AudioScout-R1: RL Based Search Policy Optimization for Audio Evidence Localization

Anonymous ACL submission

Abstract

Long-form audio understanding requires finding sparse evidence in recordings that span tens of minutes to hours. We formulate this as *budgeted agentic evidence search*, where an agent decides how to search, which regions to inspect, and when to submit evidence. We introduce AudioScout, a task-agnostic scaffold for iterative long-audio exploration, and AudioScout-R1, a reinforcement learning (RL)-optimized search agent trained from timestamped evidence supervision. On long-audio question answering and localization benchmarks, AudioScout-R1 improves evidence localization over untrained agents and a strong offline RAG baseline that preprocesses full recordings with specialized audio models. At the 4B scale, it achieves up to 5.2× and 6.3× higher F1 score than this baseline on descriptive and multiple choice QA, respectively, with gains transferring to downstream answer quality. Analysis shows that RL induces a more decisive and targeted search policy, improving submission reliability while producing tighter final evidence spans with fewer irrelevant segments.

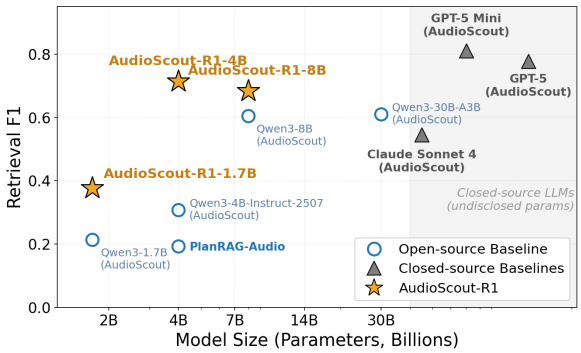

Figure 1: Retrieval performance by policy model size. Retrieval-F_1 is averaged over descriptive QA and MCQ QA across 10, 30, and 60 minute recordings. AudioScout-R1 surpasses substantially larger open-source untrained agents and moves closer to the performance of frontier closed-source agents.

1 Introduction

Many language understanding tasks require identifying small amounts of task-relevant evidence within much larger contexts. In environments such as software repositories, web environments, long documents, and long-form audio, exhaustively inspecting the entire input is often prohibitively expensive. As a result, solving these tasks increasingly depends not only on reasoning over retrieved information, but also on deciding what information to retrieve. Recent work increasingly treats evidence acquisition itself as a sequential decision problem rather than a fixed retrieval stage (Zhang et al., 2025; Song et al., 2025a). Instead of passively consuming full contexts, agents iteratively gather information under limited interaction budgets, deciding what to inspect next, which tools to use, and when sufficient evidence has been acquired. In software engineering, for example, repository-level agents use reinforcement learning to optimize exploration and code localization before downstream reasoning or synthesis (Sutawika et al., 2026; Zhang et al., 2026). These results suggest that search behavior itself can be learned and optimized, especially where evidence is sparse, distributed, and costly to inspect.

Long-form audio presents a particularly challenging instance of sequential evidence search. Recordings such as meetings, lectures, podcasts, and public proceedings can span tens of minutes to hours, while the evidence needed to answer a query may occupy only a few short temporal regions. Unlike text, audio is costly to inspect exhaustively: it is continuous, weakly structured, and highly token-intensive, and relevant evidence may depend not only on lexical content but also on speaker behavior, acoustic events, and paralinguistic cues (Kong et al., 2024; Défossez et al., 2024; Yu et al., 2025). Moreover, evidence in long recordings is often temporally distributed, requiring iterative exploration

rather than one-shot retrieval. A system may need to first locate a relevant discussion and then follow later responses or behavioral changes occurring far away in time. Long-audio understanding is therefore naturally a sequential decision problem under limited inspection budgets.

Existing approaches to long-form audio understanding primarily rely on either direct long-context processing (Xu et al., 2025; Yu et al., 2025) or retrieval-augmented pipelines (Lee et al., 2024). Retrieval-based systems such as PlanRAG-Audio (Someki et al., 2026) mitigate this cost by planning and retrieving query-relevant temporal regions and modalities. However, these pipelines still largely treat retrieval as a fixed preprocessing stage, making them brittle to query reformulation and less effective for questions requiring adaptive multi-step evidence acquisition across distant events. For example, answering *"What proposal was discussed before the speaker changed their position, and what reason did they give afterward?"* requires locating and connecting evidence across multiple temporally separated regions of a recording.

In this work, we propose AudioScout-R1, which formulates long-form audio understanding as budgeted agentic evidence search. Given a recording and question, the agent decides which retrieval channels to use, which regions to inspect, and when enough evidence has been gathered.

Our contributions are:

- We introduce AudioScout, a task-agnostic scaffold for iterative long-audio exploration.
- We present the RL recipe behind AudioScout-R1, optimizing search policies from timestamped evidence supervision.
- At the 4B scale, AudioScout-R1 achieves up to 5.2× and 6.3× higher localization F1 than PlanRAG-Audio on descriptive QA and MCQ QA, respectively, while also improving downstream QA performance.

2 Related Work

2.1 Retrieval-Augmented Audio Understanding

The retrieve-then-generate paradigm introduced by retrieval-augmented generation (Lewis et al., 2021) has also influenced speech and audio understanding. Prior work has explored spoken passage retrieval and audio question answering using both transcript-based and automatic speech recognition (ASR) free formulations (Lin et al., 2024; Sudarsanam and Virtanen, 2023; Abdelnour et al., 2022). In text domains, recent methods move beyond naive top-k retrieval by incorporating planning, self-reflection, and iterative refinement (Asai et al., 2023; Liu et al., 2024; Lee et al., 2024; Verma et al., 2025). These methods show that retrieval can be improved when models reason about what information is needed before or during retrieval. Recent long-audio systems adopt related ideas. For example, PlanRAG-Audio (Someki et al., 2026) combine preprocessing and planning-based retrieval for efficient long-context audio understanding. Our work instead studies long-audio understanding as an interactive search problem, where the policy decides which retrieval channel to use, which segment to inspect, and when to stop is itself optimized.

2.2 Tool-Using and Agentic Systems

Tool-using language agents extend language models from passive generation to interactive information gathering (Yao et al., 2022). Such agents have been applied to domains including software engineering, web navigation, and multimodal reasoning (Wang et al., 2025; Song et al., 2025b; Maben et al., 2025). Recent work also explores reinforcement learning for multimodal and audio agents (Zhu et al., 2026; Li et al., 2025). Our focus differs from prior agentic systems in that we optimize task-agnostic timestamped evidence localization for long-audio under inspection budgets rather than downstream task interaction success.

2.3 RL-Optimized Search Policies

Search agents are especially common in large information spaces such as software repositories, where relevant evidence is sparse and distributed across many files. Recent work on code and repository search agents develops scaffolds for iterative context retrieval, and methods such as CodeScout use reinforcement learning to optimize search behavior directly (Sutawika et al., 2026). These systems show that search is not just a preprocessing step, but a sequential decision problem in which the agent must decide what to inspect next and when enough evidence has been gathered. Our work extends RL-optimized search policies to long-form audio, where evidence is temporally grounded, multimodal, costly to inspect, and far less structured than code. Rather than optimizing for a single

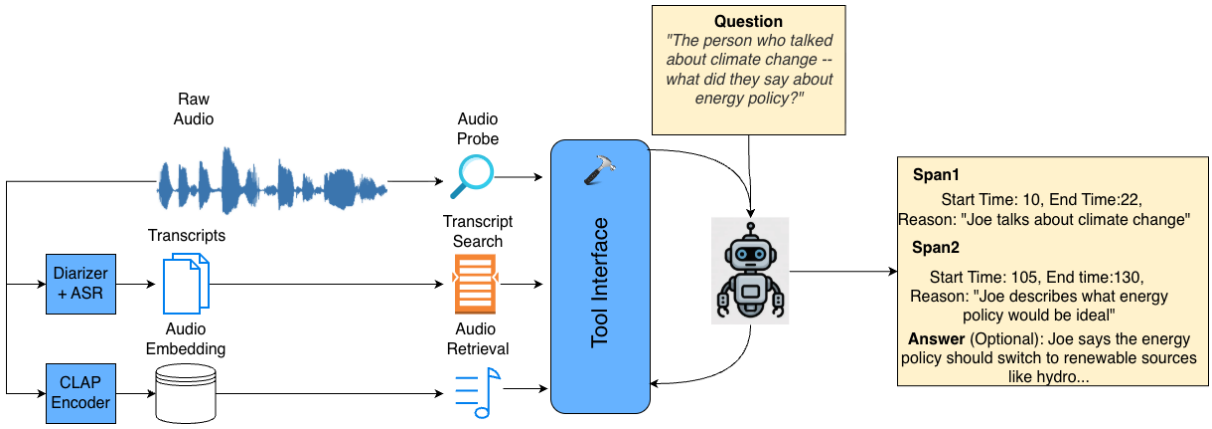

Figure 2: Overview of the AudioScout / AudioScout-R1 framework. Given a long audio recording, metadata, and a question, the agent uses search and inspection tools to gather evidence, then submits snippets and an answer.

downstream task, we train a task-agnostic policy for localizing timestamped evidence under a limited interaction budget.

3 AudioScout-R1: RL-Based Search Policy Optimization for Audio Evidence Localization

We present a complete recipe for optimizing long-audio evidence search with reinforcement learning. Figure 2 illustrates the agent workflow. Each task provides a long audio recording, a diarized transcript, and precomputed CLAP embeddings. Given a question, the agent iteratively uses its tool interface to search the transcript, retrieve acoustic candidates, and inspect selected short audio spans with targeted queries. Once it has gathered sufficient evidence, the agent submits the timestamped spans it predicts are relevant, optionally with a task-level answer. The rest of this section describes the four main components of our approach: data curation, the agent scaffold, reward design, and policy optimization.

3.1 Data Curation and Synthesis

Data format. To train the agent, each datapoint contains the long audio recording, its derived searchable representations, the question, supporting evidence spans, and the ground-truth answer. We represent each example as

$$(q, A, D, C, \mathcal{G}, a),$$

where q is the question, A is the audio recording, D is the diarized transcript, C denotes precomputed CLAP embeddings over fixed audio windows, $\mathcal{G} = \{g_1, \ldots, g_m\}$ is the set of timestamped evidence spans, and a is the reference answer.

This formulation provides timestamped evidence supervision together with the retrieval structures exposed to the agent, enabling direct RL optimization of search policies using retrieval-F_1 as the primary localization signal (Appendix E).

Base corpus. Our primary training corpus is constructed on top of MeetingBank (Hu et al., 2023), a long-form meeting understanding benchmark containing city council recordings paired with ASR transcripts and speaker diarization metadata. Each transcript is converted into a unified schema consisting of timestamped utterances with speaker identifiers and timestamp-aligned text.

Transcript segmentation and event extraction. We merge consecutive utterances from the same speaker into larger speaker turns and construct overlapping transcript windows of approximately 2,000 tokens. This window size is chosen heuristically to capture complete meeting events while remaining within the context budget for downstream LLM processing. Each window is processed by an LLM-based extraction pipeline that identifies discrete meeting events, timestamps, participating speakers, and canonical event labels such as `motion`, `vote`, `presentation`, `discussion`, and `question`. Because overlapping windows can produce duplicate extractions, we merge overlapping event candidates and retain the union of their labels and speakers. Additional extraction details and the full event label set are provided in Appendix A.

Event graph construction. To support multi-hop question generation, we construct an event graph over extracted meeting events. Edges connect events that are temporally or semantically related, including temporal proximity, question-response

pairs, motion-to-vote transitions, and presentation-discussion links. We use this graph to sample linked event pairs for questions that require reasoning across multiple audio regions rather than a single contiguous span.

Question generation. We generate questions with Qwen3-Next-80B-A3B-Instruct[1], conditioned on event transcripts and the event graph. Single-event questions are answerable from one localized segment, while multi-hop questions are generated from edge-linked event pairs and require evidence from both events. Each question includes timestamped supporting spans and multiple-choice answer options. Prompt templates are provided in Appendix B.

Verification and filtering. We verify generated questions with Qwen3-Next-80B-A3B-Instruct, checking correctness, answerability, ambiguity, and support from the timestamped evidence spans. Questions that fail verification are discarded, yielding grounded examples suitable for retrieval-F_1 evaluation. Prompt templates and filtering details are provided in Appendix B.

Needle-in-haystack augmentation. To diversify training beyond meeting-style speech retrieval, we create two synthetic needle-in-haystack datasets by injecting short clips into long MeetingBank recordings. AudioSet (Gemmeke et al., 2017) acoustic needles add non-speech sound events, while IEMOCAP (Busso et al., 2008) emotion needles add emotion-labeled speech segments. The resulting examples cover localization, event or emotion identification, counting, duration, and timing questions. Because insertion times are known, both datasets provide precise timestamp supervision for retrieval-F_1 training.

3.2 Agentic Search Formulation

Given a recording A, associated metadata M, and a natural language question q, the goal is to return a set of timestamped evidence snippets and a final answer. The final prediction is represented as:

$$y = \{s_1, s_2, \ldots, s_k, h\}, \qquad s_i = (t_i^{\text{start}}, t_i^{\text{end}}, r_i),$$

where t_i^{start} and t_i^{end} denote the temporal boundaries of the i-th evidence region, r_i is a short justification for why the segment is relevant, h is the hypothesized answer (which is optional and can be null), and k is the number of submitted spans decided by the agent. An illustration of the agentic framework is provided in Fig 2.

Instead of producing y in a single generation, we formulate retrieval as an iterative tool-use process. At each step, the agent observes the current state and selects a set of tool calls, allowing for parallel exploration of candidate evidence regions. Formally, the trajectory is defined as

$$\tau = (o_1, U_1, o_2, U_2, \ldots, o_T, U_T),$$

where o_t is the observation at step t, $U_t = \{u_t(1), \ldots, u_t^{(m)}\}$ is a set of tool calls sampled based on the preceding trajectory with $m \leq 5$ at timestep t, and T is the maximum search budget. Each set of tool calls U_t is executed in parallel, and the resulting observations are aggregated into the next state o_t. If U_t contains the SUBMIT, the trajectory terminates and the agent generates the final output y based on the τ.

3.3 Agent Tool Interface

The agent interacts with long audio through a modular tool interface. We use the terms *tool* and *action* interchangeably, since each agent action corresponds to a tool call in our formulation. Unlike structured domains such as code, where agents can search over files, functions, and symbols, long audio is continuous and only weakly structured. Our tools therefore impose search structure at two granularities: inexpensive tools propose candidate time regions, while inspection tools reason over selected short spans. A separate submit action terminates the trajectory with timestamped evidence. The interface is task-agnostic by design, exposing general search, inspection, and submission actions rather than benchmark-specific tools. The agent's tool-use strategy is conditioned by a system prompt that specifies tool semantics, search strategies, parallel tool-use constraints, and submission requirements; the full prompt is provided in Appendix C. Tool implementation details are provided in Appendix D.

Transcript search. This tool searches the diarized ASR transcript and returns timestamped transcript spans. It provides an inexpensive first step for locating explicit mentions of entities, names, numbers, topics, or phrases.

Acoustic retrieval. This tool performs text-to-audio retrieval over precomputed CLAP embeddings and returns timestamped candidate audio re-

[1]Model card: https://huggingface.co/Qwen/Qwen3-Next-80B-A3B-Instruct

gions. It complements transcript search for non-speech events and descriptive acoustic queries, but is used mainly intended to propose candidates because embedding retrieval can be noisy.

Audio probe. This tool performs targeted inspection of selected audio windows using an LALM [2]. Given a bounded segment and a query, it returns observations such as detected events, speaker cues, and an answer to the query. It is the most expensive tool, so the agent is encouraged (via system prompt) to use it selectively after cheaper retrieval.

Submit. This tool terminates the trajectory. The agent submits timestamped evidence snippets, short justifications, and an optional answer. This makes stopping part of the policy: the agent must decide when it has gathered enough evidence.

3.4 Reward Function

The primary objective of RL training is to improve evidence localization, measured by retrieval-F_1. We therefore use retrieval-F_1 as the main reward term. However, prior work on RL-trained search agents reports an over-thinking failure mode, where agents take increasingly long trajectories as training progresses, exhaust the `max_step` budget, and collapse into zero-reward no-submit episodes (Sutawika et al., 2026). Following their reward design, we add an auxiliary submission reward that is 1 only when the agent submits at the maximum step and 0 otherwise.

We also include a task-dependent answer reward because long audio may contain multiple valid supporting regions even when only a subset is annotated as gold evidence. This term provides partial credit when the submitted answer is correct despite imperfect overlap with the annotated spans. The full reward is:

$$R = F_1^{\text{retrieval}} + \lambda_{\text{aux}} \mathbf{1}[\text{submitted}] + \lambda_{\text{ans}} \text{Score}_{\text{answer}}. \quad (1)$$

We set $\lambda_{\text{aux}} = 0.1$ and $\lambda_{\text{ans}} = 0.25$, following the scale of secondary rewards used in CodeScout (Sutawika et al., 2026) so that auxiliary terms remain smaller than the primary retrieval reward. Retrieval-F_1 is computed over discretized time windows; details on retrieval-F_1 and task-dependent answer scoring are provided in Appendix E.

[2]We used a model from huggingface hub: https://huggingface.co/Qwen/Qwen3-Omni-30B-A3B-Instruct.

3.5 Reinforcement Learning Setup

We train the search policy using Group Sequence Policy Optimization (GSPO) (Zheng et al., 2025). For each question, the policy samples multiple search trajectories, each receives a scalar reward, and GSPO updates the policy using group-relative comparisons. This is well suited to our setting because rewards are assigned to entire generated sequences rather than individual tokens, matching our trajectory-level objective over tool selection, evidence inspection, and submission timing.

Training is implemented with the AReaL asynchronous RL framework (Fu et al., 2026), which decouples rollout generation from policy optimization. Rollout workers continuously generate trajectories while training workers update on available batches, improving GPU utilization when trajectory lengths and tool latencies vary.

During optimization, loss is applied only to model-generated tokens; tool observations and other environment-provided tokens are masked out. Trajectories that exceed the maximum number of tool-use steps without a valid `submit` action receive zero reward. Following the Dr. GRPO recipe (Liu et al., 2025), we use mean-only advantage normalization without standard-deviation normalization and do not add a KL loss during training.

4 Experimental Setup

4.1 Training Setup

We train Qwen3-family policies at three model scales: Qwen3-1.7B[3], Qwen3-4B-Instruct-2507[4], and Qwen3-8B[5]. The 1.7B and 8B models are reasoning models run in a no-think configuration. All models use the same tool interface and training recipe. We use an 8-turn interaction budget for the 1.7B and 4B models, and a 5-turn budget for the 8B model because early trajectories frequently exceeded the 10,240-token context window.

Training uses the long-audio data described in Section 3.1. We train with the AReaL asynchronous RL framework, using two H200 GPUs: one for vLLM (Kwon et al., 2023) rollout generation and one for FSDP policy optimization. Rollout generation and training run concurrently with up to 32 active episodes, and trajectories from policies more than two update steps stale are discarded.

[3]https://huggingface.co/Qwen/Qwen3-1.7B

[4]https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507

[5]https://huggingface.co/Qwen/Qwen3-8B

We use a GSPO group size of 8, sampling eight trajectories per prompt. We train with a batch size of 8 prompts per update, yielding 64 trajectories per step. Runs are capped at 400 training steps, corresponding to 3,200 unique training prompts. Each trajectory is limited to the model-specific tool-use budget and a maximum context length of 10,240 tokens. We use a constant learning rate of 5×10^{-6}, bfloat16 precision, and gradient clipping at 1.0.

4.2 Evaluation Setup

Datasets. We evaluate on the long-audio benchmark suite derived from PlanRAG-Audio (Someki et al., 2026). The evaluation data is held out from RL training: our training mixture uses MeetingBank and synthetic needle tasks, while the evaluation suite from PlanRAG-Audio. We focus on the subsets most relevant to evidence localization and downstream answer quality: LibriSQA (Zhao et al., 2024) over long-form LibriSpeech (Panayotov et al., 2015) recordings for descriptive QA and MCQ QA, AudioSet-in-VoxPopuli (Wang et al., 2021) needle tasks for sound-event localization, and MSP-Podcast (Busso et al., 2025) for emotion recognition. The diarized transcripts are generated using OWSM-CTC v4 (Peng et al., 2025) and Pyannote (Plaquet and Bredin, 2023; Bredin, 2023).

Each task is evaluated at three audio lengths: 10, 30, and 60 minutes. We use 300 examples per task-length pair, yielding 3,600 examples per system across four tasks.

Evaluation protocol. All agentic systems use the same AudioScout scaffold, prompts, tools, and 8-turn budget, including transcript search, acoustic retrieval, audio metadata, audio probing, and submit tools. They share Qwen3-Omni-30B-A3B-Instruct as the audio probe, isolating search and tool-orchestration policy from perceptual-model differences.

We evaluate two axes: temporal evidence localization and downstream answer quality. Localization is measured using precision, recall, and F1 as described in Appendix E. Downstream quality is measured using task-specific metrics: exact option accuracy for MCQ QA, ROUGE-L for descriptive QA , and Macro-F1 for emotion recognition using the same label set as PlanRAG-Audio.

4.3 Baselines

PlanRAG-Audio. PlanRAG-Audio is our main long-audio RAG baseline (Someki et al., 2026). It performs full-recording offline preprocessing with specialized audio models, including modules for sound event detection and emotion recognition, and retrieves evidence from these precomputed modality-specific streams before answer generation. This makes it a strong comparison point: unlike AudioScout-R1, which uses a general-purpose LALM to inspect only a small number of agent-selected candidate spans under an interaction budget, PlanRAG-Audio has access to dense annotations over the entire recording. We therefore view this as a hard baseline for a task-agnostic online search agent to beat, especially on acoustic and emotion tasks.

AudioScout. We compare against untrained AudioScout agents to isolate the effect of RL optimization. These agents use the same scaffold, tools, prompts, audio probe model, and interaction budget as AudioScout-R1, but the policy LLM is not RL-trained. We evaluate open-weight Qwen3-family policies, including Qwen3-1.7B, Qwen3-4B-Instruct-2507, Qwen3-8B, and Qwen3-30B-A3B-Instruct-2507, as well as closed-source policy LLMs inside the same scaffold as strong references. If an untrained agent does not call the submit tool within the budget, we force a final submission at the last turn for fairness.

Oracle. We report oracle downstream scores to estimate the upper bound given perfect localization. For QA tasks, the oracle reader receives the gold transcript span. For emotion recognition, the oracle receives the gold audio segment. These results help separate errors due to evidence localization from errors due to the reader or audio classifier.

5 Results

We evaluate whether RL improves long-audio evidence localization and whether these gains transfer to downstream answer quality. Table 1 reports retrieval-F_1, with full precision and recall in Appendix F; Table 2 reports downstream metrics.

5.1 AudioScout-R1 improves over untrained AudioScout agents

Across QA evidence localization tasks, RL improves the search policy at all model scales, with the largest and most consistent gains for AudioScout-R1-4B. On descriptive QA, AudioScout-R1-4B achieves 2.1×, 2.2×, and 2.2× the retrieval-F_1 of AudioScout-4B across 10, 30,

Table 1: Retrieval-F_1 on long-audio localization tasks. **Bold** marks the best non-closed result, underlining marks the best closed-source reference, and green highlights PlanRAG-Audio. Full precision/recall results are provided in Appendix F.

System	10m	30m	60m
Descriptive QA			
AudioScout-1.7B	0.389	0.350	0.348
AudioScout-4B	0.339	0.276	0.287
AudioScout-8B	0.631	0.523	0.565
Qwen3-30B-A3B	0.543	0.503	0.544
AudioScout-R1-1.7B	0.424	0.388	0.420
AudioScout-R1-4B	**0.698**	**0.595**	**0.631**
AudioScout-R1-8B	0.642	0.572	0.615
PlanRAG-Audio	0.253	0.168	0.121
GPT-5	0.762	0.732	0.780
GPT-5-mini	0.844	0.770	0.799
Sonnet-4	0.375	0.432	0.498
MCQ QA			
AudioScout-1.7B	0.085	0.049	0.063
AudioScout-4B	0.340	0.274	0.332
AudioScout-8B	0.615	0.658	0.638
Qwen3-30B-A3B	0.711	0.680	0.684
AudioScout-R1-1.7B	0.394	0.305	0.334
AudioScout-R1-4B	**0.777**	**0.769**	**0.811**
AudioScout-R1-8B	0.775	0.753	0.741
PlanRAG-Audio	0.298	0.195	0.129
GPT-5	0.811	0.796	0.788
GPT-5-mini	0.821	0.821	0.804
Sonnet-4	0.663	0.671	0.631
Sound Event Localization			
AudioScout-1.7B	0.480	0.424	0.329
AudioScout-4B	0.546	0.418	0.359
AudioScout-8B	0.561	0.458	0.386
Qwen3-30B-A3B	0.585	0.451	0.370
AudioScout-R1-1.7B	0.496	0.408	0.346
AudioScout-R1-4B	0.550	0.426	0.353
AudioScout-R1-8B	0.521	0.408	0.305
PlanRAG-Audio	**0.776**	**0.711**	**0.654**
GPT-5	0.537	0.439	0.367
GPT-5-mini	0.567	0.437	0.366
Sonnet-4	0.629	0.503	0.436

Table 2: Downstream answer-quality results. Descriptive QA is measured by ROUGE-L, MCQ QA by accuracy (%), and emotion recognition by Macro-F1 (%). **Bold** marks the best non-closed, non-oracle result, underlining marks the best closed-source reference, and green highlights PlanRAG-Audio. Oracle is shown as an upper bound.

System	10m	30m	60m
Descriptive QA ROUGE-L			
AudioScout-1.7B	0.381	0.368	0.367
AudioScout-4B	0.356	0.311	0.317
AudioScout-8B	0.418	0.412	0.429
AudioScout-R1-1.7B	0.389	0.384	0.379
AudioScout-R1-4B	**0.442**	**0.420**	0.437
AudioScout-R1-8B	0.438	0.419	**0.438**
PlanRAG-Audio	0.343	0.351	0.342
GPT-5	0.464	0.480	0.493
GPT-5-mini	0.477	0.492	0.501
Sonnet-4	0.271	0.302	0.308
Oracle	0.503	0.520	0.530
MCQ QA Accuracy (%)			
AudioScout-1.7B	42.0	44.1	43.3
AudioScout-4B	54.7	54.0	56.2
AudioScout-8B	68.5	67.6	67.0
AudioScout-R1-1.7B	50.5	48.1	50.5
AudioScout-R1-4B	71.6	**75.1**	**73.9**
AudioScout-R1-8B	**73.9**	65.9	68.9
PlanRAG-Audio	65.0	54.0	56.0
GPT-5	80.6	76.8	79.9
GPT-5-mini	77.0	76.7	81.7
Sonnet-4	67.2	64.8	65.4
Oracle	78.7	77.3	75.7
Emotion Macro-F1 (%)			
AudioScout-1.7B	22.5	18.7	20.5
AudioScout-4B	25.4	27.3	26.5
AudioScout-8B	27.0	23.8	26.6
AudioScout-R1-1.7B	23.1	17.1	20.6
AudioScout-R1-4B	28.6	**27.4**	**29.4**
AudioScout-R1-8B	**29.0**	27.2	28.4
PlanRAG-Audio	16.9	22.4	19.6
GPT-5	28.0	28.9	27.5
GPT-5-mini	18.4	17.4	19.3
Sonnet-4	23.5	23.0	23.3
Oracle	29.9	22.4	16.2

and 60 minute recordings. On MCQ QA, the gains are 2.3×, 2.8×, and 2.4×. The 1.7B model also improves after RL, with especially large gains on MCQ QA, where AudioScout-R1-1.7B reaches 4.6–6.2× the retrieval-F_1 of AudioScout-1.7B.

RL also improves parameter efficiency. AudioScout-R1-4B outperforms both the larger untrained AudioScout-8B and the much larger untrained Qwen3-30B-A3B agent across all QA settings, reaching up to 1.3× higher retrieval-F_1. Since these agents use the same tools and prompts, the gains indicate that RL improves the search policy itself: how the agent allocates its budget across retrieval, inspection, and final evidence submission.

5.2 AudioScout-R1 outperforms full-preprocessing RAG on QA retrieval

On QA evidence retrieval, AudioScout-R1 substantially outperforms PlanRAG-Audio despite using budgeted online search rather than full-audio preprocessing. At the 4B scale, AudioScout-R1 achieves 2.8×, 3.5×, and 5.2× higher retrieval-F_1

than PlanRAG-Audio on descriptive QA, and 2.6×, 3.9×, and 6.3× higher retrieval-F_1 on MCQ QA across 10, 30, and 60 minute recordings.

PlanRAG-Audio tends to retrieve broad evidence sets with high recall but low precision, whereas AudioScout-R1 returns more compact evidence while maintaining strong recall. Since downstream answer generation must reason over the retrieved context, broad retrieval can introduce noise even when the correct span is present. Our results suggest that learned agentic search is better suited to submitting concise, answer-supporting spans.

5.3 AudioScout-R1 is competitive under selective audio inspection

On sound-event localization, AudioScout-R1 trails PlanRAG-Audio but remains competitive with other agentic systems. AudioScout-R1-4B obtains 0.550, 0.426, and 0.353 retrieval-F_1 across 10, 30, and 60 minutes, compared with 0.776, 0.711, and 0.654 for PlanRAG-Audio. This result suggests that policy optimization does not fully overcome the advantage of dense, full-recording preprocessing for direct acoustic event localization. However, AudioScout-R1 remains competitive with other agentic systems while using the same general search interface as in QA, highlighting the trade-off between task-specific dense preprocessing and task-agnostic selective search.

5.4 Retrieval gains transfer to downstream answer quality

Although we primarily optimize for timestamped evidence localization in a task agnostic manner, the gains transfer to downstream QA. On MCQ QA, AudioScout-R1-4B achieves 71.6%, 75.1%, and 73.9% accuracy across 10, 30, and 60 minutes, outperforming PlanRAG-Audio at 65.0%, 54.0%, and 56.0%. On descriptive QA, AudioScout-R1-4B also improves ROUGE-L over PlanRAG-Audio, reaching 0.442, 0.420, and 0.437 versus 0.343, 0.351, and 0.342.

Emotion recognition shows weaker transfer: AudioScout-R1 improves over PlanRAG-Audio across most lengths, but the gains are smaller than in QA. This suggests that fine-grained emotion classification becomes the bottleneck once the relevant segment is identified.

6 Analysis

RL induces a search-then-submit policy. We analyze saved trajectories on descriptive and MCQ QA, covering 1,800 episodes per system across 10, 30, and 60 minute recordings. The clearest change is valid termination: submit rate increases from 56.2% to 96.6% for 1.7B, 11.6% to 95.4% for 4B, and 90.7% to 96.6% for 8B. The effect is largest at 4B, where RL prevents the untrained agent from searching until the turn limit and instead learns a consistent search-then-commit policy.

RL improves tool orchestration. The gains do not come from using fewer tools. Trained agents make slightly more calls per episode, but use them more purposefully: at 4B, tool calls increase from 3.36 to 4.69 while submit rate rises sharply. Trajectory logs show a consistent pattern after RL: transcript search proposes candidate regions, audio probing verifies spans, and submit commits final evidence. Since trained and untrained agents use the same tools and prompts, these changes indicate that RL improves tool orchestration: the agent learns when to search, inspect, and submit.

RL improves listening efficiency. Trained agents verify evidence more efficiently. Even when they maintain or increase audio-probe usage, they send less total audio to the probe: at 4B, mean probed audio drops from 49s to 23s per episode, and at 8B from 61s to 38s. This suggests that RL learns to inspect tighter regions rather than broad uncertain windows. Final submissions also become more compact, with the 8B mean submitted duration falling from 84.8s to 50.1s. Together, these changes show that RL produces a more targeted evidence localizer: the agent submits reliably, probes more selectively, and reduces noisy over-retrieval.

7 Conclusion

We presented AudioScout, a task-agnostic scaffold for long-audio evidence search, and AudioScout-R1, an RL-optimized agent trained from timestamped evidence supervision. By formulating long-audio understanding as budgeted agentic evidence search, AudioScout-R1 learns to retrieve, inspect, and submit concise evidence spans under a fixed interaction budget. Experiments show that RL substantially improves retrieval-F_1 over untrained agents and outperforms a strong full-preprocessing RAG baseline on QA evidence localization, with gains transferring to downstream QA quality. Behavioral analysis further shows that RL induces a more targeted search policy, improving submission behavior and reducing over-retrieval.

The Use of AI Assistant

Parts of this manuscript were edited for clarity and language using an AI-based writing assistant. The authors take full responsibility for the content.

Limitations

Our approach relies on preprocessing components such as ASR, diarization, and CLAP-based audio retrieval. Errors in these components can propagate to the agent, and their preprocessing cost is incurred before interaction, although this cost can be amortized across multiple queries over the same recording. From an ethical and risk perspective, our framework may inherit the risks and biases associated with the pretrained models it relies on.

Although AudioScout-R1 learns a search policy, the underlying tools are still based on pretrained models whose robustness may vary across domains and acoustic conditions. For example, ASR and diarization can degrade under noisy or overlapping speech, while CLAP retrieval may struggle with out-of-domain or fine-grained acoustic events.

Our reward depends on timestamped gold evidence spans, but long recordings may contain multiple valid supporting regions beyond the annotated spans. We partially address this with an answer reward, but retrieval-F_1 can still penalize evidence that is valid but unannotated.

Finally, our evaluation focuses on the PlanRAG-Audio benchmark suite and selected long-audio tasks. Broader evaluation on more diverse domains, noisier audio, multilingual recordings, and real interactive user queries is needed to establish generality.

References

Jerome Abdelnour, Jean Rouat, and Giampiero Salvi. 2022. Naaqa: A neural architecture for acoustic question answering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–12.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2023. Self-rag: Learning to retrieve, generate, and critique through self-reflection. *Preprint*, arXiv:2310.11511.

Hervé Bredin. 2023. pyannote.audio 2.1 speaker diarization pipeline: principle, benchmark, and recipe. In *Proc. INTERSPEECH 2023*.

Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N. Chang, Sungbok Lee, and Shrikanth S. Narayanan. 2008. IEMOCAP: Interactive emotional dyadic motion capture database. *Language Resources and Evaluation*, 42(4):335–359.

Carlos Busso, Reza Lotfian, Kusha Sridhar, Ali N. Salman, Wei-Cheng Lin, Lucas Goncalves, Srinivas Parthasarathy, Abinay Reddy Naini, Seong-Gyun Leem, Luz Martinez-Lucas, Huang-Cheng Chou, and Pravin Mote. 2025. The msp-podcast corpus. *Preprint*, arXiv:2509.09791.

Alexandre Défossez, Laurent Mazaré, Manu Orsini, Amélie Royer, Patrick Pérez, Hervé Jégou, Edouard Grave, and Neil Zeghidour. 2024. Moshi: a speech-text foundation model for real-time dialogue. *Preprint*, arXiv:2410.00037.

Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, Tongkai Yang, Binhang Yuan, and Yi Wu. 2026. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *Preprint*, arXiv:2505.24298.

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

Yebowen Hu, Tim Ganter, Hanieh Deilamsalehy, Franck Dernoncourt, Hassan Foroosh, and Fei Liu. 2023. Meetingbank: A benchmark dataset for meeting summarization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, Toronto, Canada. Association for Computational Linguistics.

Zhifeng Kong, Arushi Goel, Rohan Badlani, Wei Ping, Rafael Valle, and Bryan Catanzaro. 2024. Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities. *Preprint*, arXiv:2402.01831.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Myeonghwa Lee, Seonho An, and {Min Soo} Kim. 2024. Planrag: A plan-then-retrieval augmented generation for generative large language models as decision makers. NAACL 2024. Association for Computational Linguistics (ACL).

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Gang Li, Jizhong Liu, Heinrich Dinkel, Yadong Niu, Junbo Zhang, and Jian Luan. 2025. Reinforcement learning outperforms supervised fine-tuning: A case study on audio question answering. *arXiv preprint arXiv:2503.11197*.

Chyi-Jiunn Lin, Guan-Ting Lin, Yung-Sung Chuang, Wei-Lun Wu, Shang-Wen Li, Abdelrahman Mohamed, Hung-Yi Lee, and Lin-Shan Lee. 2024. Speechdpr: End-to-end spoken passage retrieval for open-domain spoken question answering. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Yanming Liu, Xinyue Peng, Xuhong Zhang, Weihao Liu, Jianwei Yin, Jiannan Cao, and Tianyu Du. 2024. RA-ISF: Learning to answer and understand from retrieval augmentation via iterative self-feedback. In *Findings of the Association for Computational Linguistics: ACL 2024*, Bangkok, Thailand. Association for Computational Linguistics.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding r1-zero-like training: A critical perspective. *Preprint*, arXiv:2503.20783.

Leander Melroy Maben, Gayathri Ganesh Lakshmy, Srijith Radhakrishnan, Siddhant Arora, and Shinji Watanabe. 2025. Aura: Agent for understanding, reasoning, and automated tool use in voice-driven tasks. *Preprint*, arXiv:2506.23049.

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

Yifan Peng, Muhammad Shakeel, Yui Sudo, William Chen, Jinchuan Tian, Chyi-Jiunn Lin, and Shinji Watanabe. 2025. OWSM v4: Improving Open Whisper-Style Speech Models via Data Scaling and Cleaning. In *Interspeech 2025*, pages 2225–2229.

Alexis Plaquet and Hervé Bredin. 2023. Powerset multi-class cross entropy loss for neural speaker diarization. In *Proc. INTERSPEECH 2023*.

Masao Someki, Chien-yu Huang, Siddhant Arora, Samuele Cornell, Markus Müller, Nathan Susanj, Rupak V Swaminathan, Grant P Strimel, Jing Liu, and Shinji Watanabe. 2026. Planrag-audio: Planning and retrieval augmented generation for long-form audio understanding. *Preprint*, arXiv:2605.20414.

Huatong Song, Jinhao Jiang, Wenqing Tian, Zhipeng Chen, Yuhuan Wu, Jiahao Zhao, Yingqian Min, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025a. Smart-searcher: Incentivizing the dynamic knowledge acquisition of llms via reinforcement learning. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, Suzhou, China. Association for Computational Linguistics.

Yueqi Song, Frank Xu, Shuyan Zhou, and Graham Neubig. 2025b. Beyond browsing: API-based web agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, Vienna, Austria. Association for Computational Linguistics.

Parthasaarathy Sudarsanam and Tuomas Virtanen. 2023. Attention-based methods for audio question answering. In *31st European Signal Processing Conference (EUSIPCO)*, pages 750–754, Helsinki, Finland. IEEE.

Lintang Sutawika, Aditya Bharat Soni, Bharath Sriraam R R, Apurva Gandhi, Taha Yassine, Sanidhya Vijayvargiya, Yuchen Li, Xuhui Zhou, Yilin Zhang, Leander Melroy Maben, and Graham Neubig. 2026. Codescout: An effective recipe for reinforcement learning of code search agents. *Preprint*, arXiv:2603.17829.

Prakhar Verma, Sukruta Prakash Midigeshi, Gaurav Sinha, Arno Solin, Nagarajan Natarajan, and Amit Sharma. 2025. Plan*RAG: Efficient test-time planning for retrieval augmented generation. In *Workshop on Reasoning and Planning for Large Language Models*.

Changhan Wang, Morgane Riviere, Ann Lee, Anne Wu, Chaitanya Talnikar, Daniel Haziza, Mary Williamson, Juan Pino, and Emmanuel Dupoux. 2021. VoxPopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 993–1003, Online. Association for Computational Linguistics.

Xingyao Wang, Boxuan Li, Yufan Song, Frank F. Xu, Xiangru Tang, Mingchen Zhuge, Jiayi Pan, Yueqi Song, Bowen Li, Jaskirat Singh, Hoang H. Tran, Fuqiang Li, Ren Ma, Mingzhang Zheng, Bill Qian, Yanjun Shao, Niklas Muennighoff, Yizhe Zhang, Binyuan Hui, and 5 others. 2025. Openhands: An open platform for AI software developers as generalist agents. In *The Thirteenth International Conference on Learning Representations*.

Jin Xu, Zhifang Guo, Hangrui Hu, Yunfei Chu, Xiong Wang, Jinzheng He, Yuxuan Wang, Xian Shi, Ting He, Xinfa Zhu, Yuanjun Lv, Yongqi Wang, Dake Guo, He Wang, Linhan Ma, Pei Zhang, Xinyu Zhang, Hongkun Hao, Zishan Guo, and 19 others. 2025. Qwen3-omni technical report. *Preprint*, arXiv:2509.17765.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*.

Wenyi Yu, Siyin Wang, Xiaoyu Yang, Xianzhao Chen, Xiaohai Tian, Jun Zhang, Guangzhi Sun, Lu Lu, Yuxuan Wang, and Chao Zhang. 2025. Salmonn-omni: A standalone speech llm without codec

injection for full-duplex conversation. *Preprint*, arXiv:2505.17060.

Weinan Zhang, Junwei Liao, Ning Li, Kounianhua Du, and Jianghao Lin. 2025. Agentic information retrieval. *Preprint*, arXiv:2410.09713.

Zhaoxi Zhang, Yitong Duan, Yanzhi Zhang, Yiming Xu, Zhixiang Wang, Kun Liang, Yang Li, Jiahui Liang, Deguo Xia, Jizhou Huang, Jiyan He, and Yunfang Wu. 2026. One tool is enough: Reinforcement learning for repository-level llm agents. *Preprint*, arXiv:2512.20957.

Zihan Zhao, Yiyang Jiang, Heyang Liu, Yu Wang, and Yanfeng Wang. 2024. Librisqa: A novel dataset and framework for spoken question answering with large language models. *IEEE Transactions on Artificial Intelligence*.

Chujie Zheng, Shixuan Liu, Mingze Li, Xiong-Hui Chen, Bowen Yu, Chang Gao, Kai Dang, Yuqiong Liu, Rui Men, An Yang, Jingren Zhou, and Junyang Lin. 2025. Group sequence policy optimization. *arXiv preprint arXiv:2507.18071*.

Yifan Zhu, Xinyu Mu, Tao Feng, Zhonghong Ou, Yuning Gong, and Haoran Luo. 2026. Omnirag-agent: Agentic omnimodal reasoning for low-resource long audio-video question answering. *Preprint*, arXiv:2602.03707.

A Data Curation Details

This appendix provides additional details for the data curation pipeline described in Section 3.1, including event extraction, question generation, verification, needle-in-haystack synthesis, prompt templates, and dataset statistics.

A.1 MeetingBank Question Construction

We construct MeetingBank training questions from the publicly released city-council recordings and ASR transcripts. Each transcript is converted into a canonical schema in which every utterance contains `start`, `end`, `speaker`, and `text` fields, with timestamps in seconds. This time alignment allows every generated question to be paired with timestamped evidence spans and scored using retrieval-F_1.

All LLM-based stages use Qwen3-Next-80B-A3B-Instruct, served locally through vLLM.

Segmentation and event extraction. We first merge consecutive utterances from the same speaker into larger speaker turns and construct overlapping transcript windows of approximately 2,000 tokens. Each window is processed by an LLM to extract discrete meeting events with timestamps, participating speakers, summaries, and canonical event types. The twelve event types are: `agenda_transition`, `presentation`, `public_comment`, `motion`, `vote`, `vote_result`, `proclamation`, `question`, `response`, `discussion`, `adjournment`, `other`. Time bounds are snapped to a three-second grid. Since overlapping windows can produce duplicate extractions, we merge events whose time ranges overlap and whose canonical labels share at least 60% overlap, retaining the union of time bounds and speakers.

Speaker enrichment. For each meeting, we show the LLM short excerpts from each speaker and ask it to infer the speaker name and role when possible. Roles are restricted to `chair`, `councilmember`, `staff`, `presenter`, `public_commenter`, and `unknown`. This enrichment is used only to make later question-generation prompts clearer.

Event graph construction. We construct a heuristic event graph for each meeting. Nodes are extracted events and edges represent relationships used to seed multi-hop question generation. We include five edge types: `temporal`, `same_topic`, `motion_to_vote`, `question_to_response`, and

presentation_to_discussion. Temporal edges connect nearby events, while other edges connect structurally related events such as motions followed by votes or questions followed by responses. The graph is used only for question generation and is not exposed to the agent.

Question generation and verification. Single-event questions are generated from one event transcript and are answerable from a single localized span. Multi-hop questions are generated from pairs of graph-linked events and require evidence from both events. We also generate speaker count questions but did not use it in training or evaluation.

Generated questions are verified by an LLM for correctness, answerability, ambiguity, and evidence support. We retain questions that are correct, answerable, and unambiguous.

A.2 Needle-in-Haystack Synthesis

We construct two synthetic needle-in-haystack datasets by splicing short clips into long MeetingBank recordings. These datasets diversify training beyond meeting-style speech retrieval and provide exact timestamp supervision because insertion times are known.

AudioSet acoustic needles. We sample non-speech AudioSet clips with specific class labels and insert them into MeetingBank recordings. Generic or explicitly speech-related labels are removed. For each injected clip, we generate question families covering event identification, localization, window-conditioned MCQ, counting, duration, and time-until-first-occurrence. Transcript timestamps after each insertion are shifted by the clip duration, and CLAP indices are recomputed over the modified audio.

IEMOCAP emotion needles. We similarly insert IEMOCAP emotion clips into MeetingBank recordings. Each clip has a categorical emotion label and continuous valence/arousal annotations. We generate questions covering categorical emotion, valence, arousal, counting, and aggregate emotional properties. As with AudioSet needles, the exact insertion time gives precise gold spans for retrieval-F_1 training.

A.3 Dataset Statistics

Table 3: MeetingBank verified question counts. Speaker-count questions are excluded from RL training because their gold span is the full audio.

Tag	Train	Val	Test	Total
single_event	4,119	515	530	5,164
multi_hop	1,957	236	245	2,438
speaker_count	222	27	29	278
Total	**6,298**	**778**	**804**	**7,880**

Table 4: AudioSet acoustic needle dataset counts.

Tag	Total
needle_acoustic	828
needle_localize	828
needle_window_mcq	397
needle_count	417
needle_duration	411
needle_time_until	419
single_event carried over	5,125
multi_hop carried over	2,429
speaker_count carried over	276
Total	**11,130**

Table 5: IEMOCAP emotion needle dataset counts.

Tag	Total
emo_valence	1,380
emo_arousal	1,380
emo_categorical	1,380
emo_count	1,380
emo_count_valence	653
emo_overall_valence	245
single_event carried over	5,126
multi_hop carried over	2,426
speaker_count carried over	276
Total	**14,246**

B Data Curation Prompts

B.1 Event Extraction Prompt

System prompt for event extraction

```
You are an expert at analyzing city council meeting transcripts.
You identify discrete events (motions, votes, presentations,
public comments, etc.) with precise time boundaries.
Always respond with valid JSON and nothing else.
```

User prompt template for event extraction

```
Given this transcript window from a city council meeting
({start} to {end}):

{formatted_turns}

Extract all discrete events. For each event, output:
- event_type: one of {event_types}
- label: short canonical name
- speakers: list of speaker IDs involved
- start_sec: start time in seconds
- end_sec: end time in seconds
- summary: 1-2 sentence description

Return a JSON array. If no clear events, return [].
```

Figure 3: Prompt used for MeetingBank event extraction.

B.2 Speaker Enrichment Prompt

B.3 Single-Event Question Generation Prompt

B.4 Multi-Hop Question Generation Prompt

B.5 Question Verification Prompt

System prompt for speaker enrichment

```
You are an expert at identifying speakers in city council meetings.
You infer names and roles from transcript context.
Always respond with valid JSON and nothing else.
```

User prompt template for speaker enrichment

```
Below are transcript excerpts from different speakers in a city
council meeting.
For each speaker ID, infer:
- name: the speaker's name, or "Unknown" if not identifiable
- role: one of {roles}

{speaker_excerpts}

Return a JSON array:
[{"speaker_id": 0, "name": "Jane Smith", "role": "chair"}, ...]
```

Figure 4: Prompt used for speaker enrichment.

System prompt for question generation

```
You are a question generation expert for long-form audio
recordings of city council meetings. You generate accurate
multiple-choice questions based on transcript excerpts. Always
respond with valid JSON and nothing else.
```

User prompt template for single-event question generation

```
Below is a transcript excerpt from a city council meeting.

Event: {event_type} -- "{label}"
Time: {start} to {end}
Speakers: {speaker_names}
Transcript:
{text}

Generate one multiple-choice question (A/B/C/D) that:
- Is answerable ONLY from this transcript
- Has one clearly correct answer
- Has 3 plausible but wrong distractors
- Tests comprehension of specific details, not vague summaries

Return ONLY this JSON:
{"question": "...", "options": {"A": "...", "B": "...",
"C": "...", "D": "..."}, "answer": "A",
"comment": "brief explanation"}
```

Figure 5: Prompt used for single-event question generation.

User prompt template for multi-hop question generation

```
Below are two related transcript excerpts from a city
council meeting.

Event 1: {type1} -- "{label1}" ({start1} to {end1})
Speakers: {speakers1}
{text1}

Event 2: {type2} -- "{label2}" ({start2} to {end2})
Speakers: {speakers2}
{text2}

Relationship: {edge_type}

Generate one multiple-choice question that
REQUIRES information from BOTH events.
The reader must first extract a fact from
one event, then use it to answer about the other.

Return ONLY this JSON:
{"question": "...", "options": {"A": "...", "B": "...",
"C": "...", "D": "..."}, "answer": "A",
"is_multi_hop": true,
"comment": "explain the reasoning chain across both events"}
```

Figure 6: Prompt used for multi-hop question generation.

System prompt for question verification

```
You are a quality assurance expert for educational questions.
You verify that questions are answerable, correct, and unambiguous.
Always respond with valid JSON and nothing else.
```

User prompt template for question verification

```
Question: {question}
Options: {opts}
Claimed answer: {answer}

Supporting transcript:
{gold_span_text}

Verify:
1. Is the claimed answer correct?
2. Is the question answerable from ONLY the supporting transcript?
3. Are ALL supporting spans necessary?
4. If multi-hop: does answering truly require chaining across spans?
5. Is the question unambiguous?

Return ONLY this JSON:
{"correct": true, "answerable": true,
"all_spans_necessary": true,
"multi_hop_confirmed": false,
"unambiguous": true, "issues": ""}
```

Figure 7: Prompt used for question verification and filtering.

C Agent System Prompt

The following system prompt is provided to the policy LLM at the start of each episode. The fields `{max_turns}`, `{audio_id}`, and `{duration}` are filled at runtime. The prompt describes the available tools, encourages parallel tool use, and requires the agent to terminate using the `submit` tool.

Audio search agent system prompt

```
You are an audio search agent. You answer questions about a
long audio recording(typically 30 min - 3 hr) by calling tools
that search, probe, and submit.

# Tools

- transcript_search: grep/regex over the diarized ASR transcript.
Returns matching segments with timestamps + text. Best for
any speech-derivable content.
- clap_search: dense retrieval over CLAP audio embeddings.
Best for non-speech acoustic events. Query format MUST be 1-4
word acoustic labels.
- audio_metadata: zero-cost lookup returning total duration,
number of distinct speakers, per-speaker speaking time, and silence
fraction.
- omni_probe: send an audio window (<=180 s) plus a question to the
Omni multimodal model, which listens to the selected span. Include
MCQ options verbatim when applicable.
- submit: deliver the final answer + supporting time spans.

# Universal rules

- Use transcript_search generously for speech-derived content.
- Use clap_search for non-speech acoustic events.
- Verify candidate windows with omni_probe before submitting when
possible.
- Parallelize tool calls within a turn, up to 5 tool calls per turn.
- Include MCQ options in omni_probe prompts.
- You MUST call submit before your turns end. Plain-text answers
do not count.
- For MCQ questions, submit the option LETTER, not the option text.
- For localization, submit JSON with start and end seconds.

Budget: you have {max_turns} turns.

The audio_id for this episode is: {audio_id}
The recording is approximately {duration} long.
```

Figure 8: System prompt provided to the audio search agent.

D Tool Interface Details

This appendix provides implementation details for the tools described in Section 3.3. All tools operate over the same audio metadata, diarized transcript, and precomputed retrieval indexes associated with each recording.

Transcript search. The transcript search tool operates over the diarized ASR transcript. It supports lexical and regular-expression queries generated by the agent and returns timestamped transcript spans with surrounding context and speaker metadata. This tool is intended for inexpensive retrieval of explicit mentions such as names, entities, numbers, agenda items, or repeated phrases.

Acoustic retrieval. The acoustic retrieval tool performs text-to-audio retrieval over precomputed CLAP embeddings. For acoustic retrieval, we use `laion/larger_clap_general` [6] as the CLAP embedding model for both audio indexing and text-query encoding. The audio is indexed using fixed windows, and the tool compares the agent's text query against these window embeddings. It returns timestamped candidate segments with similarity scores. This tool is used to propose candidate regions for non-speech events or descriptive acoustic queries that may not appear in the transcript.

Audio probe. The audio probe tool uses Qwen3-Omni-30B-A3B-Instruct to inspect a bounded audio window selected by the agent. The tool receives a start time, end time, and query, and returns structured observations about the selected segment, such as detected events, speaker cues, or an answer to the query. To avoid overloading the shared audio-LLM server and to encourage targeted inspection, we cap each probe request at a maximum duration of three minutes.

Submit. The submit tool terminates the trajectory. It records the final predicted evidence spans, short justifications, and optional task-level answer. These submitted spans are used to compute retrieval-F_1, while the answer is used for task-specific downstream evaluation when applicable.

E Reward Details

Retrieval-F_1. Let P and G denote the sets of predicted and gold evidence spans. We discretize both into one-second windows and treat each span set as the set of covered windows. Precision, recall, and retrieval-F_1 are computed as:

$$\text{Precision} = \frac{|P \cap G|}{|P|}, \qquad \text{Recall} = \frac{|P \cap G|}{|G|},$$

$$F_1^{\text{retrieval}} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

This metric rewards evidence that is both precise and complete: low precision indicates over-retrieval, while low recall indicates missed evidence.

Answer reward. The answer reward is a task-dependent score in $[0, 1]$. For multiple-choice questions, it is binary correctness. For localization-style needle tasks, it is computed from the overlap between the submitted and gold temporal regions. This term is included because long recordings may contain multiple answer-supporting regions even when the dataset annotates only a subset of them.

F Detailed Retrieval Results

[6] We use this model from HuggingFace hub: https://huggingface.co/laion/larger_clap_general

Table 6: Full retrieval results on long-audio localization tasks. We report precision (P), recall (R), and retrieval-F_1 on a 10-second grid. AudioScout denotes the untrained agent and AudioScout-R1 denotes the RL-trained agent. **Bold** marks the best non-closed result, underlining marks the best closed-source reference, and green highlights PlanRAG-Audio.

System	10 min			30 min			60 min		
	P	R	F1	P	R	F1	P	R	F1
Descriptive QA									
AudioScout-1.7B	0.357	0.527	0.389	0.315	0.479	0.350	0.311	0.461	0.348
AudioScout-4B	0.325	0.387	0.339	0.267	0.298	0.276	0.277	0.316	0.287
AudioScout-8B	0.606	0.746	0.631	0.502	0.627	0.523	0.550	0.644	0.565
Qwen3-30B-A3B	0.529	0.617	0.543	0.492	0.550	0.503	0.532	0.575	0.544
AudioScout-R1-1.7B	0.391	0.542	0.424	0.361	0.480	0.388	0.387	0.497	0.420
AudioScout-R1-4B	**0.689**	0.724	**0.698**	**0.584**	0.619	**0.595**	**0.620**	0.654	**0.631**
AudioScout-R1-8B	0.626	**0.769**	0.642	0.563	**0.626**	0.572	0.602	**0.662**	0.615
PlanRAG-Audio	0.204	0.587	0.253	0.115	0.582	0.168	0.088	0.556	0.121
GPT-5	0.701	0.908	0.762	0.674	0.882	0.732	0.723	0.916	0.780
GPT-5-mini	0.811	0.922	0.844	0.727	0.880	0.770	0.761	0.894	0.799
Sonnet-4	0.347	0.435	0.375	0.384	0.555	0.432	0.451	0.608	0.498
MCQ QA									
AudioScout-1.7B	0.078	0.110	0.085	0.043	0.071	0.049	0.058	0.076	0.063
AudioScout-4B	0.340	0.343	0.340	0.269	0.287	0.274	0.329	0.340	0.332
AudioScout-8B	0.605	0.711	0.615	0.652	0.688	0.658	0.629	0.674	0.638
Qwen3-30B-A3B	0.701	0.745	0.711	0.671	0.704	0.680	0.673	0.715	0.684
AudioScout-R1-1.7B	0.357	0.494	0.394	0.285	0.360	0.305	0.303	0.402	0.334
AudioScout-R1-4B	**0.769**	0.794	**0.777**	**0.761**	0.786	**0.769**	**0.798**	**0.841**	**0.811**
AudioScout-R1-8B	0.764	**0.813**	0.775	0.742	**0.788**	0.753	0.732	0.761	0.741
PlanRAG-Audio	0.250	0.642	0.298	0.137	0.659	0.195	0.089	0.634	0.129
GPT-5	0.756	0.942	0.811	0.732	0.955	0.796	0.727	0.940	0.788
GPT-5-mini	0.793	0.896	0.821	0.794	0.887	0.821	0.773	0.879	0.804
Sonnet-4	0.625	0.765	0.663	0.633	0.777	0.671	0.595	0.734	0.631
Sound Event Localization									
AudioScout-1.7B	0.552	0.477	0.480	0.486	0.436	0.424	0.378	0.340	0.329
AudioScout-4B	0.443	0.782	0.546	0.341	0.616	0.418	0.282	0.556	0.359
AudioScout-8B	0.523	0.725	0.561	0.410	0.640	0.458	0.336	0.579	0.386
Qwen3-30B-A3B	0.570	0.696	0.585	0.428	0.567	0.451	0.335	0.499	0.370
AudioScout-R1-1.7B	0.580	0.472	0.496	0.479	0.395	0.408	0.410	0.327	0.346
AudioScout-R1-4B	0.667	0.471	0.550	0.513	0.368	0.426	0.420	0.309	0.353
AudioScout-R1-8B	0.612	0.514	0.521	0.454	0.462	0.408	0.337	0.372	0.305
PlanRAG-Audio	**0.729**	**0.903**	**0.776**	**0.675**	**0.892**	**0.711**	**0.624**	**0.857**	**0.654**
GPT-5	0.458	0.849	0.537	0.352	0.795	0.439	0.296	0.703	0.367
GPT-5-mini	0.548	0.765	0.567	0.409	0.726	0.437	0.343	0.642	0.366
Sonnet-4	0.701	0.660	0.629	0.534	0.583	0.503	0.448	0.553	0.436