SPEECHGYM: AN AUDIO-NATIVE GYM FOR TRAINING VOICE AGENTS VIA REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Voice agents must call tools and hold multi-turn dialogue entirely through speech, yet the dominant paradigm trains them in text. Existing frameworks either cascade TTS and ASR around a proprietary voice API, whose weights cannot be trained and whose per-call cost makes on-policy reinforcement learning prohibitive, or stay in text: they measure voice agents but cannot improve them. We present SpeechGym, an audio-native agentic environment in which two omni-modal models converse in native audio, with no external ASR or TTS and no API boundary, over the unmodified tasks, tools and success check of an established text agentic benchmark, so that the interaction modality is the only variable and the loop stays local and trainable end to end. The failures speech introduces are perceptual rather than reasoning deficits: the agent picks the right tool and the right argument slot but fills it with a value misheard from the waveform, and that single error cascades into a failed call, a retry of the same call, and a wasted step budget. A second failure is behavioural: under an insistent user the agent performs an unauthorised write and ends the episode believing it helped. Both are trainable, because the environment labels them for free: a call with a misheard argument fails against the database while a correct one succeeds. The obstacle is sparsity, not signal. Outcome-only GRPO is gradient-starved here, since most rollout groups fail identically, while a per-turn process reward crediting each successful tool call restores variance to nearly every group. Trained this way, the agent transfers with no further tuning to an independently implemented voice benchmark, more than doubling task success and lifting an open-weights model above proprietary systems such as GPT-Realtime-2 and Gemini-Live, while using fewer tokens than before training. Along the way it acquires repair behaviours no reward names, such as asking a user to spell a misheard name, and the same checkpoint, which never saw audio question answering, improves on three public audio reasoning and understanding benchmarks.

1 INTRODUCTION

A voice agent that changes a booking or disputes a charge must do everything a text agent does—call tools against a live database, respect a domain policy, drive a multi-turn dialogue to a verifiable end state—with speech as its only channel. The dominant recipe trains the policy in text and attaches speech at the edges, assuming competence acquired in text survives the round trip through audio. Omni-modal models (Xu et al., 2025; Qwen Team, 2025; Défossez et al., 2024; Fu et al., 2024; Xie & Wu, 2024) dissolve this cascade architecturally but supply no way to *train* in that regime: RL on them has so far addressed single-turn tasks with no tools, no dialogue partner and no environment state (Rouditchenko et al., 2025; Fan et al., 2026; Zhao et al., 2025), while agentic RL is almost entirely textual (Qian et al., 2025; Modecrua et al., 2026). How to train an agent natively in audio, and what breaks once the text channel is gone, remains open.

τ-Voice (Ray et al., 2026) makes the gap concrete by wrapping τ^2-bench (Barrès et al., 2025), itself an extension of τ-bench (Yao et al., 2024), in a voice loop: a user LLM writes each user turn, a TTS service speaks it, and the agent is a proprietary real-time API. It reports a steep drop from text to speech, but cannot close the gap it exposes: a closed API cannot be trained, and its latency and price rule out the rollout volume on-policy RL needs (Section 2). Voice-agent benchmarks (Chen et al.,

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.

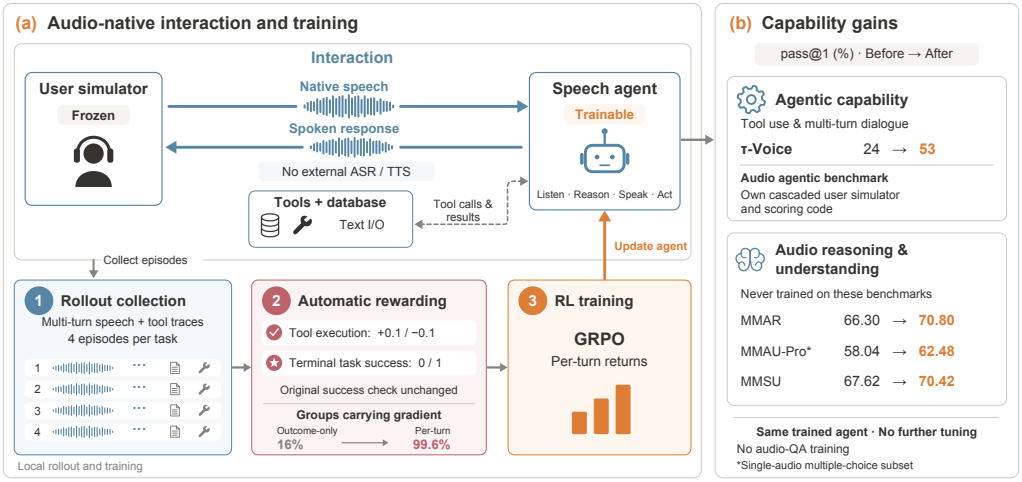

Figure 1: **SpeechGym: train in one environment, improve multiple capabilities.** **(a)** The loop is audio-native and entirely local: a frozen user simulator speaks in native audio and the trainable agent answers in speech, with no external ASR or TTS between them and only the tool channel in text. Episodes are scored by τ^2-bench's outcome verifier, with per-turn credit added for tool calls, and turned into a policy update by GRPO. **(b)** One training run improves agentic success and audio reasoning and understanding. Agentic success is measured on τ-Voice (Ray et al., 2026), a new benchmark whose user simulator the agent never met; the same untuned checkpoint never saw audio question answering.

2026; Wang et al., 2025a; Jain et al., 2025; Lin et al., 2026) share the property: voice agents can be measured but not improved.

We introduce **SpeechGym** (Fig. 1), an audio-native agentic environment whose loop is entirely local and therefore trainable. Two omni-modal models converse in native audio: a frozen user model π_U plays the user, and the trainable Thinker–Talker agent π_θ chooses at every step between making a structured tool call, executed deterministically against the database, and speaking back to the user. With no external ASR, TTS or API boundary in the loop, rollouts cost only local compute and the agent's weights are open to training. Tasks, tools, databases, policies and the success check are inherited unmodified from the underlying text benchmark, so only the interaction modality changes.

Our contributions are threefold:

- **An audio-native gym that makes voice agents trainable.** It is, to our knowledge, the first environment that combines audio-native interaction, multi-turn tool use and RL training of the agent; prior audio frameworks are evaluation-only (Section 2). It is designed to accept any text agentic domain of this shape and any open omni-modal model, and an omni-modal vLLM (Kwon et al., 2023) rollout server keeps training local, several times faster, at $0 API cost (Section 3).
- **A diagnosis and a recipe.** The most frequent failure that speech adds is perceptual: the agent mishears a slot value an order of magnitude more often in speech than in text, and that one error cascades into tool errors and dead loops (Table 2). Because success is rare, most outcome-only GRPO (Shao et al., 2024) groups carry no gradient; a per-turn process reward (Ng et al., 1999) makes nearly all of them informative (Table 1). Ablations show that both audio-native rollouts and the per-turn reward are needed (Table 5).
- **Gains that transfer.** Untuned, on a new benchmark with a user it never met (τ-Voice), the trained agent more than doubles pass@1, from 24% to 53%, lifting the open 30B-A3B model above proprietary systems such as GPT-Realtime-2 and Gemini-Live (Fig. 2) while using fewer tokens. Unauthorised writes fall by more than half, the agent learns repairs that no reward names, such as asking a user to spell a name (Table 3 and Fig. 3), and the same checkpoint improves on three audio reasoning benchmarks it never saw (Table 4).

2 Related Work

Agents and voice-agent benchmarks. Tool-using agents act on external state (Schick et al., 2023; Qin et al., 2024; Zhou et al., 2024; Jimenez et al., 2024), but in text; τ-bench (Yao et al., 2024) and τ^2-bench (Barrès et al., 2025) score customer service with a simulated user and a final-state check. Standard environments make capabilities trainable, as the Arcade Learning Environment did for RL (Bellemare et al., 2013). τ-Voice (Ray et al., 2026) brings τ^2-bench to voice with a TTS-voiced user and a proprietary real-time agent, so it measures but cannot train: the agent's weights are closed, and its price precludes on-policy RL (Appendix A.1). Other voice and audio benchmarks (Jain et al., 2025; Lin et al., 2026; Chen et al., 2026; Wang et al., 2025a; Ma et al., 2025; Kumar et al., 2026; Wang et al., 2025b) only evaluate.

Omni-modal models. Qwen2.5-Omni (Xu et al., 2025) introduces the Thinker–Talker design, Qwen3-Omni (Qwen Team, 2025) scales it with Mixture-of-Experts, and VITA (Fu et al., 2024), Mini-Omni (Xie & Wu, 2024) and Moshi (Défossez et al., 2024) are open speech-to-speech models. None is trained with RL for multi-turn spoken tool use.

RL for reasoning and tool use. RL with verifiable rewards elicits reasoning (DeepSeek-AI, 2025); we use GRPO (Shao et al., 2024; Schulman et al., 2017) with per-turn shaping (Ng et al., 1999). Audio RL targets single-question reasoning (Zhao et al., 2025; Rouditchenko et al., 2025; Fan et al., 2026), and text RL for tool use mostly a single call (Qian et al., 2025) or τ-bench's text mode (Modecrua et al., 2026). To our knowledge, no prior work applies RL to omni-modal agents that call tools over multi-turn native speech.

3 SpeechGym

SpeechGym takes a text agentic benchmark and makes it audio-native and trainable, leaving the task definition alone: tasks, tools, databases and the success check are inherited unmodified, and only the channel between user and agent changes, from text to native speech. We instantiate it on τ^2-bench (Barrès et al., 2025) throughout, but nothing in the design is specific to that suite.

3.1 Problem formulation

An audio agentic episode is a partially observable Markov decision process $(\mathcal{S}, \mathcal{A}, \mathcal{O}, P, R)$ with machine-checkable elements. The state $s \in \mathcal{S}$ is the hidden domain database — customer records, reservations, order lines, account balances — never observed directly and never described in the context, read only through a documented read tool and changed only by a write tool invoked with correct arguments. An observation $o \in \mathcal{O}$ is either a user audio waveform from the frozen user model or the textual result of an executed tool. An action $a \in \mathcal{A}$ is a tool call $a_{\text{tool}} = (\texttt{name}, \texttt{args})$, emitted as structured text and dispatched to the executor, or a text response a_{speech} that the Talker renders as speech; which one it emits is part of the policy's decision at every step (Section 3.2). Tool calls transition s deterministically, with invalid calls returning an error and leaving it untouched; a speech action instead conditions the user model, which replies in audio. The episode ends on the user's resolution signal or at the step budget $T_{\max}$.

This is the text-mode POMDP of τ-bench (Yao et al., 2024) with raw audio in place of the user's transcript, which introduces three difficulties: slot values must be extracted from a waveform rather than copied from a string, so one mis-perceived character propagates into a tool argument; prosody can carry urgency, hesitation and insistence, which may bias the decision to act; and the policy must alternate between tool-call syntax and free spoken language within one context.

3.2 Environment architecture

SpeechGym has four components, drawn together in Appendix B.2.1.

User model π_U. A frozen Qwen3-Omni-30B-A3B (Qwen Team, 2025), given the task scenario (persona, goal, what the user knows), generates its turns as native audio through its own Talker, with no external text-to-speech stage. Never updated, it fixes the speech distribution against which policy improvement is measured.

Agent model π_θ. The trainable policy: the same Thinker–Talker omni-modal model, Qwen3-Omni-30B-A3B (Qwen Team, 2025). The Thinker consumes the audio and tool observations, reasons in text, and *autonomously* decides the action type: output containing a tool-call pattern is parsed and dispatched; otherwise the Talker synthesises it into speech.

Tool executor $\mathcal{E}$. τ^2-bench's executor and databases, inherited unmodified. Failed calls return an error message and leave the state unchanged.

Evaluator. τ^2-bench's outcome verifier, inherited unchanged (Section 3.3).

Why tool calls stay textual. Only the user–agent channel is audio; tool calls and results stay structured text. This separates *perceptual* error — mishearing the user and writing a wrong slot value — from *behavioural* error — wrong tool, skipped step, action without authorisation: over a noiseless tool interface, a wrong argument must have been misheard or mis-reasoned, not corrupted in transit. Tool calls also execute instantly and do *not* consume a conversational turn, so the agent may chain several before speaking, whereas a speech action always advances the dialogue with π_U.

Domains. We instantiate SpeechGym on the domains of τ^2-bench, including dual-control Telecom, where the user also holds tools.

3.3 Reward

We inherit τ^2-bench's outcome verifier as the reward: used unchanged, it decides whether the task is completed, so a SpeechGym reward and a τ^2-bench score mean the same thing. The verifier defines five binary components — a database-state match, the required information strings reaching the user, the required write tools called with correct arguments, the user's device reaching its expected state, and the task's natural-language assertions — of which each task selects a subset, its reward basis $\mathcal{B}(\tau)$ (Appendix B.2.2). The episode reward is the product over that basis,

$$R(\tau) \;=\; \prod_{c \,\in\, \mathcal{B}(\tau)} r_c, \qquad r_c \in \{0, 1\}, \tag{1}$$

so every required condition must hold; there is no partial credit, and an episode that exhausts the step budget scores $R = 0$.

Eq. (1) is identical in training and evaluation, and modality-independent: it reads database rows, tool arguments and required strings, never how the interaction was conducted. The only training-time addition is the shaping of Section 3.5, which leaves the success check untouched. Reward parity was audited: the SpeechGym reward path reproduces τ^2-bench's scores exactly on golden trajectories, and the vLLM and non-vLLM code paths agree.

3.4 Training with GRPO

For each task τ_i we sample K complete speech episodes under the current policy: full multi-turn conversations with the frozen user model interleaved with tool executions, terminating on the user's signal or at $T_{\max}$. GRPO (Shao et al., 2024) replaces PPO's value function (Schulman et al., 2017) with a group baseline: with $r_{i,j}$ the return of the j-th rollout of τ_i and μ_i, σ_i the group's mean and standard deviation,

$$\hat{A}_{i,j} \;=\; \frac{r_{i,j} - \mu_i}{\sigma_i + \epsilon}, \tag{2}$$

so a group with identical returns has $\sigma_i = 0$ and no gradient (Section 3.5). In the per-turn variant the same normalisation is applied per turn: G_t is standardised by the group statistics and broadcast to the turn's tokens. The update optimises

$$\mathcal{L}(\theta) \;=\; -\sum_{i,j} \min\!\big(\rho_{i,j}\hat{A}_{i,j},\; \mathrm{clip}(\rho_{i,j}, 1-\varepsilon, 1+\varepsilon)\,\hat{A}_{i,j}\big) \;+\; \beta\, D_{\mathrm{KL}}\big[\pi_\theta \,\|\, \pi_{\text{ref}}\big], \tag{3}$$

with $\rho_{i,j}$ the importance ratio to the policy that generated the rollout. We adopt the KL-free variant, $\beta = 0$, relying on the clip term and the adapter's low-rank constraint to stay near π_{ref}. Only a LoRA adapter (Hu et al., 2022) is trained, on the linear projections of the Thinker of a Mixture-of-Experts backbone. The speech-synthesis path is untouched: credit assignment is scoped to *what the agent does*, not *how it sounds*.

Table 1: **The per-turn reward keeps groups informative.** Because success on audio agentic tasks is rare, most groups are all-zero under outcome-only reward; per-turn shaping keeps nearly all of them informative. Statistics are over the training rollouts of the two reward configurations on the same task suite.

	Outcome-only	**Per-turn (ours)**
Groups carrying gradient ($\sigma_i > 0$)	16%	**99.6%**
Groups skipped ($\sigma_i = 0$)	84%	**0.4%**

3.5 Densifying the reward: per-turn process shaping

Eq. (1) is computed once, at termination, and on audio agentic tasks success is rare: most outcome-only groups contain no success (Table 1). When almost every episode fails, almost every group of K rollouts fails *identically*: all returns are zero, $\sigma_i = 0$ in Eq. (2), the group yields no gradient, and most rollout compute produces no learning signal.

Following shaping practice (Ng et al., 1999; Modecrua et al., 2026), we credit progress per turn. Each successful tool execution receives $+0.1$ and each failed call -0.1; the outcome $R(\tau)$ is appended to the last step, so the success check is unchanged, although the shaped objective is not potential-based (Section 4.3). Per-turn returns are discounted sums,

$$G_t \;=\; \sum_{k=t}^{T} \gamma^{\,k-t} r_k, \tag{4}$$

with discount γ, and replace the flat episode reward in Eq. (2). Table 6 lists γ, K, $T_{\max}$ and the adapter settings. Failing rollouts can now differ in how far they got, so variance appears in groups where none succeeded: four failing rollouts give $\{0, 0, 0, 0\}$ under outcome-only reward and no gradient, whereas under per-turn shaping the same group might give $\{0.30, -0.10, 0.20, 0.10\}$ (Appendix B.2.3), with $\sigma_i > 0$ and a gradient toward the rollout with more successful tool calls. Table 1 reports the effect on the fraction of groups that carry gradient.

3.6 System: making online speech RL affordable

Rollout collection, not the gradient step, is the bottleneck: an episode is dozens of turns, each generating audio from two 30B models. We serve both with vLLM-Omni (Kwon et al., 2023), one request per turn with audio in and text plus speech out, and serve the trained LoRA per request by adapter name, so an updated policy reaches the rollout workers without reloading weights. A training epoch becomes $5.4\times$ faster than with Hugging Face `transformers` (Appendix B.4), and the loop has $0 API cost, where an API cascade charges for every rollout (Section 2). Deployment, near-on-policy sampling and full pseudocode are in Appendices B.4 and C.

4 Experiments

We ask what breaks when an agentic task moves from text to speech (Section 4.2), whether SpeechGym's reward is dense enough to train on (Section 4.3), whether the result survives outside it (Section 4.4), what changed in the agent (Sections 4.5 and 4.6), how far the change reaches (Section 4.7), and what each component of the recipe contributes (Section 4.8).

4.1 Experimental setup

Models. Agent and frozen user simulator are both Qwen3-Omni-30B-A3B (Qwen Team, 2025); only the agent is updated, through a LoRA adapter (Hu et al., 2022) on the Thinker (Section 3.4), so the user is a fixed — if idealised — speech distribution. Appendix B lists every training and decoding setting.

Training in SpeechGym. We instantiate SpeechGym on τ^2-bench (Barrès et al., 2025): the agent trains, in speech, on the training split of its Airline, Retail and dual-control Telecom domains with

Table 2: **Failure modes amplified by speech.** Fraction of SpeechGym rollouts exhibiting each failure under the two channels; same tasks, same tools, same reward.

Failure type	Speech	Text	Ratio	Interpretation
Mishearing (slot value)	32%	2%	16×	hears the wrong name / ID / digit
Tool error rate	42%	26%	1.6×	misheard arguments → downstream errors
Dead loop	29%	18%	1.6×	retries the same failing call, no recovery

GRPO and the per-turn reward (Sections 3.4 and 3.5), and the checkpoint at the end of training is evaluated as is.

Evaluation on a new benchmark. We then test the agent, with no further tuning, on τ-Voice (Ray et al., 2026) under its standard protocol: a new benchmark over the same three domains, with its own user simulator, which the agent never met during training. pass@1 requires the final database state to match the one produced by the gold actions *and* all required information to reach the user. For audio we evaluate the same checkpoint on MMAR (Ma et al., 2025), MMAU-Pro (Kumar et al., 2026) (its single-audio multiple-choice subset) and MMSU (Wang et al., 2025b), against the base model and against the same run without the per-turn reward (*outcome-only*).

4.2 What breaks in speech

We annotated SpeechGym rollouts of the same τ^2-bench tasks, tools and reward in the two channels. Mishearing a slot value is near-absent in text and common in speech, an order-of-magnitude amplification, and tool errors and dead loops rise with it (Table 2; Appendix E.1 shows one task in both channels). The agent picks the right tool and argument slot, then fills it with a misheard value — a digit of a zip code, a character of an order ID, a spelling of a name — so the plan is right and only the perceived value is wrong: a perceptual, not a reasoning, failure. Because the database check is exact, that one error cascades into a failed call, a retry of the identical call, and a loop that burns the step budget and ends at zero reward. A second failure is behavioural: under an insistent user the agent performs a write it was not authorised to make. Both are trainable, because the environment already penalises them: a call carrying a misheard argument fails against the database while a correct one succeeds, and an unauthorised write fails the final-state check.

4.3 Training in SpeechGym: outcome-only versus per-turn reward

Because few episodes succeed, most outcome-only groups are K identical failures and carry no gradient (Section 3.5): only a small minority of groups produce one, and the rest are generated and discarded (Table 1; Appendix B.2.3). Under per-turn shaping nearly every group is informative, since two rollouts that both fail still differ in how many tool calls they got right. A non-zero variance shows that a gradient exists, not that it points the right way; the ablation in Section 4.8 tests that. The per-turn credit is not potential-based, so it is the terminal criterion — τ^2-bench's unmodified check — and not the objective that is preserved. These statistics come from inside SpeechGym; the headline result is measured outside it (Section 4.4).

4.4 Generalising to a new benchmark: τ-Voice

τ-Voice tests generalisation directly: the agent runs with *no further tuning* against a user simulator it never met in training (Section 4.1), so a gain cannot come from adapting to SpeechGym's own user. The base and the trained agent go through the same τ-Voice pipeline at a single attempt, so only the agent's weights differ between them.

Overall pass@1 more than doubles, with gains in every domain (Fig. 2, left); the gain holds when averaged across four seeds (Appendix D.1). After training, the open 30B-A3B model surpasses GPT-Realtime-2, GPT-Realtime-1.0, Gemini-Live-2.5 and the cascaded baseline, all of which it trailed before (right). Their scores are taken from the leaderboard, so this is not a controlled head-to-head comparison. Telecom, the dual-control domain where the user also holds tools, remains the hardest for both agents.

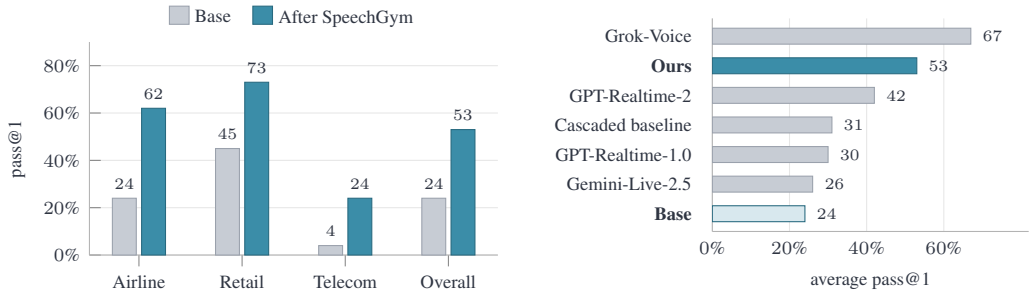

Figure 2: **Transfer to a new benchmark, τ-Voice** (Ray et al., 2026). *Left:* the SpeechGym-trained agent, with no further tuning, against a user simulator it never met in training; both bars in a group come from the same pipeline at a single attempt, and only the agent's weights differ. Gains in all three domains. *Right:* the same model against several voice agents on the τ-Voice leaderboard under the same standard protocol; we did not run them. After training, it surpasses GPT-Realtime-2, GPT-Realtime-1.0, Gemini-Live-2.5 and the cascaded baseline. For a fair comparison, we follow τ-Voice and report the macro average over domains.

Table 3: **What changes across the whole τ-Voice benchmark** (every task, both agents). *Top:* the failures diagnosed in Section 4.2, as % of tasks. *Bottom:* pass@1 split by whether a task's gold solution requires a state-changing call. pass@1 here pools all tasks, whereas Fig. 2 reports the macro average over domains. Twenty further markers are in Appendix D.2.

	Base	**After SpeechGym**	Δ
Behaviour (% of tasks)			
Unauthorised writes (lower is better)	23	10	−13
Dead loops when stuck (lower is better)	14	5	−9
Recovers after a mishearing	42	62	+20
pass@1 by task type (%)			
Requires a state-changing call	27	54	+27
Refusal or read-only	18	44	+26

4.5 What changes across the whole benchmark

A doubling of success invites the suspicion that a metric was gamed. Annotating every τ-Voice task for both agents gives three checks against the most common ways of gaming a metric (Table 3). **The diagnosed failures move:** unauthorised writes and dead loops both fall by more than half, and recovery after a mishearing rises — the agent got better at exactly what Section 4.2 said was broken, rather than better at something incidental. **It is not a refusal bias:** the gain is as large on tasks whose gold solution requires a state-changing call, where refusing scores zero, as on the rest, so an agent that had merely learned to decline more would have scored nothing there. **It is not bought with length:** the trained agent uses slightly fewer tokens per task than the base (Appendix D.2), so the gain does not come from the two cheapest ways to inflate a success rate — acting until something works, or stalling.

4.6 Emergent repair behaviours

Training produces repair behaviours that the reward never names, and the evidence is not only anecdotal. We scored twenty behaviour markers on every τ-Voice task for both agents (definitions and thresholds in Appendix D.2), and all twenty move in the intended direction (several are complementary pairs, such as database state correct and incorrect): the trained agent more often spells out or corrects a misheard value and confirms before acting, and less often loops on a failing call, rambles, or ends a task confidently wrong. The largest moves are in the outcome itself: the final database state is correct far more often, and the agent far less often ends a task confidently wrong. Fig. 3 shows two episodes in which both agents mishear a name and the first lookup fails. In the first, the base agent asks the user to repeat, fails again and hands over to a human, while the trained agent

Case 1 — asks the user to spell it out (τ-Voice retail)

Base (success 0)	**After SpeechGym** (success 1)
"Yusuf Gonzalez, nine, one, four, five, five."	*"Yusuf Gonzalez, nine, one, four, five, five."*
`find_user_id_by_name_zip(Yousef, Gonzalez, 91455)` → `Error: User not found`	`find_user_id_by_name_zip(Yousef, Gonzalez, 91455)` → `Error: User not found`
"…Could you please double-check and provide your first name, last name, and zip code again?"	"…Could you please spell out your first and last name letter by letter to ensure accuracy?"
"Yusuf Gonzalez, nine, one, four, five, five."	*"Y, U, S, U, F—G, O, N, Z, A, L, E, Z."*
`find_user_id_by_name_zip(Yousef, Gonzalez, 91455)` → `Error: User not found`	`find_user_id_by_name_zip(YUSUF, GONZALEZ, 91455)` → `yusuf_gonzalez_8900`
"…I'll need to transfer you to a human agent who can assist you further."	*[order lookups]*
	`return_delivered_order_items(#W1679211)` → `ok`
	"…The refund of $346.93 will be issued to your original payment method (PayPal)."

Case 2 — tries another spelling, then switches to the right tool (τ-Voice retail)

Base (success 0)	**After SpeechGym** (success 1)
"Lei Li. Zip code is eight, five, zero, three, three."	*"Lei Li, eight, five, zero, three, three."*
`find_user_id_by_name_zip(Lay, Lee, 85033)` → `Error: User not found`	`find_user_id_by_name_zip(Lay, Lee, 85033)` → `Error: User not found`
"…Could you please double-check and provide your first name, last name, and zip code again?"	"…Could you please provide your email address instead?"
*[the user repeats; the identical call is issued **seven** times in all]*	*"Sorry, I really don't have one."*
"I said I don't have one. This isn't working. Bye."	*[nothing new from the user]*
	`find_user_id_by_name_zip(Lei, Li, 85033)` → `lei_li_6575`
	[later, exchanging the laptop]
	`exchange_delivered_order_items(#W5166363)` → `Error: Non-delivered order cannot be exchanged`
	"…still pending, so it cannot be exchanged. We can modify the order instead."
	`modify_pending_order_items(#W5166363)` → `ok`

Figure 3: **Two τ-Voice episodes, base versus trained, on the same task.** Excerpts are verbatim; *[bracketed]* text summarises elided turns, and each strip is the RMS envelope of that turn's actual audio (user, agent). Both agents mishear the name and the lookup fails; only the trained agent repairs the channel. Success is τ-Voice's database-verified check.

Table 4: **Audio reasoning and understanding, never trained on.** pass@1 on MMAR, the single-audio multiple-choice subset of MMAU-Pro, and MMSU; Δ is relative to the base model; per-sub-task results are in Appendix D.3.

Benchmark	**Base**	**Outcome-only**	**Per-turn (ours)**	Δ
MMAR (Ma et al., 2025)	66.30	68.20	**70.80**	+4.50
MMAU-Pro (Kumar et al., 2026)	58.04	61.71	**62.48**	+4.44
MMSU (Wang et al., 2025b)	67.62	69.50	**70.42**	+2.80

asks for a spelling and completes the refund. In the second, the base agent reissues the identical call until the user gives up, while the trained agent retries with another spelling of the same name and later, when the exchange tool fails because the order is still pending, switches to the modify tool. The two episodes are illustrations; the marker counts above are the benchmark-wide evidence.

4.7 Beyond agentic tasks: audio reasoning and understanding

If what training improves includes perception, the gain may reach beyond agentic tasks. On three benchmarks the checkpoint never trained on, pass@1 rises on every one; outcome-only training already lifts all three, and the full recipe lifts them further (Table 4). A text-trained control stays at the base level, so these gains come from RL in the audio-native environment rather than coming for free (Appendix D.4). The gains are not confined to speech, the only audio the agent heard in training. On MMAR and MMAU-Pro the largest sub-task gains are on non-speech sound and on spatial audio, larger than on speech itself, while pure music, the audio furthest from training, barely moves (Appendix D.3). Consistent with this, MMSU, which is entirely spoken language, gains the least of the three. On MMSU the largest gain is consonant–vowel perception, a phoneme-level skill close to the mishearings of Fig. 3, followed by background-scene recognition and speaker attributes such as age, accent and emotion; some reasoning sub-tasks, such as polysemy reasoning, also rise,

Table 5: **Ablations on τ-Voice** (pass@1, %). Each row changes one part of the recipe; all start from the same base checkpoint and are scored untuned, in speech, by τ-Voice's own harness. Overall is the macro average over domains, following τ-Voice.

Method	Ablation	Airline	Retail	Telecom	Overall
Full recipe (ours)	—	**62**	**73**	**24**	**53**
Outcome-only	terminal reward only, no per-turn credit	48	61	11	40
Text mode	trained over text instead of speech	40	54	12	35
Base (no RL)	no RL training in SpeechGym	24	45	4	24

while a minority decline, among them pause perception (unlike pause-based reasoning) and pun interpretation. In the reasoning traces, on paired items from MMAR and MMAU-Pro under the same prompt and greedy decoding, the trained model more often cites concrete acoustic properties and tracks temporal order, far less often denies that the needed cue is in the audio or ends without an answer, and nearly stops falling into dead loops (Appendix D.5; two examples are in Appendix E.2). The one marker that barely moves is eliminating competing options. Fewer unanswered items may account for part of the gain, so the traces suggest, without proving, that the change lies more in perception than in reasoning.

4.8 Ablations: what each component contributes

Each row of Table 5 changes one part of the recipe, and every row is scored the same way: untuned, in speech, inside τ-Voice. *Base (no RL)* is the starting checkpoint, so everything above it is due to training. *Text mode* keeps the recipe and the Qwen3-Omni user simulator, but user and agent converse in plain text instead of audio during training; it asks whether the skill must be learned in the channel it is used in. It improves on the base model, but by only part of what the full recipe gains: a misheard slot value is near-absent in text (Table 2), so training over text rarely meets the failure that dominates speech, so the rollouts should be audio-native. *Outcome-only* keeps speech but scores each episode by the terminal reward alone, asking whether the per-turn reward is needed. GRPO still improves the agent, yet it falls short of the full recipe in every domain, consistent with most of its groups being identical failures that carry no gradient (Table 1).

The domains separate the two components. Outcome-only training gains most in Airline and least in Telecom, where success is rarest; the per-turn reward then adds a similar margin in every domain, so its benefit does not hinge on how far outcome-only training got. Text mode trails the full recipe most in Airline and Retail, whose tasks turn on names, reservation codes and order IDs the user speaks, and least in Telecom, consistent with speech mattering most where slot values must be heard. Overall, both ablations land between the base model and the full recipe: neither audio-native rollouts nor the per-turn reward can be dropped.

5 Conclusion

In this paper, we propose SpeechGym, to our knowledge the first audio-native agentic environment that both evaluates and trains voice agents in speech. Two open omni-modal models converse in native audio over the unmodified tasks, tools and success check of an existing text agentic benchmark, so the whole loop runs locally and can be trained end to end. Using it, we find that the most frequent failure that speech adds is a perceptual rather than a reasoning deficit, and that closing the gap takes both audio-native rollouts and a per-turn process reward: training over text transfers only partly, and outcome-only GRPO leaves most groups without a gradient. The trained agent generalises untuned to a new, independent benchmark, where it surpasses several proprietary voice systems while using fewer tokens; it repairs a noisy channel in ways no reward specifies, and it improves on audio reasoning and understanding benchmarks it never trained for. Because SpeechGym leaves the task definition untouched, any text agentic domain of the same shape can be made audio-native and trainable in the same way. Closing the loop turns the text-to-audio gap from an open measurement into an optimisation problem with an objective, a gradient and a stopping criterion.

AI USE STATEMENT

We used generative AI tools only to aid or polish writing: copy-editing prose and improving the phrasing of passages written by the authors. We did not use them to generate research ideas, to design the environment or the training method, to write the experimental code, or to produce, analyse or interpret any result; all numbers reported in this paper come from runs executed and checked by the authors. Beyond this authoring use, generative models are also the object of study: the agent and the user simulator are open omni-modal models, as described in Section 3.2. We have reviewed all AI-assisted text and verified every claim against the underlying runs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our work develops a training environment and method for voice agents. Our experiments instantiate SpeechGym on the open-source τ^2-bench, whose databases are synthetic and whose users are fictional, and every user is simulated by a model, so no human subjects or real personal information are involved; the audio benchmarks we evaluate on are public. We do not introduce ethical concerns beyond those that voice agents already raise.

REPRODUCIBILITY STATEMENT

We have made efforts to ensure that our results can be reproduced. The environment is described in Section 3, with its components, reward and a worked example of the per-turn shaping in Appendix B.2; all training and decoding settings are listed in Table 6 (Appendix B.1), and Appendix C gives the training loop as pseudocode. The evaluation protocol is in Appendix B.3 and the serving setup in Appendix B.4. Our experiments instantiate SpeechGym on public resources: the tasks, tools, databases and outcome verifier of τ^2-bench (Barrès et al., 2025), and the open Qwen3-Omni-30B-A3B checkpoint (Qwen Team, 2025) used for both agent and user simulator. Evaluation uses public benchmarks: τ-Voice (Ray et al., 2026) under its standard protocol, and MMAR, MMAU-Pro and MMSU (Ma et al., 2025; Kumar et al., 2026; Wang et al., 2025b). The behaviour markers are defined in Appendix D.2, and the reasoning-trace detector by the keyword lists in Appendix D.5.

REFERENCES

Victor Barrès, Honghua Dong, Soham Ray, Xujie Si, and Karthik Narasimhan. τ^2-bench: Evaluating conversational agents in a dual-control environment. *CoRR*, abs/2506.07982, 2025. doi: 10.48550/ARXIV.2506.07982. URL https://doi.org/10.48550/arXiv.2506.07982.

Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *J. Artif. Intell. Res.*, 47:253–279, 2013. doi: 10.1613/JAIR.3912. URL https://doi.org/10.1613/jair.3912.

Yiming Chen, Xianghu Yue, Chen Zhang, Xiaoxue Gao, Robby T. Tan, and Haizhou Li. Voicebench: Benchmarking llm-based voice assistants. *Trans. Assoc. Comput. Linguistics*, 14:378–398, 2026. doi: 10.1162/TACL.A.628. URL https://doi.org/10.1162/tacl.a.628.

DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL https://doi.org/10.48550/arXiv.2501.12948.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. Moshi: a speech-text foundation model for real-time dialogue. *CoRR*, abs/2410.00037, 2024. doi: 10.48550/ARXIV.2410.00037. URL https://doi.org/10.48550/arXiv.2410.00037.

Jiajun Fan, Roger Ren, Jingyuan Li, Rahul Pandey, Prashanth Gurunath Shivakumar, Ivan Bulyko, Ankur Gandhe, Ge Liu, and Yile Gu. Incentivizing consistent, effective and scalable reasoning capability in audio llms via reasoning process rewards. In Carl Vondrick, Bharath Hariharan, Colin Raffel, Lerrel Pinto, Diyi Yang, and Aleksandra Faust (eds.), *The Fourteenth International Conference on Learning Representations, ICLR 2026, Rio de Janeiro, Brazil, April 23-27, 2026*. proceedings.iclr.cc,

2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/04a80267ad46fc730011f8760f265054-Abstract-Conference.html.

Chaoyou Fu, Haojia Lin, Zuwei Long, Yunhang Shen, Meng Zhao, Yifan Zhang, Xiong Wang, Di Yin, Long Ma, Xiawu Zheng, Ran He, Rongrong Ji, Yunsheng Wu, Caifeng Shan, and Xing Sun. VITA: towards open-source interactive omni multimodal LLM. *CoRR*, abs/2408.05211, 2024. doi: 10.48550/ARXIV.2408.05211. URL https://doi.org/10.48550/arXiv.2408.05211.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Dhruv Jain, Harshit Shukla, Gautam Rajeev, Ashish Kulkarni, Chandra Khatri, and Shubham Agarwal. Voiceagentbench: Are voice assistants ready for agentic tasks? *CoRR*, abs/2510.07978, 2025. doi: 10.48550/ARXIV.2510.07978. URL https://doi.org/10.48550/arXiv.2510.07978.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik R. Narasimhan. Swe-bench: Can language models resolve real-world github issues? In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=VTF8yNQM66.

Sonal Kumar, Simon Sedlácek, Vaibhavi Lokegaonkar, Fernando López, Wenyi Yu, Nishit Anand, Hyeonggon Ryu, Lichang Chen, Maxim Plicka, Miroslav Hlavácek, William Fineas Ellingwood, Sathvik Udupa, Siyuan Hou, Allison Ferner, Sara Barahona, Cecilia Bolaños, Satish Rahi, Laura Herrera-Alarcón, Satvik Dixit, Rupali S. Patil, Soham Deshmukh, Lasha Koroshinadze, Yao Liu, Leibny Paola García-Perera, Eleni Zanou, Themos Stafylakis, Joon Son Chung, David Harwath, Chao Zhang, Dinesh Manocha, Alicia Lozano-Diez, Santosh Kesiraju, Sreyan Ghosh, and Ramani Duraiswami. Mmau-pro: A challenging and comprehensive benchmark for holistic evaluation of audio general intelligence. In Sven Koenig, Chad Jenkins, and Matthew E. Taylor (eds.), *Fortieth AAAI Conference on Artificial Intelligence, Thirty-Eighth Conference on Innovative Applications of Artificial Intelligence, Sixteenth Symposium on Educational Advances in Artificial Intelligence, AAAI 2026, Singapore, January 20-27, 2026*, pp. 22688–22697. AAAI Press, 2026. doi: 10.1609/AAAI.V40I27.39430. URL https://doi.org/10.1609/aaai.v40i27.39430.

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 Jason Flinn, Margo I. Seltzer, Peter Druschel, Antoine Kaufmann, and Jonathan Mace (eds.), *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pp. 611–626. ACM, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

Guan-Ting Lin, Chen Chen, Zhehuai Chen, and Hung-yi Lee. Full-duplex-bench-v3: Benchmarking tool use for full-duplex voice agents under real-world disfluency. *CoRR*, abs/2604.04847, 2026. doi: 10.48550/ARXIV.2604.04847. URL https://doi.org/10.48550/arXiv.2604.04847.

Ziyang Ma, Yinghao Ma, Yanqiao Zhu, Chen Yang, Yi-Wen Chao, Ruiyang Xu, Wenxi Chen, Yuanzhe Chen, Zhuo Chen, Jian Cong, Kai Li, Keliang Li, Siyou Li, Xinfeng Li, Xiquan Li, Zheng Lian, Yuzhe Liang, Minghao Liu, Zhikang Niu, Tianrui Wang, Yuping Wang, Yuxuan Wang, Yihao Wu, Guanrou Yang, Jianwei Yu, Ruibin Yuan, Zhisheng Zheng, Ziya Zhou, Haina Zhu, Wei Xue, Emmanouil Benetos, Kai Yu, Eng-Siong Chng, and Xie Chen. MMAR: A challenging benchmark for deep reasoning in speech, audio, music, and their mix. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December*

5, 2025, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/610a7d6507d55be70c6d13d0b663227d-Abstract-Datasets_and_Benchmarks_Track.html.

Wachiravit Modecrua, Krittanon Kaewtawee, Krittin Pachtrachai, and Touchapon Kraisingkorn. Multi-turn reinforcement learning for tool-calling agents with iterative reward calibration. *CoRR*, abs/2604.02869, 2026. doi: 10.48550/ARXIV.2604.02869. URL https://doi.org/10.48550/arXiv.2604.02869.

Andrew Y. Ng, Daishi Harada, and Stuart Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In Ivan Bratko and Saso Dzeroski (eds.), *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999), Bled, Slovenia, June 27 - 30, 1999*, pp. 278–287. Morgan Kaufmann, 1999.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tur, Gokhan Tur, and Heng Ji. Toolrl: Reward is all tool learning needs. In Danielle Belgrave, Cheng Zhang, Laura N. Montoya, Hsuan-Tien Lin, Razvan Pascanu, Piotr Koniusz, Marzyeh Ghassemi, Nancy Chen, Iván Vladimir Meza Ruíz, and Arturo Loaiza-Bonilla (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2025, NeurIPS 2025, San Diego, CA, USA, December 2-7, 2025 / Mexico City, Mexico, November 30 - December 5, 2025*, 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/97c5b2707228e7e3fb67e4ecc2e0e607-Abstract-Conference.html.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. Toolllm: Facilitating large language models to master 16000+ real-world apis. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=dHng2O0Jjr.

Qwen Team. Qwen3-omni technical report. *CoRR*, abs/2509.17765, 2025. doi: 10.48550/ARXIV.2509.17765. URL https://doi.org/10.48550/arXiv.2509.17765.

Soham Ray, Keshav Dhandhania, Victor Barrès, and Karthik Narasimhan. τ-voice: Benchmarking full-duplex voice agents on real-world domains. *CoRR*, abs/2603.13686, 2026. doi: 10.48550/ARXIV.2603.13686. URL https://doi.org/10.48550/arXiv.2603.13686.

Andrew Rouditchenko, Saurabhchand Bhati, Edson Araujo, Samuel Thomas, Hilde Kuehne, Rogério Feris, and James R. Glass. Omni-r1: Do you really need audio to fine-tune your audio llm? In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2025, Honolulu, HI, USA, December 6-10, 2025*, pp. 1–7. IEEE, 2025. doi: 10.1109/ASRU65441.2025.11434780. URL https://doi.org/10.1109/ASRU65441.2025.11434780.

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. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine (eds.), *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL http://papers.nips.cc/paper_files/paper/2023/hash/d842425e4bf79ba039352da0f658a906-Abstract-Conference.html.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL http://arxiv.org/abs/1707.06347.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

Bin Wang, Xunlong Zou, Geyu Lin, Shuo Sun, Zhuohan Liu, Wenyu Zhang, Zhengyuan Liu, AiTi Aw, and Nancy F. Chen. Audiobench: A universal benchmark for audio large language models. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, pp. 4297–4316. Association for Computational Linguistics, 2025a. doi: 10.18653/V1/2025.NAACL-LONG.218. URL https://doi.org/10.18653/v1/2025.naacl-long.218.

Dingdong Wang, Jincenzi Wu, Junan Li, Dongchao Yang, Xueyuan Chen, Tianhua Zhang, and Helen Meng. MMSU: A massive multi-task spoken language understanding and reasoning benchmark. *CoRR*, abs/2506.04779, 2025b. doi: 10.48550/ARXIV.2506.04779. URL https://doi.org/10.48550/arXiv.2506.04779.

Zhifei Xie and Changqiao Wu. Mini-omni: Language models can hear, talk while thinking in streaming. *CoRR*, abs/2408.16725, 2024. doi: 10.48550/ARXIV.2408.16725. URL https://doi.org/10.48550/arXiv.2408.16725.

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. *CoRR*, abs/2503.20215, 2025. doi: 10.48550/ARXIV.2503.20215. URL https://doi.org/10.48550/arXiv.2503.20215.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A benchmark for tool-agent-user interaction in real-world domains. *CoRR*, abs/2406.12045, 2024. doi: 10.48550/ARXIV.2406.12045. URL https://doi.org/10.48550/arXiv.2406.12045.

Shuaijiang Zhao, Tingwei Guo, Cheng Wen, Bajian Xiang, Wei Zou, and Xiangang Li. Ke-omni-r: Achieving advanced audio reasoning with a concise 50-words think process. https://github.com/shuaijiang/Ke-Omni-R, 2025.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=oKn9c6ytLx.

A DISCUSSION

Voice agents have been easy to measure and hard to improve. Every benchmark that scores them puts a commercial API where the learning signal would have to be: a text user simulator feeding a speech engine on one side, a proprietary real-time endpoint on the other. That design ranks systems precisely and improves none of them, because the agent's weights are closed and the per-rollout price rules out the volume on-policy RL consumes. SpeechGym removes both boundaries by making each side a local open model, and the rest of this paper is what became visible once the loop closed.

Key insight. The text-to-audio gap need not be a fixed property of the modality. Once both sides of the conversation are models we run, the gap becomes an optimisation problem with an objective, a gradient and a stopping criterion — and what has to be optimised turns out to be mainly perception rather than reasoning.

We state the evidence as five findings, each tied to a specific result.

Finding 1. Understanding audio is not the same as acting on it. The base 30B model answers multiple-choice audio questions reasonably well (Table 4), yet in spoken dialogue it mishears a slot value in 32% of rollouts, against 2% in text (Table 2).

Pre-training buys comprehension. It does not buy the ability to hold a service conversation while calling tools, which is why that ability needs an environment of its own.

Finding 2. The dominant failure is perceptual, and the environment already penalises it. A call carrying a misheard argument fails against the database and a correct one succeeds, so τ^2-bench's outcome verifier supplies the signal, with no reward we had to invent (Sections 3.3 and 4.2).

This is what makes the diagnosis actionable. Had the deficit been one of reasoning, there would be no cheap external signal saying which of several plausible plans was right.

Finding 3. The obstacle is sparsity, not signal. With success this rare, most outcome-only groups of rollouts fail identically and carry no gradient; crediting each tool call as it lands raises the share of groups that produce a gradient from 16% to 99.6% (Table 1).

The fix is not a new algorithm but a change in what is rewarded. Dense shaping for sparse-reward RL is classical; what is new is the measurement that audio agentic tasks sit exactly in the regime where the sparse variant starves.

Finding 4. The skill survives outside the environment that produced it, and it is not a metric artefact. Evaluated untuned on a new, independently implemented benchmark, the agent more than doubles its pass@1; the gain is as large on tasks that require a state-changing call as on tasks that do not, and tokens per task fall (Sections 4.4 and 4.5).

Three checks against gaming come out clean: the diagnosed failures are the ones that move, refusing more would have scored zero on the action-required split, and buying success with more interaction would have raised token counts rather than lowered them. The agent also acquires repairs no reward names — asking a user to spell a name, retrying with another spelling, abandoning a tool that cannot apply (Section 4.6).

Finding 5. What the environment teaches reaches beyond agentic tasks. The same checkpoint, trained only on agentic speech tasks with no audio question answering, improves on three audio reasoning benchmarks, and its reasoning traces cite concrete acoustic evidence more often and deny that the cue is present far less often (Section 4.7).

The marker that barely moves is the reasoning one: eliminating competing options. That asymmetry suggests that the change lies more in hearing than in reasoning, though fewer unanswered items may also contribute (Appendix D.5).

A.1 DESIGN RATIONALE

Four choices in the environment deserve explicit justification, because each could reasonably have gone the other way.

The user side is a model we run, not a service we call. A frozen open omni-modal checkpoint generates the user's turns directly as audio, so a rollout costs local compute rather than per-call credit and nothing in the loop is closed to training. This single decision is what separates a gym from an evaluation harness (Section 2). The prices decide it: at the providers' list rates a single epoch of rollouts over this task suite costs over $200 in API calls, a seven-epoch run over $1,400, and a long on-policy run far more — while the agent behind the endpoint still cannot be trained. Local rollouts cost GPU time instead.

The user model is frozen. Freezing fixes the speech distribution against which improvement is measured, so a gain cannot come from the two sides drifting into a private protocol that is easier for both. The cost is that the user is idealised; unfreezing it is the obvious next turn of that knob, and the interface already permits it.

Tool calls stay textual. Only the user–agent channel is audio. Over a noiseless tool interface, a wrong argument must have been misheard or mis-reasoned, never corrupted in transit, which is what lets us separate perceptual error from behavioural error at all (Appendix B.2.1).

The reward is τ^2-bench's outcome verifier. We inherit τ^2-bench's outcome verifier, unchanged, to decide whether a task is completed: an episode succeeds only if every required condition holds. Keeping the verifier unchanged is what makes our scores comparable with τ^2-bench and τ-Voice (Appendix B.2.2).

A.2 LIMITATIONS

SpeechGym is, to our knowledge, the first audio-native gym for voice agents, and our experiments show that training in audio rather than in text matters (Tables 5 and 9). Its scope, however, is everyday spoken service tasks. Other settings where an agent acts on speech, such as spoken coding assistance or voice-controlled robotics, are left to future work. Audio also makes rollouts slower than text, since every turn is synthesised and heard. vLLM-Omni serving and parallel rollouts reduce this cost (Appendix B.4), but rollout collection remains the bottleneck, and fully asynchronous RL architectures are a natural next step that can build on SpeechGym.

B EXPERIMENTAL DETAILS

B.1 MODELS AND TRAINING

Both sides of the conversation are the same open checkpoint, Qwen3-Omni-30B-A3B (Qwen Team, 2025): one frozen copy plays the user, one trainable copy is the agent. Only a LoRA adapter (Hu et al., 2022) on the Thinker's linear projections is updated, and the speech-synthesis path is left untouched, so training changes what the agent does rather than how it sounds. Table 6 lists the settings. One is worth singling out: the KL penalty is switched off entirely ($\beta = 0$), and the clip term and the adapter's low rank are what keep the policy near its reference (Section 3.4). The checkpoint at the end of training is the one evaluated throughout.

Table 6: **Training and decoding configuration.** The environment settings are those of Sections 3.1, 3.3 and 3.5; the optimiser settings are those of Section 3.4.

Policy and adapter		**Optimisation**	
Agent / user model	Qwen3-Omni-30B-A3B	Objective	GRPO, PPO-clip
Trained parameters	LoRA on the Thinker's linear projections	KL penalty	none ($\beta = 0$)
		Clip range	$\varepsilon = 0.2$
LoRA rank / α	8 / 16	Learning rate	5×10^{-6}, constant
Speech-synthesis path	not trained	Gradient clipping	1.0
Environment and reward		**Rollout decoding**	
Rollouts per group	$K = 4$	Sampling temperature	0.7
Step budget	$T_{\max} = 50$	Agent tokens per turn	≤ 512
Discount	$\gamma = 0.99$	User tokens per turn	≤ 200
Per-call credit	± 0.1	Audio in / out	native, no ASR or TTS
Terminal reward	τ^2-bench's check, unchanged		

B.2 THE ENVIRONMENT

B.2.1 THE FOUR COMPONENTS

Fig. 4 draws the environment as it is instantiated in this paper. Two of the four boxes are inherited rather than designed, and that is the point: the tool executor and the reward are the underlying benchmark's, run unmodified, so a score in SpeechGym and a score in that benchmark mean the same thing (Section 3.3). Only the channel between user and agent is changed, and only the agent is trained.

The two that are designed — a user side we run rather than a service we call, and a user model that stays frozen — are justified in Appendix A.1.

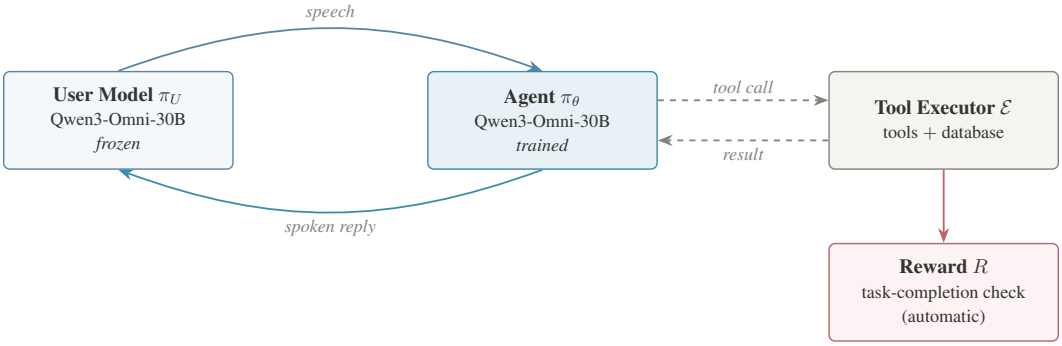

Figure 4: **The four components, as we instantiate them.** The *same* omni-modal model plays both sides: one frozen copy plays the user in native audio, while the trainable copy decides at every step between emitting a structured tool call and speaking back. The executor runs the tools against the database, and the reward is τ^2-bench's outcome verifier, computed automatically. Only the user–agent channel is audio: keeping the tool interface textual separates *perceptual* error — mishearing a slot value — from *behavioural* error — choosing the wrong tool or acting without authorisation.

B.2.2 THE REWARD, IN FULL

Section 3.3 scores an episode with τ^2-bench's outcome verifier, inherited unchanged. The verifier defines five binary components, of which each task selects a subset — its reward basis $\mathcal{B}(\tau)$ in Eq. (1):

- r_{DB} the gold action sequence is replayed on a fresh environment and the resulting database compared with the agent's; a match scores 1.
- r_{COMM} the replies contain every information string the task requires, such as a confirmation number.
- r_{ACTION} the required write tools were called with correct arguments.
- r_{ENV} the user's device reaches the expected final state, for example data enabled — the dual-control criterion.
- r_{NL} the task's natural-language assertions hold, for example that the agent confirmed the refund.

The product form of Eq. (1) is what makes the sparsity problem in Section 3.5 as sharp as it is. Because every selected component must hold, an episode that gets four of five right scores exactly what an episode that gets none right scores. A misheard identifier therefore costs the same as never attempting the task — which is precisely why the failures of Section 4.2 collapse into one undifferentiated zero, and why crediting progress per turn recovers so much signal. We keep the product form rather than softening it because the terminal criterion has to stay identical to the benchmark's for the transfer comparison in Section 4.4 to mean anything.

B.2.3 PER-TURN SHAPING, WORKED

To make Section 3.5 concrete, consider a group of $K = 4$ rollouts on one task in which *all four fail*: no episode reaches the terminal criterion, so $R(\tau) = 0$ for every one of them. Under outcome-only

reward the group is $\{0, 0, 0, 0\}$, its standard deviation is zero, and Eq. (2) yields no gradient — the rollouts are generated and then discarded.

Under per-turn shaping the same four rollouts are no longer identical, because they differ in how far each got before failing. Applying Eq. (4) with $\gamma = 0.99$ and the ± 0.1 credit of Section 3.5 to the tool calls each episode issued:

Rollout	**Tool calls, in order**	G_0
1	three that land	$+0.30$
2	one that lands, then two that fail	-0.10
3	two that land	$+0.20$
4	one that lands	$+0.10$

These are the values quoted in Section 3.5; they are an illustration of the shaping rule rather than a measured group. The group now has non-zero variance, so Eq. (2) produces an advantage that ranks rollout 1 above rollout 4 above rollout 2 — a gradient towards executing more calls that land, on a task where every episode still failed. Note what the shaping does *not* do: rollout 2 is penalised for issuing calls that fail, but nothing in the signal says *which* argument was wrong, so the environment rewards getting further, not diagnosing the mishearing. The terminal criterion is untouched — the outcome $R(\tau)$ is appended at the last step — so a rollout that completes the task outscores these four. In general, though, the shaped return can favour a long failing rollout with many successful calls over a short successful one, which is why the objective optimised is not task success itself (Section 4.3).

B.3 EVALUATION PROTOCOL

Agentic. pass@1 under τ^2-bench's unmodified task-completion check (Section 4.1). The evaluation runs inside τ-Voice (Ray et al., 2026), whose cascaded user simulator, telephony-grade audio and scoring harness are its own, so the only component of ours involved is the trained adapter. Table 5 gives the per-domain figures behind Fig. 2 and the two ablated training conditions.

Audio reasoning and understanding. The same checkpoint, with no further tuning, on MMAR (Ma et al., 2025), the single-audio multiple-choice subset of MMAU-Pro (Kumar et al., 2026), and MMSU (Wang et al., 2025b), scored on all items of MMAR and of that MMAU-Pro subset and on 4,994 of MMSU's 5,000 items. We report pass@1, against the base model and against the same training run without the per-turn process reward.

Reasoning traces. The marker analysis of Appendix D.5 uses a separate decoding pass — greedy, a 256-token budget, matched items from MMAR and MMAU-Pro, both models given the identical prompt. No accuracy in this paper is computed from it.

B.4 COMPUTATION RESOURCES

Deployment. One 8×H200 pod hosts the loop: GPUs 0–1, 2–3 and 4–5 run three vLLM-Omni servers, each hosting both models and using two GPUs for the reasoning and speech-synthesis paths; GPUs 6–7 run LoRA training in bf16. A group's $K = 4$ rollouts run on four threads round-robin across the three servers. Because rollout collection rather than the gradient step dominates the epoch (Section 3.6), this split — three quarters of the pod generating, one quarter training — is what the wall-clock actually responds to. Table 7 times one epoch under both serving paths: rollouts speed up $5.6\times$ and the policy update does not change, so a full epoch runs $5.4\times$ faster.

Table 7: **Wall-clock of one GRPO epoch** on the pod described above, serving the same models with Hugging Face `transformers` or with vLLM-Omni.

Stage	transformers	vLLM-Omni	Speedup
Rollout, one episode	265 s	47.6 s	5.6×
Rollout, one group ($K = 4$)	17.7 min	3.2 min	5.6×
Rollouts, one epoch	14.7 h	2.6 h	5.6×
Policy update	10 min	10 min	1×
Total epoch	**15 h**	**2.8 h**	**5.4×**

Near-on-policy sampling. The first group of a run is collected on base weights, since no adapter exists yet, and a group in flight when an update lands finishes under the previous adapter, so rollouts are *near*-on-policy. The clipped importance ratio in Eq. (3) is designed to tolerate this lag, but the deviation is real, and we state it rather than describe the loop as strictly on-policy.

C ALGORITHM PSEUDOCODE

Algorithm 1 states one iteration of training in full. Everything in it is fixed by the environment except the last three lines: the observation and action spaces are those of Section 3.1, the executor and the terminal reward are inherited from τ^2-bench (Section 3.3), and the per-turn credit is the shaping of Section 3.5. The one line worth reading twice is the group-discard test. Under outcome-only reward it fires for most groups, which is the whole of the sparsity problem; per-turn shaping exists to keep it from firing.

Algorithm 1 SpeechGym: one training iteration

Input: policy π_θ (LoRA adapter), frozen user model π_U, task suite, tool executor $\mathcal{E}$, group size K, step budget $T_{\max}$, discount γ, per-call credit $c = 0.1$
sample a task τ from the suite
for $j = 1$ **to** K *rollouts, round-robin across the rollout servers* **do**
  reset the environment; the hidden database returns to its initial state
  for $t = 1$ **to** $T_{\max}$ **do**
    $o_t \leftarrow$ an audio turn from π_U, or the textual result of the last tool call
    $a_t \sim \pi_\theta(\cdot \mid \text{history})$ *the Thinker chooses: tool call or speech*
    if a_t is a tool call **then**
      execute it against $\mathcal{E}$; $r_t \leftarrow +c$ if it succeeds, $-c$ if it fails
    else
      the Talker synthesises it and π_U is conditioned on the audio; $r_t \leftarrow 0$
    end if
    if the user signals resolution **then break**
  end for
  $R(\tau) \leftarrow \tau^2$-bench's outcome verifier on the final state *Eq.* (1)*, run unchanged*
  add $R(\tau)$ to the last step's reward, leaving the terminal criterion intact
  $G_t^{(j)} \leftarrow \sum_{k \geq t} \gamma^{k-t} r_k$ for every turn t *Eq.* (4)
end for
$\mu, \sigma \leftarrow$ mean and standard deviation of the group's returns
if $\sigma = 0$ **then** discard the group and return
$\hat{A}_t^{(j)} \leftarrow (G_t^{(j)} - \mu)/(\sigma + \epsilon)$ *Eq.* (2)
take one gradient step on θ under the clipped objective with $\beta = 0$ *Eq.* (3)
push the updated adapter to the rollout servers

D ADDITIONAL EXPERIMENTAL RESULTS

Appendix D.1 first checks that the τ-Voice gain is not noise across seeds. The results after it are two views of one change. In agentic rollouts the dominant failure is a misheard slot value (Section 4.2); after training, the agent recovers from a mishearing more often, loops less, and asks users to spell what it did not catch (Table 3, Appendix D.2). The audio benchmarks show the same shift from the other side: the largest gains fall on sub-tasks that turn on what the clip sounds like — telling consonants and vowels apart, identifying a sound, judging a speaker's accent, age or emotion (Appendix D.3) — and the model's reasoning traces cite acoustic evidence more often and deny that the cue is present less often, while the one marker of pure reasoning barely moves (Appendix D.5). Both are consistent with training in SpeechGym improving how well the agent hears.

D.1 VARIANCE ACROSS RUNS

Following τ-Voice, the main text reports pass@1 from a single run. We are nonetheless interested in its variance, so we extend that run to four seeds for each model, the base and the trained agent, and report the average across the four seeds (avg@4) with its standard error (Fig. 5 and Table 8). The gain is far larger than the variation across seeds: in every domain, and overall, the gap between the two models is many times the standard error. Overall, avg@4 rises from 24.51 ± 1.33 to 53.51 ± 1.08, more than double, in line with the main-text run, and Telecom, the dual-control domain, remains the hardest for both.

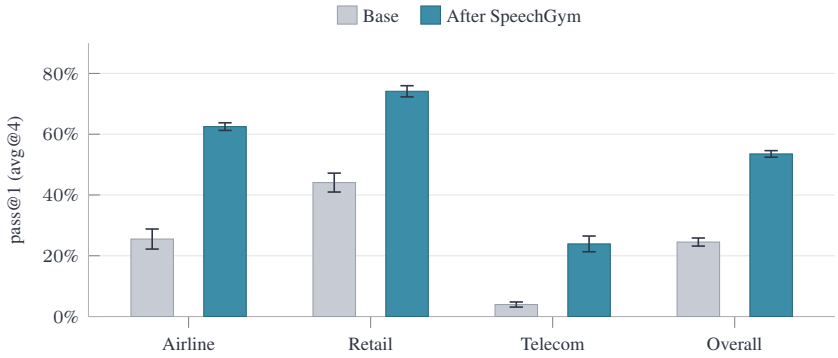

Figure 5: **Variance across runs on τ-Voice.** pass@1 (%) averaged across 4 seeds per domain (avg@4); error bars show the standard error. Overall is the macro average over domains, following τ-Voice.

Table 8: **Variance across runs on τ-Voice.** pass@1 (%) averaged across 4 seeds per domain (avg@4) $\pm$ standard error. Overall is the macro average over domains, following τ-Voice.

Model	Airline	Retail	Telecom	Overall
Base	25.50 ± 3.30	44.08 ± 3.11	3.95 ± 0.84	24.51 ± 1.33
After SpeechGym	$\mathbf{62.50 \pm 1.26}$	$\mathbf{74.12 \pm 1.84}$	$\mathbf{23.90 \pm 2.59}$	$\mathbf{53.51 \pm 1.08}$

D.2 τ-VOICE: BEHAVIOUR MARKERS

We annotated every τ-Voice task for both the base and the trained agent and counted twenty behaviour markers as a share of tasks (Fig. 6); every one moves in the intended direction, although several are complementary pairs and so are not twenty independent measurements. Tokens per task also fall, from 51.2k to 48.4k. The largest moves are in the outcome itself — the database state is correct on 20 percentage points more of the tasks, and the agent ends a task confidently wrong 20 points less often — and in how it gets there: concise solves rise, while rambling, tool dead loops and severe loops fall. The markers closest to the channel rise too: spelling out or self-correcting, and confirming before acting, are the repairs Fig. 3 shows, now counted across the benchmark. The markers are scored independently of pass@1 and of Table 3, so rows that sound alike (a dead loop "when stuck" there, four or more identical calls here) are separate counts. Thresholds: a *concise*, *fast* or *efficient* solve passes within 15 turns, within 10 turns, or with at most 5 tool calls; *rambling* and *very long* exceed 25 and 35 turns; a *tool dead loop* and a *severe loop* repeat an identical call at least 4 and 8 times; *many tool errors* means at least 3; a *low* or *high error rate* is under 20% or over 40% of calls failing.

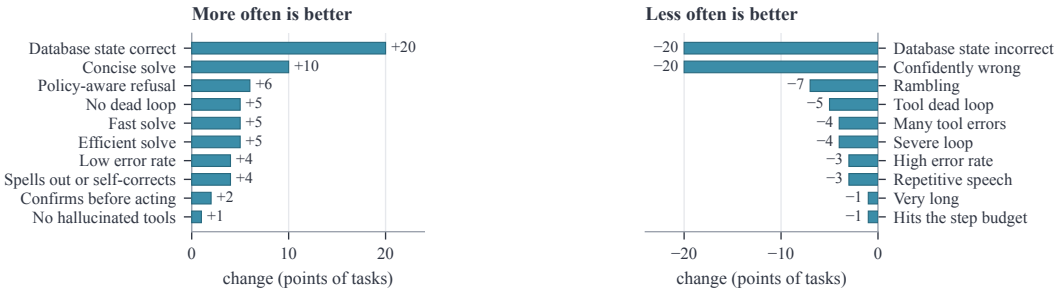

Figure 6: **Twenty behaviour markers on τ-Voice**: change after training in SpeechGym, in percentage points of tasks.

D.3 AUDIO REASONING AND UNDERSTANDING BY SUB-TASK

Figs. 7 and 8 show every sub-task on which training improves pass@1; sub-task means weighted by item count reproduce the overall figures of Table 4 exactly. On MMAR the gain is largest on sound (+11.5) and on clips that mix music with speech (+9.8) or with sound (+9.1). On MMAU-Pro it is largest, among sub-tasks of substantial size, on spatial audio (+8.9), sound (+8.0) and speech (+5.9). On MMSU the single largest gain is consonant–vowel perception (+31.1), a phoneme-level skill close to the mishearings in Fig. 3. It is followed by background-scene recognition (+13.6) and by the speaker attributes a user's voice carries: age (+13.5), accent (+12.3), emotion (+12.0) and gender (+9.2). Reasoning sub-tasks rise as well, most of all polysemy reasoning (+13.3). None of these is trained on directly.

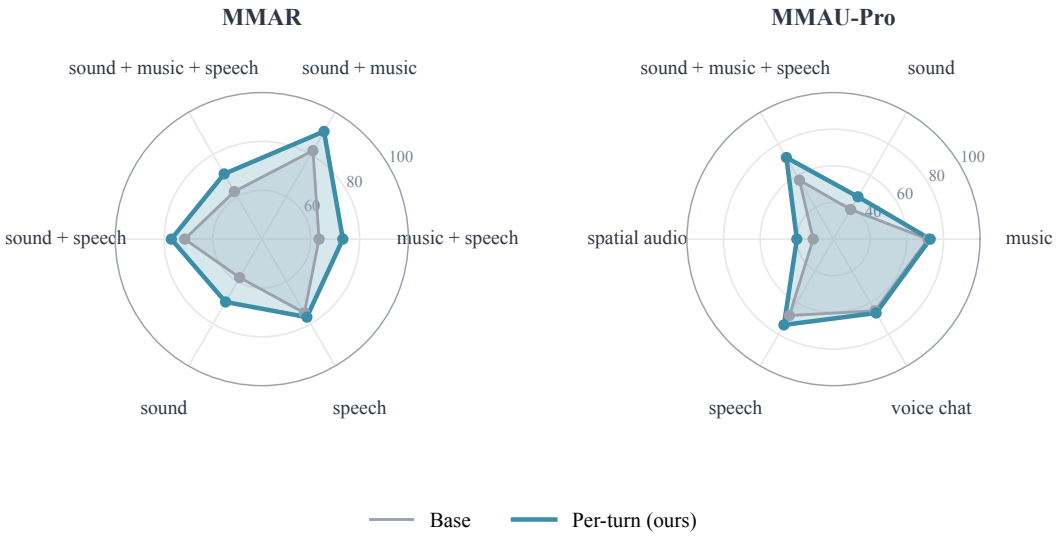

Figure 7: **MMAR and MMAU-Pro sub-tasks on which training improves pass@1** (%). MMAR's music sub-task and four MMAU-Pro sub-tasks do not improve and are not drawn.

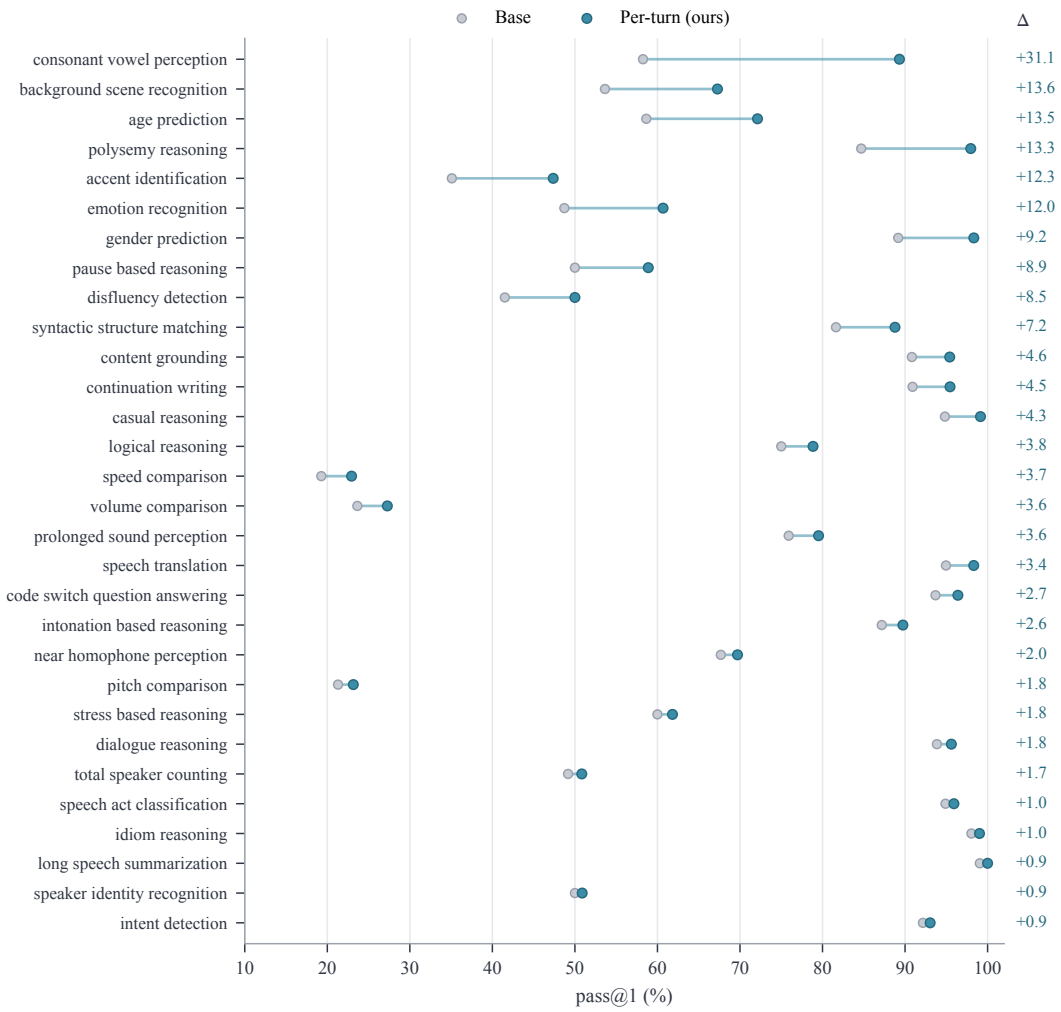

Figure 8: **MMSU sub-tasks on which training improves pass@1**, from base (grey) to trained (blue), sorted by gain. The remaining 17 of 47 sub-tasks (12 decline, 5 are unchanged) are not drawn.

D.4 AUDIO BENCHMARKS: TRAINING OVER TEXT VERSUS SPEECH

Table 9 adds the text-mode ablation of Table 5 to the audio benchmarks. That run uses the same tasks, the same Qwen3-Omni user simulator and the same per-turn recipe as ours, but user and agent converse in plain text instead of audio, so the policy never hears audio during training. It stays within one point of the base model on all three benchmarks ($+0.40$ on MMAR, -0.55 on MMAU-Pro, $+0.86$ on MMSU). Both runs trained in speech improve on all three, and on every benchmark the order is the same: text mode close to the base model, outcome-only above it, and the per-turn recipe highest.

The generalisation to audio reasoning and understanding is therefore not a free lunch of RL. Training on the same tasks with the same recipe over text leaves these benchmarks where they were, while the same training in SpeechGym's audio-native environment lifts all three: most of the gain comes from RL in the audio-native environment. The agentic ablation shows the same pattern, where training over text recovers only part of the gain in speech (Table 5).

Table 9: **Audio benchmarks after training over text versus speech** (pass@1, %). *Text mode* is the ablation of Table 5: the same recipe and Qwen3-Omni user simulator, with user and agent conversing in plain text instead of audio. The other columns repeat Table 4.

Benchmark	Base	Text mode	Outcome-only	Per-turn (ours)
MMAR	66.30	66.70	68.20	**70.80**
MMAU-Pro	58.04	57.49	61.71	**62.48**
MMSU	67.62	68.48	69.50	**70.42**

D.5 BEHAVIOUR MARKERS IN REASONING TRACES

The reasoning traces show the same change from the inside (Fig. 9). The markers of attending to the signal rise: the trained model cites a concrete acoustic property in 31.9% of traces against 24.8%, and tracks time or order in the clip and counts sources more often. The failure markers fall, most sharply ending without an answer (2.9% to 0.2%) and dead loops (1.3% to 0.1%) — the same failure that falls in agentic rollouts. Eliminating competing options, the one marker of reasoning rather than perception, barely moves.

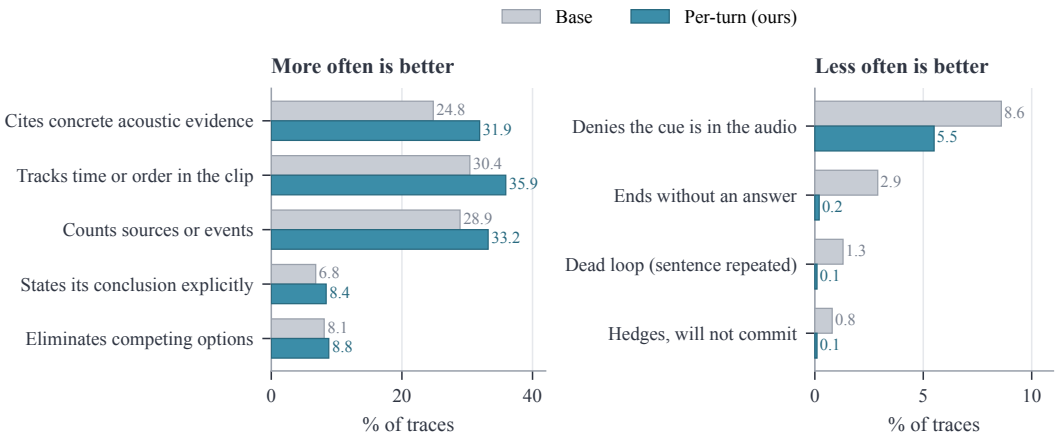

Figure 9: **Behaviour markers in reasoning traces** (% of traces), base versus trained, on the same paired items from MMAR and MMAU-Pro under the same prompt and greedy decoding.

Both models produce a trace for each item under the same prompt with greedy decoding and a 256-token budget; this pass is separate from the accuracy evaluation, whose numbers are those of Table 4. A trace shows a marker if its reasoning text matches the marker's keywords, case-insensitively and as whole words; the acoustic-evidence keywords match word-initially, so inflected forms such as *loudly* or *echoes* count. Two markers are structural: *ends without an answer* flags an empty answer field, and *dead loop* flags a trace in which some sentence longer than 25 characters occurs three or more times. Because the identical detector runs on both models over identical items, its noise — *one* also matches outside counting — affects both alike, and the paired difference is the quantity of interest. Keyword counting is a transparent proxy for what a trace does, not a semantic judgement.

Acoustic evidence pitch, timbre, rhythm, frequency, reverb, echo, tempo, loud, soft, sharp, raspy, muffled, glugging, clatter, hum, rumble, sizzl-

Time or order begins, then, after, before, over time, increases, decreases, first, second, third, sequence, followed by

Counting one, two, three, single, multiple, several, number of, how many, distinct

Elimination other options, none of the other, not consistent, rule(s) out, whereas, rather than, instead of, points away

Conclusion this indicates, therefore the, so the answer, the trend is, conclusion

Denial no information, not present, does not contain, does not provide, impossible to, cannot be determined, none of the listed, not among

Hedging not enough information, unclear, ambiguous, hard to tell, cannot tell, unknown

E Extended Qualitative Analysis

The two examples below make the aggregate results concrete. The first shows the diagnosis of Section 4.2 on a single task; the second shows the shift in hearing of Section 4.7 on single audio questions. Both are illustrations chosen to show a mechanism, not samples from which to estimate a rate; the rates are in the tables and figures they point to.

E.1 One task in two channels

Table 10 runs the same τ^2-bench retail task twice in SpeechGym, once in speech and once in text, the two channels compared in Table 2. The task, the tools and the agent are the same; only the channel between user and agent differs. In speech, the agent hears "Mei" as *maya* and "28236" as *2836*, so the account lookup fails. When the user repeats the details, it corrects the zip code but keeps the misheard name, and the second lookup fails too. The user has no email to fall back on, so the agent hands the call to a human and the episode scores 0. In text, the same values arrive exactly: the first lookup succeeds and the agent completes the exchange, scoring 1. The plan is the same in both channels — ask for name and zip code, then look up the account — and only the value that reaches the tool is wrong. This is the perceptual failure of Section 4.2, and it cascades as described there: the misheard name survives the user's repetition, so the retry fails too.

Table 10: **The same retail task in speech and in text.** Excerpts are verbatim; *[bracketed]* text summarises elided turns, and *ok* marks a successful call. In the speech column each strip is the RMS envelope of that turn's actual audio (user, agent); the text channel carries no audio. Success is τ^2-bench's database check.

Speech (success 0)	**Text** (success 1)
[user asks to exchange a water bottle and a desk lamp; has no email]	*[user asks to exchange a water bottle and a desk lamp; has no email]*
User: *"My first name is Mei and my last name is Kovacs. I live in zip code 28236."*	User: *"My first name is Mei, last name is Kovacs, and my zip code is 28236."*
`find_user_id_by_name_zip(maya, kovacs, 2836)` → Error: User not found	`find_user_id_by_name_zip(Mei, Kovacs, 28236)` → mei_kovacs_8020
Agent: "…Could you please confirm if the details you provided are correct?…"	*[the agent looks up the orders and finds the one with both items]*
User: *"Yes, my details are correct. My first name is Mei, my last name is Kovacs, and I live in zip code 28236."*	User: *"Actually, I've changed my mind. I only want to exchange the desk lamp, not the water bottle."*
`find_user_id_by_name_zip(maya, kovacs, 28236)` → Error: User not found	*[the agent finds a brighter lamp and confirms the price difference]*
[the agent asks for an email; the user has none]	`exchange_delivered_order_items(#W6390527)` → ok
`transfer_to_human_agents(...)` → Transfer successful	Agent: "The exchange for your desk lamp has been successfully requested.…"

E.2 AUDIO REASONING: TWO CASES

Table 11 shows two items, one from MMAR and one from MMAU-Pro, taken from the paired traces of Appendix D.5, on which the trained model answers correctly and the base model does not. In both, the base model's reasoning follows sensibly from what it reports hearing, but what it reports hearing is wrong. In the first, it reports the clatter of wheels where the trained model hears a treadmill motor, and it reasons its way to a playground. In the second, it hears a pepper grinder, decides that no option can be right, and repeats itself until the token budget runs out without answering — the no-answer and dead-loop markers of Appendix D.5. The trained model hears the cue that decides the question, and ordinary reasoning then reaches the right answer. This matches Section 4.7: the change lies more in hearing than in reasoning.

Table 11: **Two audio questions, trained versus base.** Reasoning is verbatim, from greedy decoding under the same prompt; . . . marks elided text. **Blue** marks the cue the trained model hears; *red italics* mark what the base model mishears and the wrong answer it leads to.

Example: hearing the decisive sound (MMAR)
Question: Where are they running **Options:** Park, Treadmill, Country road, Playground
Correct answer: Treadmill

Trained (correct)	Base (wrong)
Reasoning: *The audio contains sounds of people running and shouting encouragement. The distinct, rhythmic sound of a **treadmill motor** is audible in the background. This indicates the activity is taking place on a treadmill, likely in a gym or fitness center.*	**Reasoning:** *The rhythmic clatter of wheels on a hard surface and the sound of a group running together suggest they are on a track or paved path. The absence of natural sounds points away from a park or country road. The most likely setting is a playground with a paved surface.*
Answer: Treadmill	**Answer:** Playground

Example: identifying the source instead of denying it (MMAU-Pro)
Question: Which spice is being used for flavor? **Options:** Cumin, Basil, Thyme, Rosemary
Correct answer: Cumin

Trained (correct)	Base (wrong)
Reasoning: *The audio features the distinct sound of a **spice grinder**, followed by the sound of a spice being poured. The grinding sound is **characteristic of a hard spice like cumin**. The pouring sound is consistent with a dry spice being dispensed. The overall sounds suggest the use of a ground spice, with cumin being a likely candidate.*	**Reasoning:** *The audio features the distinct sound of a pepper grinder. None of the listed options (Cumin, Basil, Thyme, Rosemary) are ground spices. Therefore, the correct answer is not among the choices provided. . . . [the same three sentences repeat until the token budget runs out]*
Answer: Cumin	**Answer:** none given