SpeechGym: An Audio-Native Gym for Training Voice Agents via Reinforcement Learning

Jiajun Fan[1,2], Jingyuan Li[2], Prashanth Gurunath Shivakumar[2], Jia-Hong Huang[2], Qi Luo[2], M. Maruf[2], Ivan Bulyko[2], Ge Liu[1], Roger Ren[2]

[1]University of Illinois Urbana-Champaign [2]Amazon AGI Foundations

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, where gradients cannot flow and 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. Audio agentic capability does not follow from audio understanding. 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 caller 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 almost every rollout group fails 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 carrying an open-weights model from last place to second on that leaderboard, while using fewer turns and tokens than before training.

Date: August 28, 2026

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 [5, 11, 30, 32, 33] dissolve the cascade architecturally but supply no way to *train* in that regime: RL on them has addressed only single-turn tasks with no tools, no dialogue partner and no environment state [8, 25, 36], while agentic RL is almost entirely textual [20, 22, 37]. How to train an agent natively in audio, and what breaks once the text safety net goes, remains open.

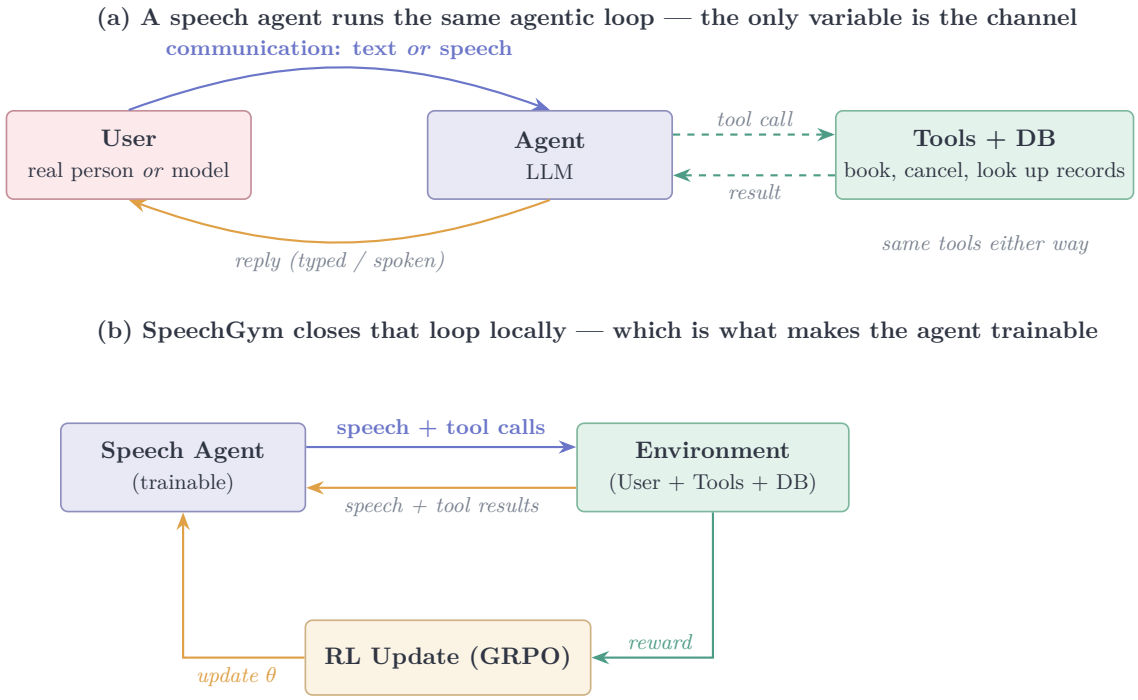

Figure 1 SpeechGym **overview. (a)** A speech agent runs the same agentic loop as a text agent: it talks to a user, calls tools against a live database, and is scored on the final state. The one thing that changes is the channel carrying the conversation — which is why holding tasks, tools and the success check fixed makes modality a controlled variable. **(b)** SpeechGym closes that loop locally. The user simulator, the tools and the reward all live inside the environment, so speech rollouts can be scored and turned into a policy update rather than merely measured. No external ASR or TTS and no proprietary API sits in the loop, which is what makes gradients — and therefore training — possible at all.

τ-Voice [24] makes the gap concrete by wrapping τ^2-bench [1], itself an extension of τ-bench [35], in a voice loop: a user LLM writes the caller's turn, a TTS service speaks it, and the agent is a proprietary real-time API. It reports a steep text-to-speech drop, but cannot close the gap it exposes: gradients do not flow through a closed API, and its latency and price rule out the rollout volume on-policy RL needs (Section 2). Voice-agent benchmarks [3, 13, 19, 31] share the property: voice agents can be measured, not improved.

We introduce **SpeechGym** (Figure 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 speaks the caller's side, and the trainable Thinker–Talker agent π_θ chooses at every step between 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 policy stays ours. Everything but the interaction modality is inherited unmodified from the underlying text benchmark—tasks, tools, databases, policies, success check—so any difference in success is attributable to modality alone.

In general, our contributions are as follows:

- **An audio-native agentic gym.** SpeechGym is, to our knowledge, the first environment to combine audio-native interaction, multi-turn tool use and end-to-end RL trainability; prior frameworks supply at most two, and every prior audio framework is evaluation-only because an API cascade on the user side can be evaluated but never trained (Section 2). It exposes a reset/step/reward interface that accepts any text agentic domain of this shape and any open omni-modal model, and adds a Banking domain stressing high-stakes numeric slots (Section 3).
- **A diagnosis of the text-to-audio gap.** The dominant failures are perceptual, not reasoning deficits: the base 30B omni model mis-hears a slot value in 32% of speech rollouts against 2% in text, a sixteen-fold increase that cascades into a 42% tool-error rate and into dead loops ending the episode at zero

reward. Alongside it sits confidently-wrong over-action, observed throughout the speech rollouts but not quantified against text (Table 1, Section 4.2).

- **A training recipe for the low-base-rate regime.** Because GRPO [4, 28] normalises returns within a group, near-floor audio success rates leave only 16% of groups carrying any gradient; a per-turn process reward, dense shaping in the classical sense [21], raises this to 99.6% (Table 2, Section 4.3). A vLLM-based [15] omni-modal rollout server makes a training epoch 5.4× faster at $0 API cost.
- **Cross-pipeline transfer.** Run with no further tuning on τ-Voice's independently implemented pipeline—a different user simulator, TTS and ASR stack, and evaluation harness—the trained agent more than doubles pass@1, from 24% to 53%, with gains in all three domains and non-overlapping confidence intervals (Figure 3, Section 4.4), moving an open 30B model from last place to second on that leaderboard, ahead of the cascaded baseline and of proprietary real-time systems as reported by their providers and not re-run by us (Figure 4). The agent also uses fewer turns and tokens (Table 4), issues fewer unauthorised writes, falls into fewer dead loops, and recovers from mis-hearings more often (Table 3).

The text-to-audio gap is therefore not a fixed cost of the modality but a deficit in identifiable competences—hearing a slot value correctly, confirming before acting, abandoning a failing plan—that closed-loop training substantially reduces.

2 Related Work

2.1 Tool-using language agents

Tool augmentation makes a language model an agent that acts on external state [23, 26]; τ-bench [35] made this rigorous: customer service as a POMDP with a simulated user, documented database tools and an automatic final-state check. τ^2-bench [1] generalises it to a dual-control Dec-POMDP where the user also holds tools. Web navigation [38], software engineering [14] and executable multi-hop retrieval [29] stress other axes but remain textual. Standardised RL environments turn a capability into a trainable, reproducible target: the Arcade Learning Environment [2] sustained a decade of algorithmic work that eventually pushed past human world records [6, 7]. SpeechGym plays that role for voice agents, keeping τ-bench's evaluation methodology while making the channel audio-native and the loop trainable.

2.2 Voice agent evaluation

τ-Voice [24] is the closest prior effort: it extends τ^2-bench to voice, driving the caller's side through a cascade—a text user simulator feeding a TTS system—while the agent is a proprietary full-duplex realtime voice API. It measures rather than trains: no gradient flows through a proprietary endpoint, and the cascade's latency and price preclude the rollout volume on-policy RL needs—one epoch costs over $200 in API calls, a seven-epoch run over $1,400, and thousand-epoch budgets approach $200k for a single run. SpeechGym runs the same loop locally at $0 API cost. Other audio benchmarks share the scope: VoiceAgentBench [13] and Full-Duplex-Bench [19] assess spoken tool use without training; VoiceBench [3] and AudioBench [31] test single-turn spoken understanding without tools.

2.3 Omni-modal models

Qwen2.5-Omni [33] introduces the Thinker–Talker design, whose hidden states drive an autoregressive speech-token decoder; Qwen3-Omni [30] scales it with a Mixture-of-Experts backbone, and VITA [11], Mini-Omni [32] and Moshi [5] pursue open speech-to-speech interaction. They understand audio and generate speech, but none is trained for audio *agentic* tasks: when to invoke a tool rather than speak, when to confirm a misheard value, when a write is authorised. SpeechGym supplies that signal.

2.4 RL for reasoning and tool use

RL with verifiable rewards is now standard for eliciting reasoning, in general domains [4] and in mathematics [34]. We optimise with GRPO [28], a group-relative alternative to PPO's learned value function [27], with per-turn shaping in the classical sparse-reward sense [21]. In audio, RL has so far targeted reasoning over audio inputs [25, 36], including with process-level rewards over the reasoning trace [8]; in all of these an episode is a single question and the model never acts on external state. In text, RL for tool use has largely optimised a single call per episode, whether through reward design [22] or procedure-aware supervision of the call [37]; closest on the task side, multi-turn GRPO has been applied to τ-bench's text mode [20]. Outside language, post-training and adaptive computation for large multimodal policies are driven by the same constraint we face — the cost of acting, not of updating — whether by scheduling and pruning a vision-language-action model [17], learning when to deliberate at all [16], or adapting inference structure to the input [18]. To our knowledge, no prior work applies RL to audio *agentic* tasks, where perception, dialogue and tool use are optimised jointly.

2.5 Positioning

Three axes matter—audio-native interaction, multi-turn tool use, trainability by online RL—and prior work supplies at most two: τ-bench, τ^2-bench [1, 35] and single-call tool-use RL [22, 37] are trainable but text-only, and every prior audio framework [3, 13, 19, 24, 31] is evaluation-only. The decisive difference is the user side: an API cascade can be measured but never differentiated through, and its per-rollout cost rules out on-policy training. The claim needs care: the text benchmarks' public code releases do expose a Gymnasium-compatible interface, so their *text* domains can in principle be trained in, even though they are presented and used as evaluation suites. No such route exists for the voice setting: there the agent is a proprietary realtime endpoint and the caller is synthesised by a commercial TTS service, so the audio loop admits no gradient at either end. SpeechGym is, to our knowledge, the only system simultaneously audio-native, multi-turn, tool-using and trainable end to end by RL, because both sides of the conversation are local open models inside the agent's loop. The roles are complementary: we train in SpeechGym and evaluate, untuned, on τ-Voice's pipeline and scoring code (Section 4.4).

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 [1] 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: it is 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 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 variable-length speech response $a_{\text{speech}} \in \mathbb{R}^L$ synthesised as a waveform; which one it emits is part of the policy's decision at every step (Section 3.2).

Tool calls transition s deterministically: read tools return records without changing it, write tools may change it, and invalid or unauthorised calls return an error message leaving it untouched. A speech action instead conditions the frozen user model, which replies in audio. The episode ends on the user's resolution signal or at the step budget $T_{\max} = 50$.

This is the text-mode POMDP of τ-bench [35] 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

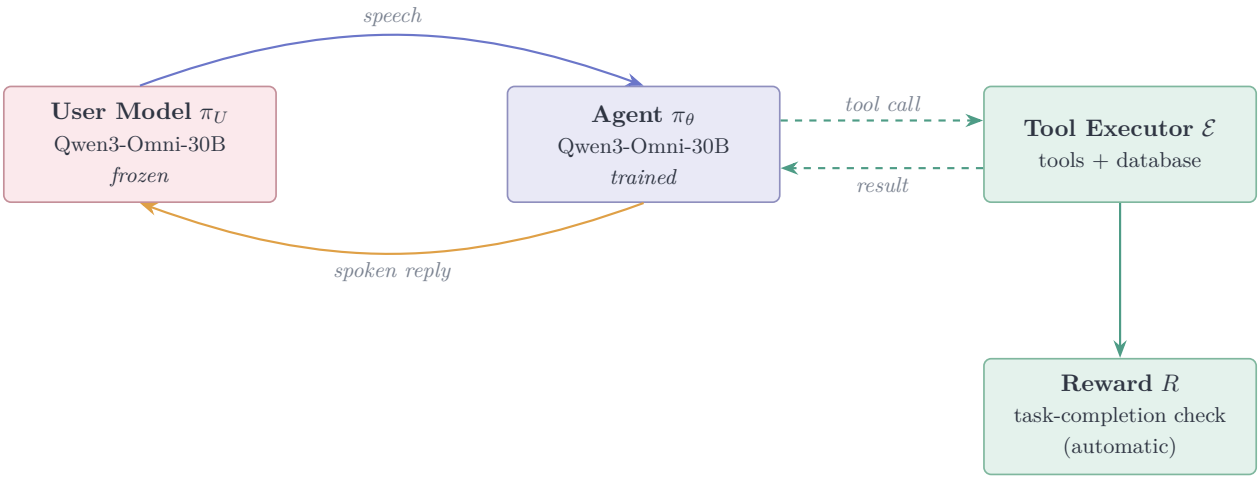

Figure 2 The four components, as we instantiate them. The *same* omni-modal model plays both sides: one frozen copy speaks the caller's side in native audio, while the trainable copy decides at every step between emitting a structured tool call and speaking back. The executor runs the tools against the database, and the reward is the benchmark's own task-completion check, 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.

mis-perceived character propagates into a tool argument; prosody carries urgency, hesitation and insistence, which 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 (Figure 2).

User model π_U. A frozen Qwen3-Omni-30B-A3B [30], given the task scenario (persona, goal, what the user knows), generates its turns directly as native audio, with no intermediate 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 [30, 33]. 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. No external controller or scripted schedule governs that choice.

Tool executor $\mathcal{E}$. The benchmark's own executor and databases, unmodified. Failed calls return an error message and leave the state unchanged.

Evaluator. The benchmark's own outcome check, run 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 caller and writing a wrong slot value — from *behavioural* error — wrong tool, skipped step, action without authorisation: over a noiseless tool interface a wrong argument was 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. The three τ^2-bench domains — Airline, Retail and Telecom, the last dual-control, where the user also holds tools the agent must talk them through — plus Banking, for high-stakes numeric slots. Banking is built like a τ^2-bench domain: accounts, balances and transactions in a relational database; read tools for lookup, write tools for transfers, disputes and limit changes; per-task gold action sequences verified in text

mode; and a final database-state check. τ-Voice [24] covers only the three original domains, hence Banking's absence from Section 4.4.

Trainability. With no intermediate ASR or TTS and both models open and local, the loop is differentiable at the agent and free to sample from; behind a proprietary voice API gradients cannot cross the boundary, and latency and price cap rollouts (Section 3.6).

3.3 Reward

We do not design a reward: the episode outcome is scored by τ^2-bench's own evaluator, run unchanged, so a SpeechGym reward and a τ^2-bench score mean the same thing. The evaluator defines five binary components, of which each task selects a subset — its reward basis $\mathcal{B}(\tau)$:

$r_{\mathbf{DB}}$ the gold action sequence is replayed on a fresh environment and the resulting database compared with the agent's; a match scores 1.

$r_{\mathbf{COMM}}$ the replies contain every information string the task requires, such as a confirmation number.

$r_{\mathbf{ACTION}}$ the required write tools were called with correct arguments.

$r_{\mathbf{ENV}}$ the user's device reaches the expected final state, for example data enabled — the dual-control criterion.

$r_{\mathbf{NL}}$ the task's natural-language assertions hold, for example that the agent confirmed the refund.

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 `max_steps` 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 terminal outcome 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 $K = 4$ complete speech episodes under the current policy: full multi-turn conversations with the frozen user model interleaved with tool executions, terminating by user signal or at $T_{\max}$. GRPO [28] replaces PPO's value function [27] 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 process 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

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

with $\rho_{i,j}$ the importance ratio to the policy that generated the rollout. How tightly to hold a fine-tuned policy to its reference is an active design axis in RL post-training of generative models, from adaptive divergence regularisation [10] to reward-weighted objectives with transport regularisation [9]. 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 [12] is trained: rank 8, $\alpha = 16$, 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*.

3.5 Densifying the reward: per-turn process shaping

Eq. (1) is computed once, at termination, and audio agentic tasks sit in a low-base-rate regime (Section 4.2). 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 [20, 21], 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 terminal criterion is preserved. Per-turn returns are discounted sums,

$$G_t \;=\; \sum_{k=t}^{T} \gamma^{\,k-t} r_k, \qquad \gamma = 0.99, \tag{4}$$

and replace the flat episode reward in Eq. (2). 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 process shaping the same group might give $\{0.30, -0.10, 0.20, 0.10\}$, with $\sigma_i > 0$ and a gradient toward the rollout that executed more tools successfully. Table 2 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 [15]: the rollout worker calls an OpenAI-compatible endpoint with `modalities:[text,audio]`, user audio passed as base64 WAV and the agent's speech returned alongside its text, so a turn is one request rather than a chain of conversions. The trained LoRA is served per request by adapter name, so an updated policy reaches the workers without reloading a merged checkpoint.

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.

Effect. Multi-turn speech rollouts, not the policy update, dominate the epoch, so serving them efficiently is what makes online speech RL practical: end to end, a training epoch becomes 5.4× faster. An API cascade running the same rollouts has a per-epoch price that puts a full training run out of reach (Section 2); SpeechGym's training loop has $0 API cost.

Near-on-policy sampling. The first group of a run is collected on base weights, no adapter existing 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.

4 Experiments

We ask what breaks when an agentic task moves from text to speech (Section 4.2), whether the reward signal inside SpeechGym is dense enough to train on (Section 4.3), whether the result survives outside the training environment (Section 4.4), and why (Section 4.5).

4.1 Setup

Models. Agent and frozen user simulator are both Qwen3-Omni-30B-A3B [30], an omni-modal mixture-of-experts model with native audio input and output. Only the agent is updated, through a LoRA adapter [12] on the Thinker (Section 3.2, Section 3.4); the user is never trained, making it a fixed — if idealised — speech distribution. One model family on both sides keeps the loop local and trainable (Section 3.6); these results are reported in that setting.

Table 1 Failure modes amplified by speech. Fraction of SpeechGym rollouts exhibiting each failure under the two channels; same tasks, same tools, same reward. Every mode is amplified by speech — mis-hearing is near-absent in text, and the downstream tool errors and dead loops are markedly rarer — which is why a speech-native environment is needed to expose them, and to supply the learning signal that fixes them.

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

Compute. All runs use one 8×H200 pod with vLLM-Omni rollouts, which makes a training epoch 5.4× faster end to end at $0 API cost since both models are local.

Domains. Airline, Retail and the dual-control Telecom domain, inherited unmodified from τ^2-bench [1], plus Banking, added for long, high-stakes numeric slots. Tasks, tools, databases and the success check are τ^2-bench's, so the only variable between text and speech runs is the channel.

Metric. pass@1 under τ^2-bench's unchanged task-completion check: an episode succeeds only if the final database state matches the state produced by replaying the gold actions *and* all required information has reached the caller. τ-Voice [24] uses the same criterion, so our in- and out-of-environment numbers are comparable. The check reads structured outcome fields only, rewarding neither fluent nor awkward-sounding speech.

Two evaluation axes. *In-gym* results, against the training-time user simulator and clean self-play audio, are diagnostic only; the headline result is measured entirely *outside* the training environment, on τ-Voice (Section 4.4).

4.2 What breaks in speech

We annotated rollouts of the same tasks, tools and reward in the two channels; only the channel differs. Mis-heard slot values appear in 32% of speech rollouts against 2% in text, a 16× amplification; tool errors rise from 26% to 42%, dead loops from 18% to 29% (Table 1). Three patterns dominate.

Slot-value extraction from audio. The agent picks the right tool and the right argument slot, then fills it with a mis-heard value: a digit of a zip code, a character of an order ID, a spelling of a name. Plan and execution are correct; only the perceived value is wrong — a perceptual, not a reasoning, failure, and the largest gap between the channels. The database check is exact, so one confused character is worth the same as no attempt.

Confidently-wrong over-action. The agent performs a state-changing write it was not authorised to perform and ends the episode believing it has helped. We report this qualitatively, as it has no matched text baseline in Table 1: an insistent, emotional caller tone is far more vivid in audio, and the agent concedes to pressure a transcript would have flattened. Where the correct resolution is to decline or escalate, the write corrupts the database and fails the check outright.

Repetitive dead loops. After an error the agent re-issues the identical call rather than changing strategy, until the step budget is exhausted.

Two of these form a single cascade rather than separate problems: a mis-hearing produces a wrong argument, the wrong argument a tool error, the tool error a retry of the same call, and the loop burns the remaining steps until the episode times out at reward zero — which an outcome-only view sees as one undifferentiated failure. Audio agentic capability does not follow from audio understanding.

Table 2 The process reward keeps groups informative. At the base success rates of audio agentic tasks almost every group is 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.

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

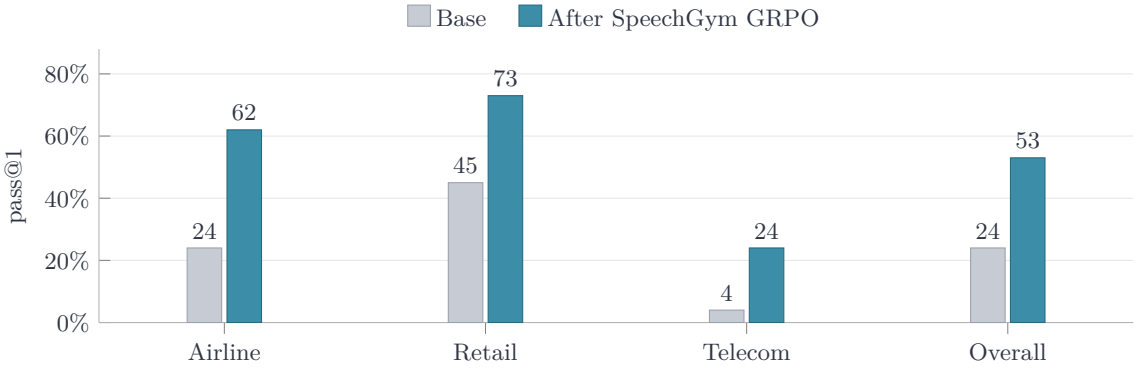

Figure 3 Cross-pipeline transfer to τ-Voice [24], scored by τ-Voice's own database-and-communication check. The SpeechGym-trained agent is run inside τ-Voice's independently implemented harness — its own cascaded user simulator, acoustics and scoring code — with *no further tuning*. Overall pass@1 more than doubles, from 24% to 53%, with gains in all three domains — Airline 24% to 62%, Retail 45% to 73%, Telecom 4% to 24% — and the largest relative gain in Telecom (6×). Both bars in a group come from the same τ-Voice pipeline — same tasks, same user simulator, same acoustics, same grader — at a single attempt; only the agent's weights differ. 95% confidence intervals do not overlap in any domain.

Every link in the cascade has a reward channel, which makes the diagnosis a training plan. Mis-heard values surface as failed tool calls, which the per-turn reward penalises while crediting calls that succeed, so two rollouts that both fail are still ranked by how many calls landed. Over-action is penalised by the outcome check itself, since the unauthorised write fails the database comparison. Dead loops are attacked directly, since each repeated failing call draws its own negative signal instead of being amortised into one terminal zero.

4.3 Training in SpeechGym

The obstacle is gradient starvation, not optimisation. At these base success rates almost every outcome-only group is K identical failures and carries no gradient (Section 3.5): only 16% of groups carry gradient, and the other 84% are discarded after being generated, their rollout compute spent on trajectories that never touch the weights (Table 2). Under per-turn shaping, 99.6% of groups carry gradient, since two rollouts that both fail still differ in how many tool calls they got right.

Outcome-only GRPO still trains, but four fifths of its rollout budget produces no gradient at all. Per-turn shaping recovers that budget in the classical manner of reward shaping [21]; note that our bonus is not potential-based, so it is the terminal criterion — τ^2-bench's unmodified check (Section 3.3) — and not the objective that is preserved unchanged.

All of this shares SpeechGym's own audio, user simulator and rollout machinery with training, so a policy could improve on it by fitting the environment.

4.4 Cross-pipeline transfer to τ-Voice

Transferable skill, or overfitted environment? We take the trained checkpoint, apply *no further tuning*, and run it inside τ-Voice [24], an evaluation-only voice-agent benchmark on the same τ^2-bench task set, implemented

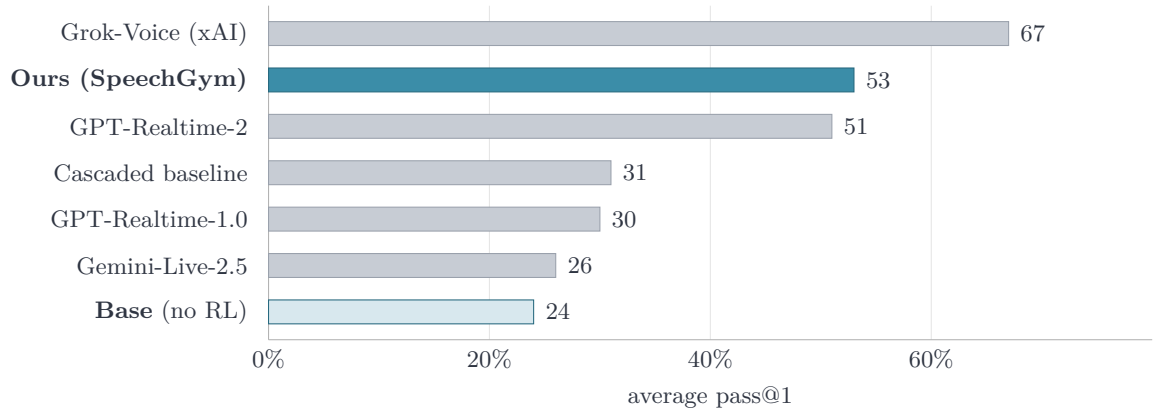

Figure 4 τ-Voice standing. Our trained open-weights 30B model against commercial voice agents on the same benchmark. Training moves the *same* model from last place to second, ahead of the cascaded baseline, both GPT-Realtime versions and Gemini-Live, with only Grok-Voice ranking higher. Scores for the other systems are as reported on the benchmark by their providers; we did not train or re-run them.

Table 3 Behavioural change on τ-Voice, computed from the raw trajectories of both models on the same task set. Every behaviour we tracked moves in the intended direction: wrong writes, i.e. over-action, drop by more than half, dead-loops fall by roughly two thirds, and the agent recovers from a mis-hearing far more often.

Behaviour (% of tasks)	**Base**	**After GRPO**	**Δ**
Wrong writes (over-action), lower better	23%	10%	−13
Dead-loops when stuck, lower better	14%	5%	−9
Recovers after a mis-hearing, higher better	42%	62%	+20

independently.

What is held fixed, and what is not. Holding task content fixed isolates the variable we care about, the audio and interaction pipeline. This is not answer-level memorisation: a τ^2-bench task is a scenario, not a fixed dialogue. The conversation does not exist until it is generated, turn by turn, by a user simulator that reveals information only when asked and reacts to whatever the agent says; there is no transcript to replay. Succeeding means executing the workflow — eliciting the right identifiers, calling the right tools with the right arguments, confirming the outcome — against a user that behaves differently every time.

The pipeline itself is independent throughout. τ-Voice drives its user side with a cascade of commercial APIs (ASR, a user LLM and TTS), not our model-native simulator; its audio comes from different TTS voices over a narrowband, telephony-grade channel, against the clean self-play audio the agent trained on; and it scores with its own harness. Acoustics, user policy and grader all change at once.

Results. Figure 3 shows overall pass@1 rising from 24% to 53%, more than doubling, with gains in every domain and the largest relative gain in Telecom (6×, from a base of 4%). The 95% confidence intervals do not overlap in any domain. The only thing that differs is the agent's weights.

Figure 4 gives context: the same open 30B model moves from last place to second among the systems reported on this benchmark. We keep the claim calibrated — these are systems we did not build, train or re-run, evaluated by their providers' own deployed stacks. What we can say is that one open-weights model, trained locally at no API cost, reaches this position under the same check, and that the change came from RL rather than scale, since the base of the same model sits at the bottom.

Table 4 Higher success at lower cost (τ-Voice). Success more than doubles while turns and tokens both fall, which rules out the two standard ways an RL agent can inflate a success metric: taking more turns until something works, or stalling. For reference the cascaded baseline needs 31.4 turns to reach 31% pass@1.

Metric	Base	After GRPO	Δ
pass@1	24%	53%	more than doubles
Average agent turns	26	24	−8%
Average tokens per task	51,195	48,398	−5%
Turns on the tasks it fixes	23.4	17.5	−25%

4.5 Why it improves: mechanistic checks

A jump from 24% to 53% invites the suspicion that a metric was gamed rather than a task solved. Two checks, either of which could have falsified the result.

(a) Did the diagnosed failures go away? Had the gain come from elsewhere than the cascade of Section 4.2, the diagnosed rates would be roughly unchanged. We annotated the raw τ-Voice trajectories of both models on the same task set for the behaviours that the cascade identifies (Table 3). Unauthorised writes drop from 23% to 10% of tasks, dead loops when stuck from 14% to 5%, and recovery after a mis-hearing rises from 42% to 62%. The failures the environment was built to expose are the failures that move.

(b) Did it buy success with more interaction? Inflating a success rate by spending more of the episode budget — retrying, or stalling until the user concedes — predicts turns and tokens rising with pass@1. They fall (Table 4). Agent turns go from 26 to 24 and tokens per task from 51,195 to 48,398 while pass@1 more than doubles; on the tasks the trained model fixes it is 25% more concise than the base was (23.4 to 17.5 turns). Against the cascaded baseline: 24 turns at 53% pass@1 versus 31.4 turns at 31%. Success rising while compute falls is the opposite of length-based reward hacking.

Strategies we did not design. The annotations also record repair strategies no part of the system specifies: asking the caller to spell a name out, retrying a lookup with a corrected spelling, switching lookup key when the first fails. Nothing in the reward mentions spelling, retries or lookup keys — it scores task completion and tool-call success — so RL found them, consistent with their raising the chance of completing a task over an unreliable channel.

5 Conclusion

SpeechGym is, to our knowledge, the first audio-native agentic environment that both evaluates and trains voice agents end to end in speech, with a local omni-modal user in the loop in place of an API cascade. Using it, we find that the dominant failures of a voice agent are perceptual — slot values misheard from a waveform, and the cascade of failed calls and repetition loops that follows — rather than reasoning deficits, so that audio comprehension and audio *agency* are distinct capabilities; that GRPO with a per-turn process reward closes much of the resulting gap; and that the skills so acquired transfer without further tuning to an independently implemented benchmark, more than doubling pass@1 there from 24% to 53%. The broader point is one of framing: the text-to-audio gap has until now been an open measurement, and an environment that closes the loop recasts it as an optimisation problem with an objective, a gradient and a stopping criterion — and because the interface is a standard reset/step/reward loop over text agentic domains, any open omni-modal model drops into it, in the way standardised environments have served other capabilities [2, 6, 7].

Broader Impact

Every task in SpeechGym runs against a synthetic relational database populated with fictional users, so no real personal information is processed in training or evaluation, and the reward is defined entirely by task

completion — the correct final database state and the correct information communicated — with no term rewarding persuasion, pressure or any other manipulation of the simulated caller. More capable voice agents nevertheless carry deployment risks a simulated gym does not address, and responsible deployment requires safeguards outside its scope: content filtering, explicit user consent for recorded or synthesised speech, and reliable escalation to a human. Because the environment tracks policy-relevant behaviour directly, failures such as unauthorised writes become *measurable* quantities that respond to training (Table 3), and measurability is a prerequisite for control.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] Jiajun Fan and Changnan Xiao. Generalized data distribution iteration. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 6103–6184. PMLR, 2022. URL https://proceedings.mlr.press/v162/fan22c.html.

[7] Jiajun Fan, Yuzheng Zhuang, Yuecheng Liu, Jianye Hao, Bin Wang, Jiangcheng Zhu, Hao Wang, and Shu-Tao Xia. Learnable behavior control: Breaking atari human world records via sample-efficient behavior selection. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=FeWvD0L_a4.

[8] 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. *CoRR*, abs/2510.20867, 2025. doi: 10.48550/ARXIV.2510.20867. URL https://doi.org/10.48550/arXiv.2510.20867.

[9] Jiajun Fan, Shuaike Shen, Chaoran Cheng, Yuxin Chen, Chumeng Liang, and Ge Liu. Online reward-weighted fine-tuning of flow matching with wasserstein regularization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=2IoFFexvuw.

[10] Jiajun Fan, Tong Wei, Chaoran Cheng, Yuxin Chen, and Ge Liu. Adaptive divergence regularized policy optimization for fine-tuning generative models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=aXO0xg0ttW.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] 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, editors, *Proceedings of the 29th Symposium on Operating Systems Principles, SOSP 2023, Koblenz, Germany, October 23-26, 2023*, pages 611–626. ACM, 2023. doi: 10.1145/3600006.3613165. URL https://doi.org/10.1145/3600006.3613165.

[16] Ye Li, Huanan Liu, Kangye Ji, Yuan Meng, Jiajun Fan, Yuansong Wang, Shiyu Qin, Chenglei Wu, Shu-Tao Xia, and Zhi Wang. Elegantvla: Learning when to think for efficient vision-language-action models. *CoRR*, abs/2605.29438, 2026. doi: 10.48550/ARXIV.2605.29438. URL https://doi.org/10.48550/arXiv.2605.29438.

[17] Ye Li, Yuan Meng, Zewen Sun, Kangye Ji, Chen Tang, Jiajun Fan, Xinzhu Ma, Shu-Tao Xia, Zhi Wang, and Wenwu Zhu. SP-VLA: A joint model scheduling and token pruning approach for VLA model acceleration. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=RwdGIIjPlC.

[18] Ye Li, Chen Tang, Yuan Meng, Jiajun Fan, Zenghao Chai, Xinzhu Ma, Zhi Wang, and Wenwu Zhu. PRANCE: joint token-optimization and structural channel-pruning for adaptive vit inference. *IEEE Trans. Pattern Anal. Mach. Intell.*, 48(1):283–298, 2026. doi: 10.1109/TPAMI.2025.3605239. URL https://doi.org/10.1109/TPAMI.2025.3605239.

[19] 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.

[20] 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.

[21] 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, editors, *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999), Bled, Slovenia, June 27 - 30, 1999*, pages 278–287. Morgan Kaufmann, 1999.

[22] 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. 2025. URL http://papers.nips.cc/paper_files/paper/2025/hash/97c5b2707228e7e3fb67e4ecc2e0e607-Abstract-Conference.html.

[23] 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.

[24] 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.

[25] 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*, pages 1–7. IEEE, 2025. doi: 10.1109/ASRU65441.2025.11434780. URL https://doi.org/10.1109/ASRU65441.2025.11434780.

[26] 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, editors, *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.

[27] 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.

[28] 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.

[29] Jiashuo Sun, Jimeng Shi, Yixuan Xie, Saizhuo Wang, Jash Rajesh Parekh, Pengcheng Jiang, Zhiyi Shi, Jiajun Fan, Qinglong Zheng, Peiran Li, Shaowen Wang, Ge Liu, and Jiawei Han. Retrieval is cheap, show me the code: Executable multi-hop reasoning for retrieval-augmented generation. *CoRR*, abs/2605.12975, 2026. doi: 10.48550/ARXIV.2605.12975. URL https://doi.org/10.48550/arXiv.2605.12975.

[30] 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.

[31] 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, editors, *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*, pages 4297–4316. Association for Computational Linguistics, 2025. doi: 10.18653/V1/2025.NAACL-LONG.218. URL https://doi.org/10.18653/v1/2025.naacl-long.218.

[32] 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.

[33] 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.

[34] Bangji Yang, Hongbo Ma, Jiajun Fan, and Ge Liu. Batched contextual reinforcement. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=8Oc3Mx754M.

[35] 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.

[36] 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.

[37] Qinglong Zheng, Jiajun Fan, Chaoran Cheng, and Ge Liu. Procedure-aware reinforcement learning for tool-augmented large language models. In *Third Conference on Language Modeling*, 2026. URL https://openreview.net/forum?id=d4wrBuJ4xo.

[38] 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.