GTA-RAG: Graph-Trajectory-Augmented Reinforcement Learning for Multi-Turn Retrieval-Augmented Reasoning

Jun Chen[1,2,3] **Yongchao Liu**[3,†] **Pengyu Qiu**[3] **Jiajun Zheng**[3]
Juelu Zhang[3] **Yujie Zeng**[1,2,3] **Qin Zhang**[1,†] **Ziyue Qiao**[2,†] **Xiao Luo**[4]

[1]College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China
[2]School of Computing and Information Technology, Great Bay University, Dongguan, China
[3]Ant Group, Hangzhou, China
[4]Department of Statistics, University of Wisconsin–Madison, Madison, USA
† Corresponding Authors: zyqiao@gbu.edu.cn, yongchao.ly@antgroup.com, qinzhang@szu.edu.cn

Abstract

Retrieval-augmented generation (RAG) enables LLMs to access external knowledge for answering knowledge-intensive questions. For complex multi-hop questions, multi-turn retrieval-augmented reasoning extends RAG into an iterative process that repeatedly searches for and integrates evidence across documents. However, existing reinforcement-learning (RL) approaches for agentic RAG are typically optimized with final-answer rewards, which provide sparse supervision and overlook whether the model actually retrieves the required evidence chain. We present GTA-RAG, a graph-trajectory-augmented RL framework for multi-turn retrieval-augmented reasoning. From an entity–document graph, we sample connected document paths, synthesize multi-hop QA trajectories, and validate them with the deployed retriever to obtain executable trajectory-level supervision. We then optimize the retrieval policy with Group Relative Policy Optimization (GRPO) and a trajectory-guided reward that encourages both accurate answers and acquisition of target evidence documents, followed by answer-reward training on natural QA instances. Experiments on three multi-hop and two simple QA benchmarks show that GTA-RAG consistently outperforms RL-based RAG baselines with both Qwen2.5-3B and Qwen2.5-7B backbones, while substantially improving evidence-chain coverage. Our code is available at `https://github.com/cjcj46262/GTA-RAG`.

1 Introduction

Retrieval-augmented generation (RAG) equips large language models (LLMs) with external evidence and has become a practical paradigm for knowledge-intensive question answering (Karpukhin et al., 2020; Lewis et al., 2020; Izacard and Grave, 2021). In multi-hop question answering, however, producing a correct response is not sufficient: a model must identify intermediate entities, issue successive searches, and integrate evidence distributed across multiple documents (Chen et al., 2026; Xu et al., 2025). A single retrieval step often fails because the document that directly contains the answer is only reachable after resolving one or more latent bridge entities (Qian et al., 2025).

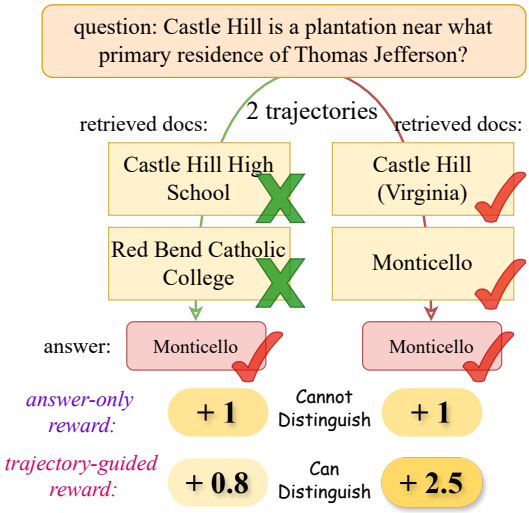

Figure 1: A correct answer can arise from an unsupported retrieval trajectory. Unlike answer-only reward, our trajectory-guided reward distinguishes it from an evidence-grounded trajectory.

Recent RL-based agentic RAG methods have shown that LLMs can learn when and how to search through interaction with a retriever (Yu et al., 2026; Luo et al., 2025b; Tan et al., 2026). Nevertheless, most training signals are dominated by answer-level rewards, such as exact match (EM) on the final prediction. This objective is attractive because it is simple and broadly applicable, but it does not distinguish an evidence-grounded reasoning path from a shortcut that happens to output the correct answer. For example, in one observed case, a model correctly answers that Castle Hill is near *Monticello* despite retrieving documents about an Australian

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 51207–51218
October 24-29, 2026 ©2026 Association for Computational Linguistics

high school and Australian locations rather than the required evidence documents, *Castle Hill (Virginia)* and *Monticello*. Rewarding this rollout solely by final-answer correctness reinforces an unfaithful retrieval behavior.

In this work, we argue that a graph retrieval environment provides a natural source of scalable process supervision. A graph knowledge base already exposes relations among entities and documents; connected document paths therefore encode potential evidence chains (Gutiérrez et al., 2024; Xiang et al., 2025). Rather than requiring manually annotated retrieval trajectories, we sample paths from the graph, prompt a strong generator to formulate multi-hop questions whose answers depend on the sampled evidence, and execute the generated search actions with the same retrieval interface used at training and inference time. Only trajectories whose intended intermediate documents are actually retrievable are retained. This *trajectory validation* step converts graph connectivity into executable, retrieval-grounded training examples.

Based on these data, we propose **G**raph-**T**rajectory-**A**ugmented RAG (GTA-RAG), an RL post-training framework for multi-turn retrieval-augmented reasoning. Our policy learns when and how to issue multi-turn search queries, while each retrieval call returns evidence obtained by combining graph-based retrieval with dense passage retrieval. We train the policy with GRPO (Shao et al., 2024) in two stages. In the first stage, a *trajectory-guided reward* gives intermediate feedback for retrieving new documents on the target evidence chain and provides an additional bonus when a correct answer is supported by a complete retrieved chain. In the second stage, the policy is further optimized with answer-level EM rewards on natural QA training instances, preserving the flexibility needed when explicit trajectories are unavailable.

Experiments on both multi-hop and simple QA benchmarks show that GTA-RAG consistently outperforms prior RL-based RAG systems under both Qwen2.5-3B and Qwen2.5-7B backbones. Retrieval-behavior analysis further demonstrates that our policy more frequently recovers complete evidence chains with fewer unproductive search turns.

Our contributions are threefold:

- We introduce a graph-based data augmentation procedure that automatically synthesizes and validates executable multi-hop retrieval trajectories from an entity–document graph.
- We propose a trajectory-guided GRPO objective that jointly encourages accurate answering and faithful evidence-chain acquisition in multi-turn RAG.
- We demonstrate consistent improvements on both multi-hop and simple QA benchmarks, and provide retrieval-level analyses showing that the gains are associated with better evidence coverage rather than answer-only shortcuts.

2 Related Work

2.1 Graph-Based Retrieval-Augmented Generation

Conventional RAG typically retrieves passages independently according to lexical or dense similarity with the input query (Fan et al., 2024; Arslan et al., 2024; Ren et al., 2026; Peng et al., 2025). This design is effective when the required evidence is locally similar to the question, but it is less suited to multi-hop queries in which the key evidence is connected through intermediate entities or relations. Graph-based RAG methods address this limitation by organizing corpus units into structured representations and retrieving through graph connectivity (Zhang et al., 2025; Guo et al., 2024; He et al., 2024). GraphRAG (Edge et al., 2024) constructs graph abstractions over a text collection to support structured retrieval and aggregation, while RAPTOR (Sarthi et al., 2024) recursively summarizes text into a retrieval hierarchy. HippoRAG (Gutiérrez et al., 2024) and its subsequent extensions use graph-style associations to improve multi-hop evidence discovery and memory-oriented retrieval.

Unlike methods that use graphs only for inference-time retrieval, we exploit graph paths to construct executable multi-document trajectories for supervising intermediate retrieval behavior.

2.2 RL Post-Training for LLM Reasoning

Reinforcement learning has recently been used to improve the reasoning ability of LLMs through outcome-based optimization (Luo et al., 2025b; Yu et al., 2026; Song et al., 2025b; Lu et al., 2025). Group Relative Policy Optimization (GRPO) provides an efficient policy-optimization formulation by estimating relative advantages from multiple sampled outputs for the same prompt, avoiding

a separate value model. In retrieval-augmented settings, recent systems such as Search-R1, R1-Searcher, and RouteRAG train an LLM to interleave reasoning with search actions, demonstrating that RL can substantially improve information-seeking behavior.

However, answer-level rewards provide limited supervision for multi-turn RAG: a model may answer correctly from parametric knowledge or incomplete evidence without learning grounded retrieval behavior. Our trajectory-guided reward addresses this issue by providing intermediate feedback for acquiring documents along the target evidence chain constructed by our augmentation procedure.

3 Method

We propose **G**raph-**T**rajectory-**A**ugmented RAG, a framework that uses a graph-structured corpus both as a retrieval environment and as a source of trajectory-level supervision for reinforcement learning (RL) post-training. Figure 2 provides an overview. The framework consists of three components: (i) constructing an entity–document graph and a graph-augmented retriever; (ii) synthesizing and validating multi-hop retrieval trajectories from graph paths; and (iii) optimizing a multi-turn retrieval policy with a trajectory-guided GRPO objective followed by answer-reward adaptation.

3.1 Problem Formulation

Let $\mathcal{D} = \{d_i\}_{i=1}^{N}$ denotes a corpus of documents and x a question. A retrieval-augmented policy π_θ generates a multi-turn trajectory

$$\tau = (z_1, a_1, o_1, \ldots, z_T, a_T), \tag{1}$$

where z_t is the textual reasoning state available before action a_t, and o_t is the observation returned by the retriever after a search action. At each turn, the policy either issues a retrieval query

$$a_t = \texttt{<search>}\ q_t\ \texttt{</search>}, \tag{2}$$

or terminates with a final prediction

$$a_t = \texttt{<answer>}\ \hat{y}\ \texttt{</answer>}. \tag{3}$$

For a multi-hop retrieval question, let $\mathcal{S}(x) = \{d_1^\star, \ldots, d_H^\star\}$ be its target evidence documents. Our objective is not only to maximize answer correctness, but also to encourage trajectories that retrieve the required evidence documents before answering.

3.2 Graph Knowledge Base and Retrieval

Entity–document graph construction. Starting from each document $d_i \in \mathcal{D}$, we apply an OpenIE model to extract factual triples

$$\mathcal{T}_i = \{(e_h, r, e_t)\}, \tag{4}$$

where e_h and e_t are head and tail entities and r is a textual relation. We construct a heterogeneous graph

$$\mathcal{G} = (\mathcal{V}_D \cup \mathcal{V}_E, \mathcal{E}_{DE} \cup \mathcal{E}_{EE}^r \cup \mathcal{E}_{EE}^s), \tag{5}$$

where $\mathcal{V}_D$ contains document nodes and $\mathcal{V}_E$ contains entity nodes. We add three types of edges: (1) a document–entity edge (d_i, e) if entity e is extracted from d_i; (2) a relation edge (e_h, e_t) for each extracted triple; and (3) a synonymy edge between two entities whose embedding cosine similarity exceeds a threshold δ. The synonymy edges reduce graph fragmentation caused by surface-form variations.

Graph retrieval. Given a search query q, we embed q and all extracted triples and score each triple by cosine similarity:

$$s(q, t) = \cos(\mathbf{h}_q, \mathbf{h}_t), \qquad t \in \mathcal{T}. \tag{6}$$

The top-scoring triples transfer their relevance scores to the participating entity nodes, yielding an initial personalization vector $\mathbf{p}_q$. Starting from these query-relevant entity nodes, we perform personalized PageRank (PPR) over the entity–document graph:

$$\mathbf{r}_q = \alpha \mathbf{p}_q + (1 - \alpha)\mathbf{P}^\top \mathbf{r}_q, \tag{7}$$

where $\mathbf{P}$ is the normalized transition matrix of $\mathcal{G}$ and α is the restart probability. The resulting scores propagate relevance through relational and entity–document connections, allowing the retriever to identify documents that may not be lexically similar to the current query but are structurally relevant to the evolving reasoning chain. We retain the top-K_g document nodes as graph-retrieved evidence:

$$\mathcal{R}_g(q) = \mathrm{TopK}_{d \in \mathcal{V}_D}\big(\mathbf{r}_q(d), K_g\big). \tag{8}$$

Returned retrieval evidence. Although graph retrieval is the central retrieval mechanism in our framework, dense passage retrieval remains useful for recovering documents with strong semantic similarity to the current query. Therefore, for each

query q, we additionally obtain the top-K_d passage results $\mathcal{R}_d(q)$ from a dense retriever and return the deduplicated combination of both sources:

$$\mathcal{R}(q) = \mathrm{Dedup}\left(\mathcal{R}_g(q) \cup \mathcal{R}_d(q)\right). \quad (9)$$

Thus, the policy only needs to formulate a search query, while the retrieval interface consistently provides evidence enhanced by graph reasoning and dense passage matching.

3.3 Multi-Turn Rollout Procedure

The policy operates as an interactive reasoning agent. At the beginning of a rollout, it observes only the input question. At each turn, the policy first generates a reasoning trace z_t and then produces an action a_t, which is either a search query or a final answer. If a search action is issued, the retrieved documents are appended inside `<info>` tags, and the policy continues reasoning based only on the question, its previous reasoning traces, and the evidence retrieved so far. This procedure prevents leakage of future support documents during both synthetic trajectory construction and policy rollout.

Formally, the initial context is $c_1 = x$. After $t-1$ search interactions, the context available to the policy is

$$c_t = [x; z_1; a_1; o_1; \cdots; z_{t-1}; a_{t-1}; o_{t-1}]. \quad (10)$$

At turn t, the policy jointly generates a reasoning trace and an action:

$$(z_t, a_t) \sim \pi_\theta(\cdot \mid c_t). \quad (11)$$

If a_t is a search action with query q_t, the retriever returns $o_t = \mathcal{R}(q_t)$ and the context is updated as

$$c_{t+1} = c_t \oplus z_t \oplus a_t \oplus \texttt{<info>}o_t\texttt{</info>}. \quad (12)$$

The rollout terminates when the policy emits an answer action or reaches the maximum action budget B.

3.4 Multi-hop QA Trajectory Construction

Optimizing a multi-turn retrieval policy with final-answer rewards alone provides sparse supervision: as shown in Figure 1, a policy can answer correctly despite retrieving irrelevant documents, receiving the same reward as an evidence-grounded rollout. To provide intermediate retrieval supervision without manual annotation, we exploit the corpus graph to sample connected document paths, synthesize multi-hop QA trajectories with explicit target-document chains, and validate them using the deployed retriever.

Algorithm 1 Multi-Turn Rollout Procedure

Require: Question x, policy π_θ, retriever $\mathcal{R}$, action budget B
Ensure: Final trajectory τ and predicted answer $\hat{y}$
$c \leftarrow x$; $\tau \leftarrow \varnothing$
for $t = 1$ to B **do**
  Generate $(z_t, a_t) \sim \pi_\theta(\cdot \mid c)$ until `</search>` or `</answer>` is produced
  if a_t matches `<search>` q_t `</search>` **then**
    $o_t \leftarrow \mathcal{R}(q_t)$
    $c \leftarrow c \oplus z_t \oplus a_t \oplus$ `<info>`o_t`</info>`
    $\tau \leftarrow \tau \oplus (z_t, a_t, o_t)$
  else if a_t matches `<answer>` $\hat{y}$ `</answer>` **then**
    $\tau \leftarrow \tau \oplus (z_t, a_t)$
    return $\tau, \hat{y}$
  else
    $c \leftarrow c \oplus z_t \oplus$ "My action is invalid. Let me rethink."
  end if
end for
return $\tau, \hat{y} = \varnothing$

Path sampling. To generate supervision without manual trajectory annotation, we sample document-connected paths from $\mathcal{G}$. Starting at a randomly selected document node, a random walk visits new nodes without replacement until it reaches H document nodes, where H controls the intended hop complexity. We restrict any two consecutive document nodes on a sampled path to be connected through at most two entity nodes. The resulting document sequence

$$P = (d_1^\star, d_2^\star, \ldots, d_H^\star) \quad (13)$$

defines a candidate supporting-document chain.

Question and trajectory synthesis. For each sampled path, a generator LLM receives the documents on P together with distractor documents and is instructed to produce a multi-hop question x, a gold answer $y^\star$, and a turn-by-turn reasoning trajectory whose search queries reveal documents sequentially. Importantly, the generated reasoning at turn t is conditioned only on the question and documents revealed in earlier turns, matching the information constraints of policy rollouts.

Retriever-based validation. Graph connectivity alone does not guarantee that a generated search query can recover its intended target document. We therefore execute every synthetic search query with the same graph-augmented retrieval interface used during policy training and evaluation. A generated example is retained only when, at every search turn, its designated support document appears in the returned evidence. This filtering yields validated

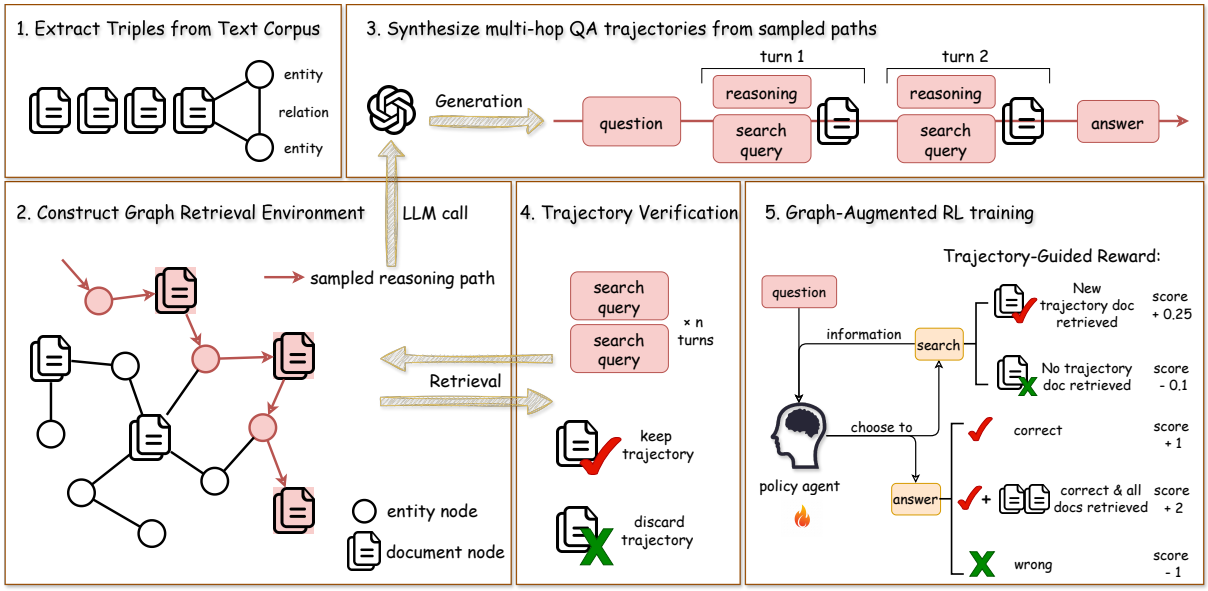

Figure 2: Overview of Graph-Trajectory-Augmented RAG. We first extract entity–relation–entity triples from the corpus and build a graph containing entity nodes, document nodes, relation edges, document–entity edges, and synonymy edges. Connected document paths sampled from this graph are converted into multi-hop question–answer trajectories by a generator model. Each synthetic trajectory is executed through the actual retrieval interface, which returns graph-retrieved evidence combined with dense passage results; trajectories that cannot retrieve their intended support documents are removed. Finally, an agent policy is optimized with GRPO, receiving intermediate trajectory-guided rewards for acquiring new target documents and outcome rewards for producing a correct answer supported by a complete evidence chain.

trajectory supervision

$$\mathcal{D}_{\text{traj}} = \{(x, y^\star, \mathcal{S}(x), \tau^\star)\}, \tag{14}$$

where $\tau^\star$ is executable by the deployed retriever.

3.5 Trajectory-Guided GRPO

The validated trajectories identify the target evidence chain for each synthetic question, enabling supervision beyond final-answer correctness. We therefore design a trajectory-guided reward that favors evidence-grounded rollouts over shortcut answers without the required supporting documents.

Two-stage RL post-training. We train the policy in two successive stages. Stage I uses validated synthetic instances in $\mathcal{D}_{\text{traj}}$ and trajectory-guided rewards, teaching the model to conduct evidence-grounded multi-turn retrieval. Stage II uses natural QA instances with answer-level exact-match reward, improving task adaptation beyond synthetic graph paths while retaining the search behavior acquired in Stage I.

Trajectory-guided reward. For a trajectory τ, let $\mathcal{C}_t$ denote the set of target documents retrieved before turn t, and let

$$\Delta_t = \big(\mathcal{R}(q_t) \cap \mathcal{S}(x)\big) \setminus \mathcal{C}_t \tag{15}$$

be the newly acquired target documents at search turn t. We define the search reward as

$$r_t^{\text{search}} = \begin{cases} +0.25 \cdot |\Delta_t|, & |\Delta_t| > 0, \\ -0.10, & |\Delta_t| = 0. \end{cases} \tag{16}$$

At the terminal answer turn, the answer reward is

$$r^{\text{ans}} = \begin{cases} -1, & \text{EM}(\hat{y}, y^\star) = 0, \\ +1, & \text{EM}(\hat{y}, y^\star) = 1 \wedge \mathcal{C}_T \not\supseteq \mathcal{S}(x), \\ +2, & \text{EM}(\hat{y}, y^\star) = 1 \wedge \mathcal{C}_T \supseteq \mathcal{S}(x). \end{cases} \tag{17}$$

Thus, a correct answer obtains a larger terminal reward when the complete evidence chain has been retrieved, rather than receiving the same score as a shortcut answer. The total Stage I reward is

$$R_{\text{traj}}(\tau) = \sum_{t: a_t \in \mathcal{A}_{\text{search}}} r_t^{\text{search}} + r^{\text{ans}}. \tag{18}$$

In Stage II, we use the broader answer-only reward

$$R_{\text{EM}}(\tau) = \mathbb{I}\left[\text{EM}(\hat{y}, y^\star) = 1\right]. \tag{19}$$

GRPO objective. For each question x, GRPO (Shao et al., 2024; Guo et al., 2025a) samples a group of G trajectories $\{\tau_i\}_{i=1}^G$ and normalizes their rewards within the group:

$$\hat{A}_i = \frac{R(\tau_i) - \text{mean}_{j=1}^G R(\tau_j)}{\text{std}_{j=1}^G R(\tau_j) + \epsilon}. \tag{20}$$

The policy is optimized using the clipped surrogate objective with a KL regularizer to a reference policy π_{ref}:

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}_{x,\{\tau_i\}}\Bigg[\frac{1}{G}\sum_{i=1}^{G}\min\Big(\rho_i(\theta)\hat{A}_i, \text{clip}(\rho_i(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_i\Big) - \beta D_{\text{KL}}\left(\pi_\theta \| \pi_{\text{ref}}\right)\Bigg], \quad (21)$$

where $\rho_i(\theta)$ is the likelihood ratio between the current and behavior policies over the sampled trajectory. In Stage I, $R = R_{\text{traj}}$; in Stage II, $R = R_{\text{EM}}$.

4 Experiments

4.1 Experimental Setup

Benchmarks and evaluation. Following the setting in (Guo et al., 2025b; Sorokin et al., 2025; Luo et al., 2025a), we evaluate on five widely adopted open-domain QA benchmarks. HotpotQA, 2WikiMultiHopQA (2Wiki), and MuSiQue require multi-hop reasoning over distributed evidence, while PopQA and Natural Questions (NQ) mainly evaluate direct factual retrieval. Following prior RAG evaluation protocols, we report exact match (EM) and token-level F1, together with their unweighted averages across datasets.

Baselines. For GPT-4o-mini, we report direct inference and representative training-free RAG systems, including GraphRAG (Edge et al., 2024), LightRAG (Guo et al., 2024), RAPTOR (Sarthi et al., 2024), HippoRAG (Gutiérrez et al., 2024), and HippoRAG 2 (Gutiérrez et al., 2025). For the trainable Qwen2.5-3B and Qwen2.5-7B policies (Hui et al., 2024), we compare with vanilla RAG, training-free RAG Search-o1 (Li et al., 2025), and RL-based retrieval agents such as Search-R1 (Jin et al., 2025), RouteRAG (Guo et al., 2025b), and R1-Searcher (Song et al., 2025a) where available.

Models and retrieval environment. We instantiate the policy with Qwen2.5-3B and Qwen2.5-7B. Our graph construction pipeline extracts OpenIE triples using Llama-3.1-8B. At each search step, PPR-based graph retrieval finds structurally connected documents, while Contriever passage retrieval provides semantically matched evidence; their deduplicated union is returned to the policy. Graph statistics are reported in Appendix A.

Trajectory augmentation and training. We generate candidate multi-hop trajectories from sampled graph paths and retain 2,000 trajectories after retriever-based and rule-based validation. Stage I performs 20 optimization steps with trajectory-guided GRPO to learn evidence-chain-oriented retrieval behavior. Stage II performs another 40 steps on 10,000 HotpotQA instances using EM reward to adapt the policy to natural questions without explicit trajectories. Each step contains 128 prompts with five rollouts per prompt. Detailed trajectory statistics are provided in Appendix A.

4.2 Main Results

Table 1 shows that GTA-RAG consistently improves RL-trained RAG agents across policy scales. With Qwen2.5-3B and Qwen2.5-7B, it obtains average EM/F1 scores of 44.5/53.6 and 50.5/59.3, respectively, outperforming the strongest RL-based alternatives in both backbone groups. The advantage is therefore not tied to a particular model capacity.

More importantly, the improvements are concentrated on multi-hop QA. Averaged over the three multi-hop datasets, GTA-RAG improves over RouteRAG by 4.5 EM and 4.3 F1 points with the 3B backbone, and by 1.7 EM and 1.9 F1 points with the 7B backbone. In contrast, performance on simple QA is competitive but not uniformly superior. This asymmetry matches the design motivation: direct factual questions often require only one semantically matched passage, whereas multi-hop questions depend on locating intermediate evidence before an answer-bearing document can be identified.

The GPT-4o-mini results serve as a reference for training-free RAG performance with a proprietary backbone rather than as a controlled comparison. Notably, after RL post-training, our method with Qwen2.5 becomes competitive with the strongest GPT-4o-mini-based training-free method, HippoRAG 2. This observation highlights that graph-derived trajectories and trajectory-guided rewards can substantially strengthen a trainable open-source policy, enabling it to approach the performance of RAG systems powered by a proprietary model. Within the controlled Qwen comparisons, the consistent multi-hop gains further demonstrate the effectiveness of our training framework. Section 4.4 examines whether these gains are accompanied by improved evidence-chain retrieval behavior.

Table 1: Comparison of retrieval-augmented generation methods on multi-hop QA and simple QA benchmarks. The best results within each backbone group are shown in **bold**.

Method	Multi-hop QA						Simple QA				Average	
	HotpotQA		2Wiki		MuSiQue		PopQA		NQ			
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
GPT-4o-mini												
Direct Inference	28.6	41.0	30.2	36.3	11.2	22.0	16.1	22.7	35.2	52.7	24.3	34.9
Training-free RAG												
GraphRAG	51.4	67.6	45.7	61.0	27.0	42.0	30.7	51.3	38.0	55.5	38.6	55.5
LightRAG	9.9	20.2	2.5	12.1	2.0	9.3	1.9	14.8	2.8	15.4	3.8	14.4
RAPTOR	50.6	64.7	39.7	48.4	27.7	39.2	41.9	55.1	37.8	54.5	39.5	52.4
HippoRAG	46.3	60.0	59.4	67.3	24.0	35.9	**42.5**	**56.2**	37.2	52.5	41.9	54.4
HippoRAG 2	**56.3**	**71.1**	**60.5**	**69.7**	**35.0**	**49.3**	41.7	55.7	**43.4**	**60.0**	**47.4**	**61.2**
Qwen2.5-3B												
Vanilla RAG	29.5	41.8	19.7	27.4	10.3	17.5	30.3	41.6	18.1	31.8	21.6	32.0
Training-free RAG												
HippoRAG 2	31.2	45.0	21.5	33.8	12.2	20.2	29.1	40.1	20.3	33.5	22.9	34.5
Search-o1	18.7	26.3	16.9	20.9	3.9	10.5	17.1	23.8	19.9	29.1	15.3	22.1
RL-based RAG												
Search-R1	45.2	56.9	42.4	50.8	22.2	30.9	45.8	53.3	**46.2**	**54.8**	40.4	49.3
RouteRAG	49.3	60.5	51.8	58.5	24.6	34.2	**46.2**	54.6	40.2	49.2	42.4	51.4
GTA-RAG (Ours)	**52.9**	**63.9**	**57.3**	**64.1**	**29.0**	**38.2**	42.7	51.7	40.6	50.3	**44.5**	**53.6**
Qwen2.5-7B												
Vanilla RAG	29.1	47.5	17.2	33.3	11.4	23.0	26.3	37.8	11.0	30.6	19.0	34.4
Training-free RAG												
HippoRAG 2	27.4	46.1	16.8	34.7	12.3	24.0	27.0	37.9	8.1	27.2	18.3	34.0
Search-o1	13.5	19.1	6.4	7.9	2.9	7.7	4.7	7.2	18.1	27.5	9.1	13.9
RL-based RAG												
R1-Searcher	46.6	56.7	41.7	49.0	29.3	37.6	28.4	41.0	41.6	52.2	37.5	47.3
Search-R1	51.0	62.0	51.8	58.9	32.0	40.8	**51.3**	57.1	**56.8**	**65.3**	48.6	56.8
RouteRAG	55.6	67.5	53.9	61.2	34.6	43.8	48.1	55.3	47.4	56.9	47.9	56.9
GTA-RAG (Ours)	**57.2**	**69.1**	**55.4**	**62.9**	**36.5**	**46.2**	50.4	**57.2**	52.9	61.5	**50.5**	**59.3**

Table 2: Ablation results on Qwen2.5-7B. Multi-hop scores are averaged over HotpotQA, 2Wiki, and MuSiQue. Full-chain denotes complete target-evidence retrieval.

Variant	Multi-hop EM	Multi-hop F1	Full-chain (%)
GTA-RAG	**49.7**	**59.4**	**74.1**
w/o TGR	46.2	56.0	58.7
w/o TV	47.0	56.8	62.4
w/o GR	46.8	56.6	60.9

4.3 Ablation Study

We ablate three components on Qwen2.5-7B: trajectory-guided reward, trajectory validation, and graph retrieval. In Table 2, *TGR* denotes trajectory-guided reward, *TV* denotes trajectory validation, and *GR* denotes graph retrieval; the variant *w/o GR* replaces graph retrieval with dense-only retrieval. All variants retain the same backbone and training scale, allowing the effect of each component to be examined under a controlled setting.

As shown in Table 2, removing TGR produces the largest degradation in full-chain coverage, decreasing it from 74.1% to 58.7%, together with a 3.4-point drop in multi-hop F1. This result supports our motivation that final-answer reward alone is insufficient: a model can obtain a correct answer without retrieving the required evidence chain, whereas trajectory-guided reward explicitly encourages progress toward grounded multi-hop retrieval.

Removing TV also leads to clear decreases in both answer quality and full-chain coverage. This indicates that graph-derived trajectories are useful only when their intended retrieval steps are executable under the deployed retriever; otherwise, noisy or unreachable target paths weaken the supervision signal. Finally, removing GR reduces multi-hop performance and evidence coverage, suggesting that graph-based retrieval helps identify later-hop documents that may be difficult to recover through semantic similarity alone. Over-

Table 3: Retrieval behavior on 1,000 held-out trajectory-based multi-hop questions using Qwen2.5-7B. Higher is better for coverage metrics; lower is better for zero-target turns and average search turns.

Variant	Target-doc ↑ Coverage (%)	Complete-trajectory ↑ Coverage (%)	Zero-target ↓ Turns	Avg. search ↓ Turns
w/o trajectory-guided reward	68.1	54.3	1,094	2.48
w/o trajectory validation	72.4	61.8	978	2.46
GTA-RAG	**82.3**	**74.1**	**755**	**2.42**

all, these results show that graph retrieval supplies structured evidence paths, trajectory validation ensures their reliability, and trajectory-guided reward effectively transfers them into retrieval behavior improvements.

4.4 Retrieval Behavior Analysis

The main results show that GTA-RAG improves end-to-end QA performance, especially on multi-hop benchmarks. We further examine whether these improvements arise from the intended effect of trajectory-guided RL: encouraging the policy to retrieve documents along the target evidence chain rather than producing correct answers from incomplete or irrelevant contexts.

We conduct a controlled retrieval-behavior analysis on 1,000 held-out synthetic multi-hop questions with known target-document trajectories. All variants use the same policy backbone, retrieval environment, training instances, and GRPO configuration, differing only in the removed trajectory-related component. Specifically, *w/o trajectory-guided reward* replaces our process-aware reward with answer correctness alone, while *w/o trajectory validation* trains on synthesized graph-path examples without filtering whether their target documents can be recovered by the deployed retriever.

We report four retrieval-oriented metrics. Let $\mathcal{Q}$ denote the evaluation set, $\mathcal{S}(x)$ the target document set for question x, and $\mathcal{C}(x)$ the set of documents retrieved by the policy before answering. **Target-document coverage** measures the proportion of target evidence documents retrieved across all examples:

$$\mathrm{TargetDocCov} = \frac{\sum_{x\in\mathcal{Q}} |\mathcal{C}(x) \cap \mathcal{S}(x)|}{\sum_{x\in\mathcal{Q}} |\mathcal{S}(x)|}. \quad (22)$$

Complete-trajectory coverage measures the percentage of examples for which all target documents are retrieved before the final answer:

$$\mathrm{CompleteTrajCov} = \frac{1}{|\mathcal{Q}|} \sum_{x\in\mathcal{Q}} \mathbb{I}\left[\mathcal{S}(x) \subseteq \mathcal{C}(x)\right]. \quad (23)$$

We additionally report **zero-target turns**, i.e., search turns that retrieve no previously unseen target document, and **average search turns**, which helps determine whether higher coverage is achieved through more effective retrieval rather than simply more retrieval calls.

Table 3 shows that trajectory-guided RL substantially improves evidence-chain acquisition. Compared with answer-only RL, GTA-RAG increases target-document coverage from 68.1% to 82.3% and complete-trajectory coverage from 54.3% to 74.1%, indicating that it more frequently recovers the entire evidence chain rather than isolated relevant documents.

Removing trajectory validation decreases both coverage metrics, showing that graph-derived trajectories are effective only when executable by the deployed retriever. Moreover, GTA-RAG reduces zero-target turns while using slightly fewer searches, demonstrating that its gains arise from more targeted and evidence-grounded retrieval rather than additional retrieval calls.

5 Conclusion

We presented GTA-RAG, an RL post-training framework that leverages graph-sampled and validated retrieval trajectories to improve multi-turn retrieval-augmented reasoning. By combining graph-augmented retrieval with trajectory-guided GRPO, our method encourages the policy to retrieve evidence along the required reasoning chain rather than relying on answer-only shortcuts. Experiments on multi-hop and simple QA benchmarks demonstrate consistent improvements over strong RL-based RAG baselines, while retrieval-behavior analysis further shows higher target-document and complete-trajectory coverage. These results highlight the potential of graph structure as scalable supervision for training more effective and evidence-grounded RAG agents.

Limitations

Our framework relies on the quality of the constructed entity–document graph and the deployed retriever used for trajectory validation. Extraction noise or missing graph connections may limit the diversity of executable trajectories that can be generated. In addition, our experiments focus on open-domain QA benchmarks; extending GTA-RAG to broader domains and more diverse retrieval environments remains an interesting direction for future work.

Acknowledgments

The work of Ziyue Qiao is supported by the Guangdong Basic and Applied Basic Research Foundation (No. 2025A1515140238, No. 2024A1515140114), the National Natural Science Foundation of China (No. 62676075, No. 62406056), and Ant Group through CCF-Ant Research Fund. The work of Qin Zhang was partially supported by the National Natural Science Foundation of China (No. 62576221) and the Guangdong Provincial Natural Science Foundation (No. 2025A1515010288).

References

Muhammad Arslan, Hussam Ghanem, Saba Munawar, and Christophe Cruz. 2024. A survey on rag with llms. *Procedia computer science*, 246:3781–3790.

Boyu Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. 2026. Pathrag: Pruning graph-based retrieval augmented generation with relational paths. In *Proceedings of the AAAI conference on artificial intelligence*, volume 40, pages 30183–30191.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 6491–6501.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, and 1 others. 2025a. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Yucan Guo, Miao Su, Saiping Guan, Zihao Sun, Xiaolong Jin, Jiafeng Guo, and Xueqi Cheng. 2025b. Routerag: Efficient retrieval-augmented generation from text and graph via reinforcement learning. *arXiv preprint arXiv:2512.09487*.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tian Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*, 2(3).

Bernal J Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems*, 37:59532–59569.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From RAG to memory: Non-parametric continual learning for large language models. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pages 21497–21515. PMLR.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems*, 37:132876–132907.

Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, and 1 others. 2024. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th conference of the european chapter of the association for computational linguistics: main volume*, pages 874–880.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*, pages 6769–6781.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances*

in neural information processing systems, 33:9459–9474.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438.

Miao Lu, Weiwei Sun, Weihua Du, Zhan Ling, Xuesong Yao, Kang Liu, and Jiecao Chen. 2025. Scaling llm multi-turn rl with end-to-end summarization-based context management. *arXiv preprint arXiv:2510.06727*.

Haoran Luo, Guanting Chen, Yandan Zheng, Xiaobao Wu, Yikai Guo, Qika Lin, Yu Feng, Zemin Kuang, Meina Song, Yifan Zhu, and 1 others. 2025a. Hypergraphrag: Retrieval-augmented generation via hypergraph-structured knowledge representation. *Advances in Neural Information Processing Systems*, 38:152206–152234.

Haoran Luo, Haihong E, Guanting Chen, Qika Lin, Yikai Guo, Fangzhi Xu, Zemin Kuang, Meina Song, Xiaobao Wu, Yifan Zhu, and Luu Anh Tuan. 2025b. Graph-r1: Towards agentic graphrag framework via end-to-end reinforcement learning. *arXiv preprint arXiv:2507.21892*.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2025. Graph retrieval-augmented generation: A survey. *ACM Transactions on Information Systems*, 44(2):1–52.

Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Defu Lian, Zhicheng Dou, and Tiejun Huang. 2025. Memorag: Boosting long context processing with global memory-enhanced retrieval augmentation. In *Proceedings of the ACM on Web Conference 2025*, pages 2366–2377.

Yingtao Ren, Ziyi Zhao, Yiwei Fu, Xiao Luo, Yu-Cheng Chang, and Chin-Teng Lin. 2026. When context bites: Detecting rag poisoning via document-level attention collapse. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4080–4086.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations*, volume 2024, pages 32628–32649.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Huatong Song, Jinhao Jiang, Yingqian Min, Jie Chen, Zhipeng Chen, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. 2025a. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Jinyeop Song, Song Wang, Julian Shun, and Yada Zhu. 2025b. Efficient and transferable agentic knowledge graph rag via reinforcement learning. *arXiv preprint arXiv:2509.26383*.

Artyom Sorokin, Nazar Buzun, Alexander Anokhin, Oleg Inozemcev, Egor Vedernikov, Petr Anokhin, Mikhail Burtsev, Trushkov Alexey, Yin Wenshuai, and Evgeny Burnaev. 2025. Q-rag: Long context multi-step retrieval via value-based embedder training. *arXiv preprint arXiv:2511.07328*.

Zhiwen Tan, Jiaming Huang, Qintong Wu, Hongxuan Zhang, Chenyi Zhuang, and Jinjie Gu. 2026. Rag-r1: Incentivizing the search and reasoning capabilities of llms through multi-query parallelism. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 33187–33195.

Zhishang Xiang, Chuanjie Wu, Qinggang Zhang, Shengyuan Chen, Zijin Hong, Xiao Huang, and Jinsong Su. 2025. When to use graphs in rag: A comprehensive analysis for graph retrieval-augmented generation. *arXiv preprint arXiv:2506.05690*.

Tianyang Xu, Haojie Zheng, Chengze Li, Haoxiang Chen, Yixin Liu, Ruoxi Chen, and Lichao Sun. 2025. Noderag: Structuring graph-based rag with heterogeneous nodes. *arXiv preprint arXiv:2504.11544*.

Chuanyue Yu, Kuo Zhao, Yuhan Li, Heng Chang, Mingjian Feng, Xiangzhe Jiang, Yufei Sun, Jia Li, Yuzhi Zhang, Qingyun Sun, and 1 others. 2026. Graphrag-r1: Graph retrieval-augmented generation with process-constrained reinforcement learning. In *Proceedings of the ACM Web Conference 2026*, pages 1398–1409.

Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Hao Chen, Yilin Xiao, Chuang Zhou, Junnan Dong, and 1 others. 2025. A survey of graph retrieval-augmented generation for customized large language models. *arXiv preprint arXiv:2501.13958*.

A Additional Experimental Details

Validated trajectory statistics. Table 4 summarizes the validated synthetic trajectories used for trajectory-guided RL training.

Graph statistics. Table 5 reports the scale of the entity–document graphs constructed for each benchmark corpus. In addition to the evaluation corpora, we construct a larger HotpotQA-10K graph for trajectory augmentation during training.

B Prompt Templates

Table 4: Statistics of the validated synthetic trajectory training set.

Statistic	Value
Validated trajectories	2,000
2-hop trajectories	1,899
3-hop trajectories	93
4-hop trajectories	8
Maximum sampled document nodes per path	10
Stage I steps / prompts per step	20 / 128
Stage II steps / prompts per step	40 / 128
GRPO rollouts per prompt	5

Table 5: Statistics of the entity–document graphs used for graph retrieval and trajectory construction.

Dataset	Document nodes	Entity nodes	Document–entity edges	Relation edges	Synonymy edges	Avg. doc. degree
2WikiMultiHopQA	6,119	44,356	61,219	112,252	207,288	10.00
HotpotQA	9,811	84,890	122,182	227,492	199,187	12.45
HotpotQA-10K	18,300	127,542	206,473	382,750	748,429	11.28
MuSiQue	11,656	88,841	131,962	240,462	170,440	11.32
NQ-REAR	9,633	76,971	132,168	224,324	159,393	13.72
PopQA	8,676	75,906	114,018	199,618	152,737	13.14

Table 6: Prompt template for synthesizing validated multi-hop retrieval trajectories from graph-sampled document paths. Runtime-populated fields are highlighted in blue.

Prompt Template for Graph-Augmented Trajectory Construction

Task Instruction. You are constructing high-quality multi-hop retrieval QA data. You are given a sampled graph path and a list of candidate documents from that path. Your job is to decide whether these documents can support a natural 2-hop, 3-hop, or 4-hop QA example.

Retrieval Behavior. Dense passage retrieval finds documents by semantic similarity between the query and document text. Graph retrieval finds documents through ranked facts, linked entities, and graph reasoning over the structured knowledge graph.

Construction Rules.

1. Use only the provided candidate documents; do not use external knowledge.
2. Construct a natural bridge, comparison, or entity-linking question.
3. The answer must be a short span supported by the selected documents.
4. Use `{min_hops}` to `{max_hops}` search rounds.
5. Each round must target one unique candidate document.
6. The first query must be justified by the question alone.
7. Later queries must be justified by previously retrieved evidence.
8. Do not use unnecessary documents.
9. Output `constructible=false` if no valid sample can be built.
10. Do not reveal the answer in the question or search queries.

Action Format.

`<search> query </search>`

`<answer> answer </answer>`

Runtime Inputs

```
Sampled path:
{sampled_path}

Candidate documents:
{candidate_documents}
```

Required JSON Output Schema

```
{
  "constructible": true or false,
  "reason_if_not_constructible": "...",
  "question": "...",
  "gold_answer": ["..."],
  "hop_count": 2 or 3 or 4,
  "reasoning_process": [
    {
      "round_id": 0,
      "visible_information": "question only",
      "reasoning": "...",
      "action_type": "search",
      "search_query": "...",
      "action": "<search> ... </search>"
    },
    {
      "round_id": 1,
      "visible_information":
          "question + previous support documents",
      "reasoning": "...",
      "action_type": "search",
      "search_query": "...",
      "action": "<search> ... </search>"
    },
    {
      "round_id": 2,
      "visible_information":
          "question + all support documents",
      "reasoning": "...",
      "action_type": "answer",
      "action": "<answer> ... </answer>"
    }
  ],
  "support_docs": [
    {
      "turn_id": 0,
      "search_query": "...",
      "expected_doc_id": "d0",
      "expected_doc_title": "..."
    }
  ]
}
```