder aggregate prediction problem by combining heterogeneous failure modes for inference-time routing.

Table 7: Cross-representation diagnostic AUROC for the two failure types illustrated in Figure 1(c) of the main paper. The gap reports the matched-source advantage, i.e., Audio over Decoder for routing failure and Decoder over Audio for conditional structured-call failure.

Diagnostic target	Audio	Decoder	Gap
Tool-name routing failure	**0.920**	0.706	+0.214
Conditional structured-call failure	0.658	**0.912**	+0.254

The Decoder probe is interpreted as conditional decision risk only on correctly routed executable examples. For wrong-tool candidates, UC-GRPO does not interpret the Decoder probe alone; the fused score combines it with the routing probe and is used only as an aggregate execution-risk signal.

System Prompt for Speech Tool Calling

```
You are a helpful voice assistant that can call tools. Listen to the user's request and decide what to do.

Available tools: {tools_json}

You have three options:
1. Call a tool: respond with ONLY: <<{"name": "tool_name", "arguments": {"arg1": "value1"}}>>
2. Ask for more information: <action>clarify</action> followed by your question
3. Answer directly if no tool is needed.

Important: only call a tool if the spoken request clearly requires it.
If the request is ambiguous, clarify. If no tool is needed, answer directly.
```

Figure 3: System prompt used for speech tool calling. The model receives the spoken request and available tool definitions, then chooses among tool execution, clarification, and direct answering.

D.4 Comparison with Output-Level Uncertainty Signals

We compare the internal probes with output-level and external uncertainty signals on the same diagnostic targets. Token entropy is computed from the generated token distribution, self-consistency uses repeated decoding agreement, and ASR confidence is computed from an external speech recognizer. Table 8 shows that output-level signals are informative but less aligned with the two-stage structure of speech tool calling. ASR confidence is relatively strong for routing failure, but remains below the Audio Encoder probe and is weaker for conditional structured-call failure. Token entropy is stronger for conditional failure than routing failure, but remains below the Decoder probe. The fused probe provides a strong unified pre-execution signal, while the matched Audio and Decoder probes give the clearest evidence of stage-specific uncertainty.

Table 8: AUROC comparison between internal probes and output-level or external uncertainty signals. Internal probes are strongest on their matched diagnostic targets, while the fused probe provides a unified pre-execution risk score.

Signal	Routing fail	Cond. call fail
Token entropy	0.674	0.759
Self-consistency	0.606	0.616
ASR confidence	0.773	0.634
Audio probe	**0.920**	0.658
Decoder probe	0.706	**0.912**
Fused probe	0.887	0.858

D.5 Sensitivity to Acoustic-Induced Errors

We further test whether the fused probe is sensitive to errors introduced by acoustic degradation rather than only to generic tool-use difficulty. We compare each query under clean speech and noisy speech, and define noise-induced errors as cases that are correct under clean speech but fail under noisy speech. Persistent errors are incorrect under both clean and noisy conditions. Table 9 shows that the fused probe assigns higher risk to noise-induced errors and separates them from persistent errors, whereas token entropy does not separate these error groups in the same direction. These group means are not within-request clean-to-noisy score changes, and the fused-score result does not isolate perceptual uncertainty in the Audio Encoder alone.

Table 9: Distinguishing noise-induced errors from persistent errors. The fused probe is more sensitive to acoustic-induced failures than token entropy.

Signal	Noise-induced	Persistent	AUROC	p-value
Fused probe	**0.387**	0.342	**0.702**	$< 10^{-6}$
Token entropy	1.056	1.141	0.432	0.238

Evaluation and Routing Judger Prompt

```
You are a structured evaluator for tool-calling responses in voice assistant scenarios.

Given: voice query metadata, tool specifications, reference decision, and model response.

Task:
1. Parse the action as execute, clarify, or direct_answer.
2. If execute, match the tool name and normalized arguments; compute AST and Param.
3. If clarify/direct_answer, compare against the reference decision.
4. If probe_uncertainty > threshold, route execute to clarify before execution.

Return strict JSON:
{
  "action": "execute" | "clarify" | "direct_answer",
  "is_correct_exec": true | false,
  "routed_to_clarify": true | false,
  "probe_uncertainty": float,
  "ast_score": float,
  "param_accuracy": float,
  "error_type": "wrong_tool" | "wrong_args" | "hallucinated_params" | null
}
```

Figure 4: Evaluation and routing prompt used to parse model responses, compute structured-call metrics, and apply uncertainty-guided routing before external tool execution.

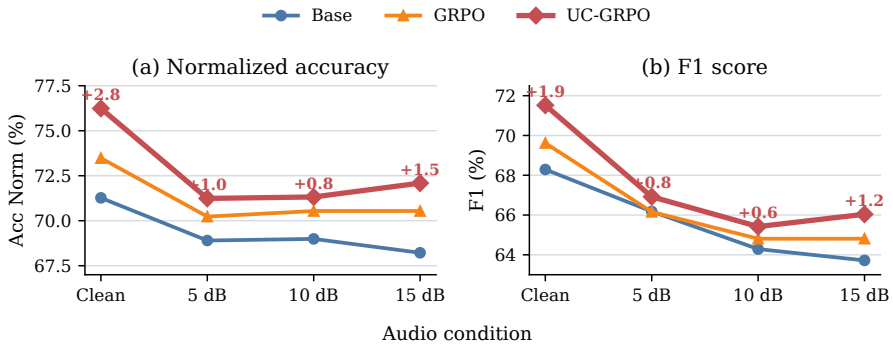


Figure 5: Audio-adapted When2Call trends for the Qwen3-Omni-30B backbone and its post-trained variants. Red annotations show the absolute gain of UC-GRPO over standard GRPO.

D.6 Clarification Routing Analysis

We analyze pre-execution uncertainty scores grouped by the final action. The score is read from the action-decision hidden state and does not use generated clarification text as probe input. Table 10 shows that examples routed to clarification have substantially higher uncertainty than executable responses. Within executable responses, wrong executions also have higher uncertainty than correct executions. These trends indicate that the fused probe captures input and decision risk before external tool execution rather than merely recognizing the surface form of a clarification response.

Table 10: Pre-execution uncertainty distribution for UC-GRPO execution and clarification actions. Clarification actions have higher uncertainty while remaining rare.

Action group	N	Mean uncertainty
Execute (all)	5350	0.329
Correct execution	2738	0.320
Wrong execution	2612	0.338
Clarify (all)	**150**	**0.524**
Correct clarify	117	0.542
Wrong clarify	33	0.466

The UC-GRPO clarification rate in this diagnostic set is 2.73%, and clarification precision is 78.00%. This suggests that the improvement is not driven by excessive abstention, but by selectively deferring a small number of high-uncertainty examples.

D.7 Threshold Sensitivity

The clarification threshold τ is selected on the validation split under a low-clarification budget and then fixed for test evaluation. Table 11 reports sensitivity around the selected value. Lower thresholds defer more examples, while $\tau = 0.5$ gives the best Acc Norm among the tested thresholds and keeps clarification below 3%.

Table 11: Threshold sensitivity around the validation-selected routing threshold. All values are percentages.

τ	Exec. cov.	Clar. rate	Acc Norm
0.45	94.73	5.27	50.82
0.50	97.27	2.73	51.91
0.55	99.09	0.91	50.55

D.8 Pre-Execution Signal Complementarity

We compare fused probe uncertainty with lightweight pre-execution behavior signals. Table 12 reports aggregate execution-risk AUROC for signal combinations rather than emphasizing single-signal ranking. Text-level behavior features already capture part of the risk, but adding the fused probe improves AUROC from 0.7410 to 0.7925 without requiring an external ASR model. Adding ASR confidence gives only a marginal further gain, suggesting that internal probe signals provide complementary information to output-form features while avoiding an additional transcription model. Here, N_{call} denotes the number of generated tool-call tags.

Table 12: Pre-execution signal complementarity on the diagnostic subset. The fused probe improves over text/format signals, while adding ASR confidence gives little additional gain.

Signal	AUROC	Req.
Text + JSON + N_{call}	0.7410	None
Probe + text + JSON + N_{call}	0.7925	None
All signals + ASR	**0.7970**	Whisper

D.9 Error Audit by Uncertainty Region

The uncertainty penalty is strongest for erroneous executions that the fused probe assigns high risk to. It does not replace the task reward for all errors. We therefore audit whether UC-GRPO reduces both high-uncertainty errors and low-uncertainty errors. Table 13 groups the diagnostic subset into uncertainty tertiles and compares wrong-execution rates before inference-time routing.

Table 13: Wrong-execution audit by fused uncertainty region. All values are percentages. UC-GRPO reduces wrong executions in every region, with the largest reduction in high-uncertainty examples.

Bucket	GRPO wrong	UC-GRPO wrong	Reduction
Low-U	49.5	42.1	-7.4
Mid-U	53.6	46.2	-7.4
High-U	73.1	58.9	-14.2

The reductions in the low- and mid-uncertainty regions occur without additional clarification, indicating that UC-GRPO does not simply defer these examples. The larger reduction in the high-uncertainty region is consistent with the uncertainty-weighted objective and selective clarification behavior.

D.10 Ablation Variant Definitions

Table 14 defines the ablation variants reported in the main paper. All variants use the same backbone, training data, prompts, and decoding settings.

Table 14: Definitions of ablation variants in the main paper.

Variant	Definition
Audio Probe Only	Uses the Audio Encoder routing probe for uncertainty rewards and routing; Decoder uncertainty is removed.
Decoder Probe Only	Uses the Decoder conditional probe for uncertainty rewards and routing; Audio routing uncertainty is removed.
GRPO + Probe Routing	Trains with standard GRPO and applies fused-probe routing only at inference.
w/o Probe in Reward	Removes probe-derived reward terms while keeping the remaining UC-GRPO training setup.
w/o U-Weighted Loss	Removes uncertainty-based loss weighting while keeping uncertainty-conditioned rewards and routing.

D.11 Computation Overhead

The uncertainty modules are lightweight logistic heads over existing hidden states. They introduce no additional speech encoder, ASR model, or repeated decoding. During inference, UC-GRPO adds mean pooling over selected Audio Encoder states and two linear probe evaluations plus the fusion head before external tool execution. This architecture adds only small probe computations beyond the existing forward pass. However, measured end-to-end latency, throughput, and peak-memory comparisons are needed to quantify the overhead.

E Prompt Design

Figures 3 and 4 summarize the system prompt and evaluation logic used for speech tool calling. The system prompt exposes three actions to the model: executing a tool call, asking for clarification, or answering directly. The evaluator parses the response, computes structured-call metrics, and applies probe-based routing before execution.

F Additional Results

Figure 5 shows the trend on audio-adapted When2Call for the Qwen3-Omni-30B backbone, standard GRPO, and UC-GRPO. Across clean speech and the 5dB, 10dB, and 15dB noisy conditions, UC-GRPO consistently achieves the highest Acc Norm and F1. This result complements the main table by showing that the gain is not limited to the noisy average, but remains visible across individual audio conditions.

G Limitations

Our evaluation focuses on single-turn spoken tool use with synthesized and human-recorded requests under controlled acoustic perturbations. This setting supports systematic comparisons across acoustic conditions, while generalization to spontaneous multi-turn conversations remains to be studied. The post-training experiments also focus on one speech-model family. Future work can examine transfer across additional backbones and how clarification affects task recovery and interaction cost when users provide further information.