# AGENTICRAG-R1: Agentic Reinforcement Learning with Stack Memory for Multi-Step Reasoning, Retrieval and Memorizing

**Anonymous ACL submission**


## Abstract

Retrieval-Augmented Generation (RAG) improves the factuality of large language models (LLMs), yet existing RAG systems often struggle with complex, multi-step reasoning that requires *adaptive retrieval and continuous revision of intermediate contexts*. Recent reinforcement learning (RL)-based agentic RAG methods partially alleviate this issue, but typically rely on *coarse-grained action spaces* and *trajectory-level rewards*, resulting in weak reward assignment and a bias toward short-horizon, stereotyped reasoning template. To address, we propose AGENTICRAG-R1, a RL framework that deeply integrates reasoning, retrieval, and memory via a memory stack and fine-grained action space, supported by hierarchical action-aware rewards and an information-aware trajectory rejection strategy to enable effective long-horizon learning. Experiments across a diverse set of multi-hop, open-domain, and agentic reasoning benchmarks, spanning multiple backbone model sizes, demonstrate that AGENTICRAG-R1 consistently outperforms strong baselines. Moreover, AGENTICRAG-R1 learns more robust, interpretable, and memory-aware reasoning behaviors, highlighting the effect of fine-grained action modeling and information-aware optimization for long-horizon reasoning. Our code is anonymous available at `https://anonymous.4open.science/r/AgenticRAG-R1/`.


## 1 Introduction

**Large Language Models (LLMs)** have achieved impressive performance across a broad range of natural language processing tasks (Kaplan et al., 2020; Yang et al., 2024b; Guo et al., 2025; Vu et al., 2024). However, despite their strengths, LLMs still remain prone to factual errors (Ji et al., 2023a; Cao et al., 2020; Ji et al., 2023b; Jiang et al., 2024b), outdated knowledge (He et al., 2022) and limited domain-specific expertise (Kandpal et al., 2023), making **R**etrieval-**A**ugmented **G**eneration (RAG) (Edge et al., 2024; Asai et al., 2024; Yang et al., 2024c) a promising solution by enabling access to external knowledge for improved accuracy and adaptability.

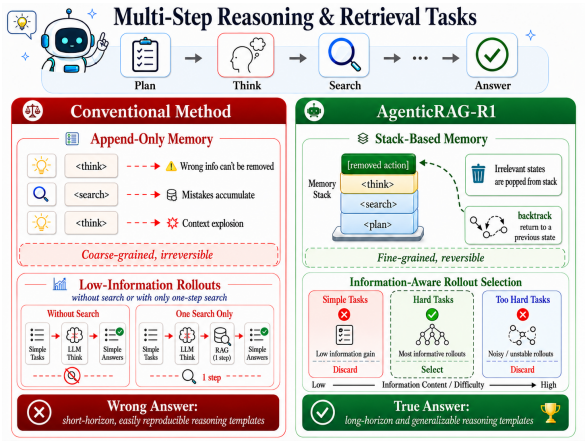


Figure 1: Conventional Agentic vs AGENTICRAG-R1 for multi-step reasoning and retrieval. Conventional methods adopt coarse-grained, append-only memory and shallow retrieval, leading to error accumulation and short-horizon. AGENTICRAG-R1 employs fine-grained, reversible memory actions and information-aware rollout selection, for robust long-horizon reasoning.

In contrast, many real-world decision-making tasks are inherently complicated (Gao et al., 2025), requiring models to decompose problems into multiple interdependent steps rather than relying on one-off reasoning or single-step retrieval (Shao et al., 2023; Press et al., 2022). Such settings demand not only access to external knowledge, but also the ability to dynamically decide *when* and *what* to retrieve as the reasoning process unfolds (Lee et al., 2024; Jeong et al., 2024). Crucially, retrieved information could be continuously integrated into the model's evolving internal state, influencing subsequent reasoning and decision making. This tight coupling between reasoning and retrieval makes their ***deep integration*** a critical

yet underexplored challenge.

Early attempts to integrate reasoning and retrieval primarily relied on manually crafted prompting strategies (Press et al., 2022; Shao et al., 2023; Trivedi et al., 2023; Jiang et al., 2024a; Li et al., 2025), and were later extended through **S**upervised **F**ine-**T**uning (SFT) to better coordinate their interaction (Wang et al., 2025b). While effective in specific settings, these approaches depend heavily on heuristic design and often overfit to **fixed reasoning patterns**, without **intrinsically** improving the model's ability to jointly reason and retrieve (Chu et al., 2025). More recently, **R**einforcement **L**earning (RL) has emerged as a promising paradigm for enhancing complex reasoning, as demonstrated by OpenAI O1 (OpenAI, 2025) and DeepSeek-R1 (Guo et al., 2025). Motivated by this success, several studies have begun to apply RL to retrieval-augmented settings, training models to invoke external retrieval tools during reasoning (Jin et al., 2025; Dong et al., 2025a,b; Zhou et al., 2025). As illustrated in Figure 1, most existing RL-based RAG rely on **coarse-grained action modeling**, treating retrieval as a single or shallow action, which limits fine-grained control over complex tasks and biases policy learning toward **short-horizon**, easily **reproducible reasoning templates** than long-horizon retrieval and reasoning, posing two challenges:

❶ ***Challenge 1. How to expose fine-grained, memory-aware action-making and reward allocation in multi-step reasoning and retrieval.*** Deep integration of reasoning and retrieval requires LLMs not only to acquire **external information**, but also to continuously **manage and revise intermediate content memory** under noisy and uncertain conditions. In multi-step reasoning, both external retrieval and internal inference inevitably introduce noise: retrieved documents may be irrelevant or misleading (Yao et al., 2022; Shinn et al., 2023; Liu et al., 2023), while intermediate reasoning steps may contain logical errors or overconfident assumptions (Jiang et al., 2024a; Xiong et al., 2024; Hao et al., 2023). If such noisy intermediate conclusions are written into context memory without the ability to be inspected, revised, or discarded, they tend to propagate and amplify over steps, effectively corrupting the agent's internal decision state (Ning et al., 2025; Hoscilowicz et al., 2024; Lai et al., 2024). However, most existing RL-based agentic methods adopt a highly **coarse-grained** action formulation, collapsing the entire reasoning and retrieval processes into a single latent and memory-appending `<think>` and `<search>` action template (Dong et al., 2025a,b; Zhou et al., 2025), which are then stored as irreversible content. This limitation is further exacerbated by reward design, where outcome rewards are evaluated only on final correctness, overlooking the contributions of intermediate actions throughout the trajectory (Shao et al., 2024; Jin et al., 2025). Without fine-grained, memory-aware action and reward signals, RL suffers from weak reward assignment and is biased toward **shallow, template-like strategies**.

❷ ***Challenge 2. How to expose reinforcement learning to sufficiently diverse and high-information reasoning and retrieval rollouts.*** Even with fine-grained action spaces and memory-aware reward mechanisms, RL may still fail to learn robust long-horizon reasoning if the training distribution is dominated by **low information trajectories**. Most multi-hop and retrieval-augmented tasks are simple—requiring only a **single retrieval step or shallow reasoning**—so these short-horizon trajectories often yield higher immediate rewards than more complex, long-horizon ones (Gao et al., 2025; Zhou et al., 2025). As a result, RL tends to overfit to **low-information**, trivially rewarding trajectories, rarely exercising actions such as revising, summarizing, or planning, and under-optimizing the long-horizon combinations of cognitive actions (Wang et al., 2025c).

To address the above challenges, we propose AGENTICRAG-R1, a reinforcement learning framework that achieves deep integration of reasoning, retrieval and memorizing through fine-grained action modeling and action-aligned reward mechanisms. ❶ For ***C1***, we design a structured multi-action space together with a stack-based memory which represents the agent's internal state across multi-step reasoning. The memory stack allows intermediate actions to be selectively pushed, revised, or popped, enabling the agent to inspect and correct noisy information accumulated during reasoning and retrieval. Specifically, our framework supports distinct cognitive actions such as `<think>`, `<search>`, and `<plan>`, as well as memory-oriented actions such as `<summary>` and `<backtrack>`, which operate over the memory stack to refine or retract intermediate actions. This action design is further complemented by an action-aware reward attribution mechanism that assigns process rewards to each actions, providing fine-grained reward supervision. ❷ For ***C2***, we intro-

duce a dynamic training trajectories rejection strategy that rejects low-information trajectories and prioritizes high-reward rollouts while up-weighting inputs whose rollouts exhibit meaningful dispersion. Inspired by rejection sampling (Liu et al.; Xiong et al., 2025), we define a **cross-batch, information-awareness metric (Upper Confidence Bound-like)** (Russo and Roy, 2014; Russo and Van Roy, 2013) that evaluates each trajectory based on both its *rollout reward* and the *sample variance*. This approach ensures trajectories eliciting diverse reasoning behaviors and requiring multi-step, long-horizon decision-making receive higher priority during training. In summary, our main contributions are as follows:

- We first formalize the agentic reinforcement learning process within a search-and-memory environment, and introduce a fine-grained, memory-aware action space together with action-aware process-level rewards. This design enables explicit supervision of heterogeneous cognitive actions and effectively mitigates error accumulation in the agent's memory (*C1*).
- We introduce a dynamic, information-aware trajectories rejection strategy that filters low-information trajectories and prioritizes challenging, long-horizon samples with high behavioral variance, ensuring RL effectively explores and optimizes complex, multi-step reasoning strategies (*C2*).
- We evaluate AGENTICRAG-R1 on multiple multi-hop and RAG benchmarks, demonstrating improved reasoning performance, generalization, and applicability to complex downstream tasks, i.e., report generation.

## 2 AGENTICRAG-R1

In this section, we present AGENTICRAG-R1, as illustrated in Figure 2, which consists of two key components: ❶ action-centric training paradigm that supports multi-turn, multi-action rollouts with reasoning, retrieval, and memory operations, and incorporates hierarchical, action-aware reward modeling for fine-grained, token-level reward assignment; ❷ Information-Gain-Aware dynamic rollout selection strategy, improves long-horizon exploration and data efficiency during training. More details of Notation Table, Full Algorithm, Prompt, Proofs, are in Appendix C, B, F, K.

### 2.1 Action Modeling with Hierarchical Rewards

**Memory-aware Multi-Action Modeling.** To overcome ***C1***, we introduce a structured *multi-action mechanism* integrated with the last-in-first-out (LIFO) memory stack that supports push and pop operations. Each stack element corresponds to a discrete and semantically meaningful reasoning unit, serving as the basic execution primitive of the agent. The memory stack enables step-wise **action decomposition** of reasoning trajectories, supports dynamic rollback through stack manipulation, and maintains a reversible and interpretable record of the agent's internal decision process:

- **`<Plan>`** (push): Generate and store a sub-goal, task decomposition, or high-level strategy for long-horizon reasoning.
- **`<Think>`** (push): Record internal reasoning, such as logical deduction or decision-making, to support subsequent inference.
- **`<Search>`** (push): Formulate a query with structured parameters and invoke the retrieval module. This action explicitly determines what and how external knowledge is accessed by external tools.
- **`<Backtrack>`** (pop, then push): Remove one or more top elements upon detecting inconsistencies or errors to restore a prior valid reasoning state for alternative exploration.
- **`<Summary>`** (pop, then push): Compress recent memory entries into a concise representation and push the summary back onto the stack, for context management and information preservation.
- **`<Conclusion>`** (push): Indicate that the current state represents a candidate final answer, concluding the reasoning sequence.

**Hierarchical Reward Modeling** To better model interaction outcomes arising from multi-turn reasoning and multi-action execution—particularly those involving **external retrieval engines and the memory stack**—we introduce a *hierarchical reward modeling* framework. Specifically, we decompose the overall reward signal into two parts: a *outcome reward* and a *process reward*. The outcome reward reflects the quality of the final outcome and is uniformly distributed across valid generation tokens, while the process reward provides fine-grained supervision and is assigned to tokens for corresponding actions.

Formally, for a rollout indexed by $k$, let $y^{(k)} = \left(x_1^{(k)}, \ldots, x_{|y^{(k)}|}^{(k)}\right)$ denote the generated token se-

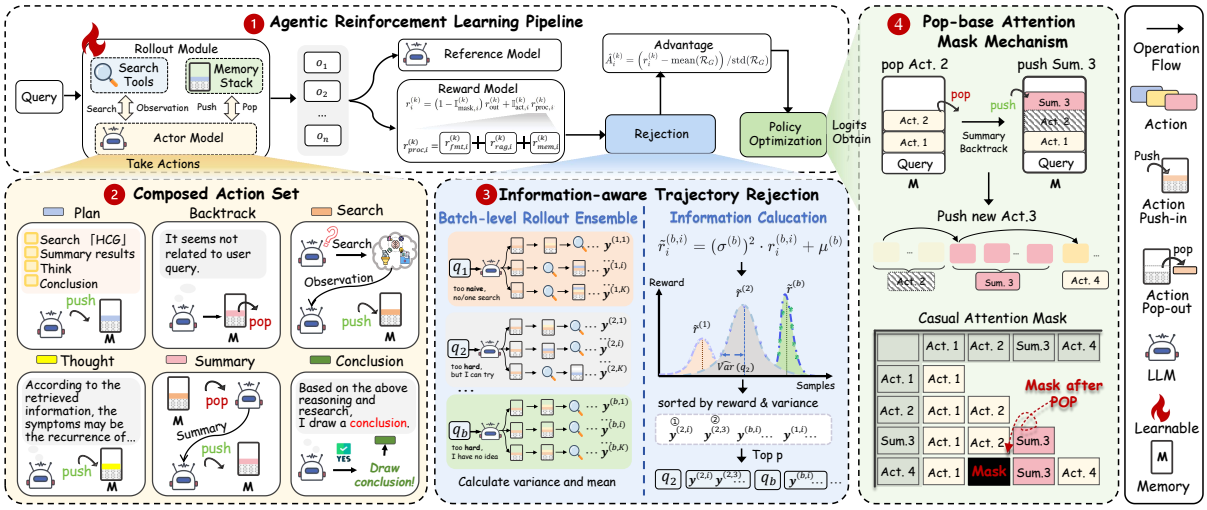


Figure 2: Overall Framework of AGENTICRAG-R1.

quence of the $k$-th trajectory. We define per-token reward as:

$$r_i^{(k)} = \left(1 - \mathbb{I}_{\text{mask},i}^{(k)}\right) r_{\text{out}}^{(k)} + \mathbb{I}_{\text{act},i}^{(k)} r_{\text{proc},i}^{(k)}, \; i \in [1, |y^{(k)}|], \tag{1}$$

where $\mathbb{I}_{\text{mask},i}^{(k)}$ indicates whether token $x_i^{(k)}$ is masked from policy optimization (e.g., retrieved observations) to avoid those false gradient updates, and $\mathbb{I}_{\text{act},i}^{(k)}$ indicates whether the token corresponds to this action invocation. This hierarchical reward formulation allows outcome-level supervision to be shared across the entire reasoning trajectory, while process-level rewards are selectively applied to action tokens, providing fine-grained control over decisions:

- **Outcome Reward** $r_{\text{out}}^{(k)}$**:** evaluates the quality of the final answer produced by the trajectory. We extract the predicted answer from <conclusion> $a_{\text{conclusion}}^{(k)}$ from $y^{(k)}$ and compare it with the ground-truth $a_{\text{gold}}$ via Exact Match (EM) for discrete answers and F1 score for graded answers:

$$r_{\text{out}}^{(k)} = \text{Acc}\left(a_{\text{conclusion}}^{(k)}, a_{\text{gold}}\right) \in [0, 1]. \tag{2}$$

- **Process Rewards** $r_{\text{proc},i}^{(k)}$**:** Beyond the final outcome, we introduce action-conditioned process rewards to supervise intermediate decisions during reasoning. These rewards are applied only at action tokens and encourage syntactically valid actions, effective external retrieval, and robust memory manipulation. Specifically, the process reward is defined as:

$$r_{\text{proc},i}^{(k)} = r_{\text{fmt},i}^{(k)} + r_{\text{rag},i}^{(k)} + r_{\text{mem},i}^{(k)}, \tag{3}$$

which contains the three types:

  - ❶ **Action Format Reward** ($r_{\text{fmt},i}^{(k)}$), enforces syntactic correctness at action level. Let $\mathbb{I}_{\text{syntax}}(a_i)$ be an indicator that equals 1 if action $a_i$ is properly formatted with valid <ACTION>...</ACTION> tags, and 0 otherwise. $r_t^{\text{fmt}}$ is defined as

$$r_{\text{fmt},i}^{(k)} = \alpha_{\text{type}}(a_i), \text{ if } \mathbb{I}_{\text{syntax}}(a_i) = 1, \tag{4}$$

where $\alpha_{\text{type}}(a_i) = 1.0$ for CONCLUSION actions and 0.5 for other valid action types, otherwise 0.

  - ❷ **Search-based Reward** ($r_{\text{rag},i}^{(k)}$). To encourage effective external retrieval, we assign a reward based on semantic relevance of retrieved content. Upon a <Search> action, a large-parameterized *general reward model* (GRM) (Wang et al., 2025a) evaluates the alignment between the issued query, the retrieved evidence, and the current task context, producing a relevance score:

$$r_{\text{rag},i}^{(k)} = \text{GRM}(q_i^s, o_i, q) \in [0, 1], \tag{5}$$

where $q_i^s$ denotes the search query, $o_i$ the retrieved observation, and $q$ the original user input.

  - ❸ **Memory-aware Reward** ($r_{\text{mem},i}^{(k)}$) supervises *useful* memory operations rather than their mere occurrence. For memory-related actions (Backtrack, Summary), we also employ the GRM to assess whether the operation is *reasonable* and *beneficial* given the current reasoning context and memory state. Specifically, we define:

$$r_{\text{mem},i}^{(k)} = \text{GRM}(a_i, M_i, q) \in [0, 1], \tag{6}$$

where $M_i$ denotes the current memory stack.

**Reward Budget and Proportional Normalization.** To prevent reward hacking via excessive action invocation, we impose a per-trajectory budget on process rewards. Let $r_{\text{proc}}^{(k)} = \sum_i r_{\text{proc},i}^{(k)}$ denote the accumulated process reward. Process reward is normalized as:

$$\tilde{r}_{\text{proc},i}^{(k)} = \begin{cases} r_{\text{proc},i}^{(k)}, & R_{\text{proc}}^{(k)} \leq B_{\text{proc}}, \\ \dfrac{B_{\text{proc}}}{R_{\text{proc}}^{(k)}} r_{\text{proc},i}^{(k)}, & R_{\text{proc}}^{(k)} > B_{\text{proc}}, \end{cases} \tag{7}$$

where $B_{\text{proc}}$ is a fixed reward budget. This ensures that when the total process reward exceeds the budget, all process rewards are proportionally scaled to share the same budget, thereby discouraging excessive or redundant action invocation.

### 2.2 Information-Aware Trajectories Rejection

Despite fine-grained, action-level reward, ***C2*** still remains: how to distinguish *informative* trajectories from noisy or redundant ones when multiple stochastic rollouts are generated per input. In practice, rollouts sampled from the same prompt can exhibit substantially different reasoning paths and reward outcomes, while rollouts across different inputs vary widely in their overall learning utility. Naively treating all rollouts equally during optimization can dilute gradient signals and bias training toward **short-sighted, low-effort, or stereotyped templated behaviors**. To address, we propose an *Information-Aware Trajectories Rejection* mechanism that explicitly accounts for both *input-level uncertainty* and *rollout-level quality*. Unlike prior approaches that perform selection independently per input, our method operates *globally* over a mini-batch of rollouts, enabling competition across both inputs and trajectories.

**Batch-level Rollout Ensemble.** At each training step, we sample a mini-batch of inputs $\mathcal{B} = \{q^{(1)}, \ldots, q^{(B)}\}$. For each input $q^{(b)}$, the policy $\pi_\theta$ generates $K$ stochastic rollouts $y^{(b)} = \{y^{(b,1)}, \ldots, y^{(b,K)}\}$, where $y^{(b,i)} \sim \pi_\theta(\cdot \mid q^{(b)})$. All rollouts in the mini-batch form a unified rollout set $\mathcal{Y}_\mathcal{B} = \bigcup_{b=1}^{B} y^{(b)}$ with cardinality $|\mathcal{Y}_\mathcal{B}| = B \times K$. Each rollout $y^{(b,i)}$ is assigned with a scalar reward $r^{(b,i)}$. For each input $q^{(b)}$, we compute the empirical reward mean and variance across its rollouts to estimate expected performance:

$$\mu^{(b)} = \frac{1}{K}\sum_{i=1}^{K} r^{(b,i)}, \; (\sigma^{(b)})^2 = \frac{1}{K}\sum_{i=1}^{K}\left(r^{(b,i)} - \mu^{(b)}\right)^2,$$

Here, the variance $(\sigma^{(b)})^2$ provides a lightweight measure of *input-level informativeness*, reflecting the variability of learning signals across reasoning and retrieval trajectories for a given input.

**Information-Aware Rejection.** Given the batch-level rollout set $\mathcal{Y}_\mathcal{B} = \{y^{(b,i)}\}_{b=1,i=1}^{B,K}$, we assign each rollout an *information-aware score* that jointly accounts for *input-level diversity* and *rollout-level quality*. Concretely, for $i$-th rollout of input $x^{(b)}$, we define it as Upper Confidence Bound (UCB) following (Russo and Roy, 2014; Russo and Van Roy, 2013):

$$\tilde{r}^{(b,i)} = (\sigma^{(b)})^2 \cdot r^{(b,i)} + \mu^{(b)}, \tag{8}$$

where $r^{(b,i)}$ denotes the aggregated scalar reward of the rollout, and $\mu^{(b)}$ and $(\sigma^{(b)})^2$ are the mean and variance of rollout rewards under the same input $x^{(b)}$. This scoring function captures two complementary signals. The variance term $(\sigma^{(b)})^2$ reflects how much the learning signal varies across different reasoning and retrieval trajectories induced by the same input, and thus emphasizes inputs that admit diverse, non-trivial solution paths. The rollout-level reward $r^{(b,i)}$ distinguishes higher-quality trajectories within the same input context. By multiplicatively coupling these terms, the score selectively amplifies **high-reward rollouts when meaningful diversity exists**, while suppressing uninformative or degenerate trajectories. Additive mean term $\mu^{(b)}$ serves as a stabilizing baseline to prevent overly rejection.

Rollout rejection is then performed *globally* over the batch-level rollout set. Specifically, we rank all rollouts $y^{(b,i)} \in \mathcal{Y}_\mathcal{B}$ according to $\tilde{r}^{(b,i)}$ and retain only the top fraction $p \in (0, 1]$. Rollouts not selected are discarded and do not participate in policy optimization. By performing rejection over the joint *batch* $\times$ *rollout* space, this mechanism prioritizes trajectories that are both locally high-quality and globally informative, encouraging exploration of diverse reasoning strategies while improving long-horizon reasoning.

## 3 Experiments

We conduct extensive experiments to evaluate AGENTICRAG-R1 and address the following key research questions:

- **RQ1:** Does AGENTICRAG-R1 consistently outperform strong baselines on in-domain and out-of-domain QA?
- **RQ2:** How do memory-aware actions and trajectory rejection jointly affect overall model performance?
- **RQ3:** What is the impact of retrieval- and memory-based rewards on the quality of learning and reasoning?
- **RQ4:** How sensitive is AGENTICRAG-R1 to the chosen rejection threshold in trajectory filtering?
- **RQ5:** How does AGENTICRAG-R1 effectively handle long-horizon reasoning with larger step budgets?

### 3.1 Experimental Setup

❶ **Training Data.** We train our models and baselines on a curated multi-hop question answering dataset constructed from the training splits of HotpotQA (Yang et al., 2018) and 2WikiMultiHopQA (Ho et al., 2020). To focus on genuinely non-trivial reasoning scenarios, we filter out instances that require no external retrieval or can be solved with only a single, trivial retrieval step. Detailed dataset construction procedures and statistics are provided in Appendix D. ❷ **Evaluation Benchmarks.** We evaluate our method on ten benchmarks spanning three settings: *multi-hop QA* (HotpotQA, 2Wiki, MusiQue (Trivedi et al., 2022), Bamboogle (Press et al., 2022)), *open-domain QA* (Natural Questions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017)), and *agentic benchmarks* (FRAMES (Krishna et al., 2025)).Additional benchmark details are reported in Appendix D. ❸ **Backbone Models.** We adopt **Qwen2.5** (Yang et al., 2024a) as the backbone language model across all experiments, with parameter scales of 1.5B, 3B, and 7B. ❹ **RAG Tools.** We employ a Wikipedia-based search tool built on a dump dated November 1, 2023. ❺ **Baselines.** We compare our approach against a broad range of baselines spanning *No-RAG*, *Naive RAG*, *Agentic RAG*, and *RL-based Agentic* paradigms. These include Base, CoT (Wei et al., 2022a), FS-RAG (Trivedi et al., 2023), FL-RAG (Khandelwal et al., 2020), ReAct (Yao et al., 2022), IR-CoT (Trivedi et al., 2023), TC-RAG (Jiang et al., 2024a), ReSearch (Chen et al., 2025), Search-R1 (Jin et al., 2025), AEPO (Dong et al., 2025a), ARPO (Dong et al., 2025b), and Mem1 (Zhou et al., 2025). Baseline descriptions are provided in Appendix E. ❻ **Evaluation Metrics.** We report the **F1 score** (Yacouby and Axman, 2020) as the primary evaluation metric for all question-answering benchmarks (Appendix G). ❼ **Additional Experiments.** More experiments are listed in Appendix I.

### 3.2 Main Result Analysis

To address **RQ1**, we evaluate AGENTICRAG-R1 on two types of tasks using three backbones (Table 1) and analyze its training behaviors and learning dynamics (Figure 3). Based on these analyses, we highlight the following main observations:

❶ **Strong Performance Compared with Baselines.** Across in-domain and out-of-domain multi-hop QA benchmarks (Table 1), AGENTICRAG-R1 achieves the best or near-best performance across backbone model sizes. On the in-domain datasets, AGENTICRAG-R1 consistently surpasses the strongest baselines, with, for example, absolute gains of **+3.02%** on *2Wiki* and **+6.76%** on *HotpotQA* using the 3B backbone. On out-of-domain benchmarks, AGENTICRAG-R1 exhibits strong generalization, e.g., achieving an absolute improvement of about **+7.2%** on *TriviaQA* with the 1.5B backbone and **+7.1%** on *Bamboogle* with the 7B backbone over the strongest Agentic RAG baselines. Notably, these gains hold across backbone scales, demonstrating the effectiveness and robustness of our reinforcement learning-based optimization in this setting.

❷ **Learning Dynamics of Cognitive Actions and Information-Aware Rejection.** We examine the evolution of cognitive actions during training and the associated learning dynamics, corroborating trends using reward trajectories from Figure 5. Using the Qwen2.5-3B model as a primary example, `<think>` actions steadily increase, reflecting a growing reliance on internal reasoning. Actions related to intermediate reasoning, namely `<summary>` and `<backtrack>`, fluctuate but overall increase; these behaviors help manage memory and remove noise from previous contexts. In contrast, `<search>` remains relatively stable with minor increases. `<conclusion>` actions rise consistently, reducing cases where the model fails to produce a final answer. For the smaller Qwen2.5-1.5B model, similar trends are observed for `<think>`, `<summary>`, `<backtrack>`, and `<plan>`. However, `<search>` frequency is lower, likely due to limited context capacity: performing excessive retrievals can disrupt reasoning, so the model reduces search while still successfully completing reasoning episodes, as reflected

| Method | | In-Domain | | Out-of-Domain | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Paradigm** | **Approach** | **2Wiki** | **HotpotQA** | **Bamboogle** | **FRAMES** | **MusiQue** | **NQ** | **TriviaQA** | **Avg** |
| | | | | | ***Qwen2.5-1.5B*** | | | | |
| No RAG | Base | 21.95 | 20.20 | 5.87 | 9.51 | 8.26 | 11.31 | 32.81 | 15.70 |
| | COT | 16.75 | 17.90 | 19.74 | 8.43 | 7.22 | 9.82 | 25.17 | 15.00 |
| Naive RAG | FS-RAG | 21.37 | 26.55 | 15.31 | 11.40 | 9.50 | 17.74 | 44.73 | 20.94 |
| | FL-RAG | 26.08 | 28.14 | 14.84 | 12.58 | 10.30 | 21.94 | 48.59 | 23.21 |
| Agentic RAG | ReACT | 12.43 | 23.48 | 12.54 | 8.63 | 7.99 | 21.06 | 33.91 | 17.15 |
| | IRCOT | 21.79 | 26.68 | 17.49 | 9.32 | 8.73 | 22.93 | 44.19 | 21.59 |
| | TCRAG | 25.23 | 22.29 | 11.63 | 10.27 | 7.90 | 14.21 | 34.05 | 17.94 |
| RL-based | ReSearch | 25.55 | 30.45 | **22.16** | 9.21 | 9.54 | 29.07 | 44.65 | 24.38 |
| | Search-R1 | 26.51 | 20.66 | 13.80 | 9.65 | 8.36 | 11.93 | 25.76 | 16.67 |
| | AEPO | 17.34 | 14.75 | 11.58 | 8.98 | 7.11 | 9.55 | 23.02 | 13.19 |
| | ARPO | 20.02 | 15.48 | 12.38 | 9.11 | 6.16 | 9.19 | 23.96 | 13.76 |
| | Mem1 | 14.43 | 14.34 | 12.03 | 8.18 | 6.08 | 18.41 | 22.06 | 13.65 |
| | Ours | **29.53** (+3.02) | **30.88** (+0.43) | 20.00 (-2.16) | **13.46** (+0.88) | **16.48** (+6.18) | **31.35** (+2.28) | **51.83** (+3.24) | **27.65** (+3.27) |
| | | | | | ***Qwen2.5-3B*** | | | | |
| No RAG | Base | 23.98 | 24.08 | 9.45 | 8.01 | 9.70 | 14.27 | 38.84 | 18.33 |
| | COT | 18.90 | 23.82 | 20.80 | 7.16 | 10.47 | 16.53 | 39.62 | 19.61 |
| Naive RAG | FS-RAG | 15.47 | 25.85 | 10.48 | 10.42 | 7.64 | 19.84 | 45.38 | 19.30 |
| | FL-RAG | 16.80 | 26.78 | 11.05 | 9.19 | 7.29 | 21.93 | 48.50 | 20.22 |
| Agentic RAG | ReACT | 25.09 | 34.37 | 24.86 | 10.53 | 13.92 | 27.19 | 46.04 | 26.00 |
| | IRCOT | 15.89 | 24.50 | 25.27 | 6.79 | 12.43 | 27.86 | 49.19 | 23.13 |
| | TCRAG | 28.47 | 21.94 | 17.59 | 7.69 | 8.99 | 20.74 | 50.46 | 22.27 |
| RL-based | ReSearch | 27.23 | 33.96 | 15.09 | 10.00 | 9.47 | 34.61 | 53.93 | 26.33 |
| | Search-R1 | 29.90 | 37.24 | 29.90 | 10.76 | 13.53 | 34.73 | 55.08 | 30.16 |
| | AEPO | 23.01 | 28.71 | 22.09 | 12.26 | 11.70 | 26.76 | 47.78 | 24.62 |
| | ARPO | 29.55 | 36.48 | 27.32 | 13.49 | 13.38 | 33.29 | 53.66 | 29.60 |
| | Mem1 | 18.06 | 20.15 | 5.19 | 4.99 | 4.47 | 19.18 | 33.09 | 15.02 |
| | Ours | **32.92** (+3.02) | **44.00** (+6.76) | **31.48** (+1.58) | **16.23** (+2.74) | **16.48** (+2.56) | **37.15** (+2.42) | **56.62** (+1.54) | **33.55** (+3.39) |
| | | | | | ***Qwen2.5-7B*** | | | | |
| No RAG | Base | 25.41 | 26.63 | 17.86 | 12.52 | 12.15 | 19.72 | 49.08 | 23.34 |
| | COT | 23.55 | 29.10 | 37.56 | 17.60 | 14.35 | 22.47 | 49.33 | 27.71 |
| Naive RAG | FS-RAG | 17.71 | 29.21 | 16.86 | 12.52 | 10.74 | 16.82 | 35.02 | 19.84 |
| | FL-RAG | 19.78 | 34.42 | 24.10 | 12.10 | 12.46 | 19.72 | 42.66 | 23.61 |
| Agentic RAG | ReACT | 27.51 | 42.81 | 27.63 | 15.29 | 19.34 | **30.01** | 54.55 | 31.02 |
| | IRCOT | 36.45 | 26.29 | 21.90 | 6.78 | 8.39 | 19.63 | 49.43 | 24.12 |
| | TCRAG | 29.70 | 40.83 | 25.13 | 16.46 | 17.56 | 29.01 | 54.78 | 30.50 |
| RL-based | ReSearch | 30.03 | 30.39 | 30.42 | 15.61 | 12.58 | 23.69 | 48.25 | 27.28 |
| | Search-R1 | 35.03 | 38.89 | 42.04 | 18.01 | 19.08 | 29.59 | 55.91 | 26.94 |
| | AEPO | 19.88 | 13.85 | 13.24 | 7.24 | 5.85 | 9.93 | 17.53 | 12.50 |
| | ARPO | 30.71 | 25.20 | 32.94 | 12.18 | 12.71 | 17.80 | 40.16 | 24.53 |
| | Mem1 | 25.29 | 29.98 | 36.50 | 14.15 | 14.13 | 26.38 | 51.04 | 28.21 |
| | Ours | **38.34** (+1.89) | **45.15** (+2.34) | **49.21** (+7.17) | **19.44** (+1.43) | **22.01** (+2.67) | 23.60 (-6.41) | **58.45** (+2.54) | **36.60** (+5.58) |

Table 1: Performance comparison (%) on multi-hop, retrieval-augmented benchmarks, including *2Wiki*, *HotpotQA*, *Bamboogle*, *FRAMES*, *MusiQue*, *NQ*, and *TriviaQA*. **Bold** indicates the best, and underline indicates the second.

by increased `<conclusion>`.

Additional training statistics track overall model behavior throughout training (Figure 5). Over 0–300 training steps, rewards steadily increase while action entropy shows a general downward trend, suggesting convergence toward more consistent policies. Response length and number of reasoning turns initially rise and then decline, indicating that the model discovers more efficient reasoning and retrieval strategies over time, ultimately producing answers faster and more accurately. Collectively, these analyses reveal how AGENTICRAG-R1 adapts its cognitive actions and information-aware rejection throughout training, balancing exploration and exploitation to optimize multi-hop reasoning.

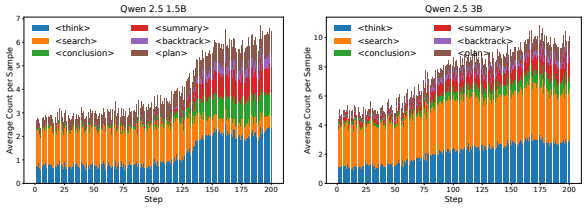


Figure 3: Action-level frequency distributions during RL training across training steps.

### 3.3 Component Analysis

We analyze the effects of key model components, process-level rewards, and rejection thresholds on performance across both in-domain and out-of-domain tasks.

❶ **Model Component Ablation.** We evaluate key components of AGENTICRAG-R1 in Table 2

| Method | | In-Domain | | Out-of-Domain | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **Paradigm** | **Approach** | **2Wiki** | **HotpotQA** | **Bamboogle** | **FRAMES** | **MusiQue** | **NQ** | **TriviaQA** | **Avg** |
| Full Model | AGENTICRAG-R1 | 32.92 | 44.00 | 31.48 | 16.23 | 16.48 | 37.15 | 56.62 | 33.55 |
| Component Ablation | w/o MemAction | 27.54 | 28.77 | 24.60 | 9.83 | 13.86 | 35.10 | 51.91 | 27.37 |
| | w/o Reject | 26.86 | 28.33 | 27.69 | 12.98 | 12.68 | 23.67 | 45.80 | 25.43 |
| | w/o MemAct&Reject | 24.56 | 28.61 | 25.49 | 11.61 | 12.54 | 22.07 | 44.83 | 24.24 |
| Reward Ablation | w/o $r_{\text{rag}}$ | 23.70 | 34.71 | 19.68 | 7.70 | 11.45 | 29.07 | 52.39 | 25.53 |
| | w/o $r_{\text{mem}}$ | 27.49 | 33.78 | 17.14 | 7.41 | 7.63 | 22.00 | 55.29 | 24.39 |
| | w/o $r_{\text{rag}}\&r_{\text{mem}}$ | 22.98 | 26.42 | 15.74 | 5.33 | 6.07 | 18.62 | 44.40 | 19.94 |

Table 2: Ablation analysis of component and reward designs in AGENTICRAG-R1 on Qwen2.5-3B.

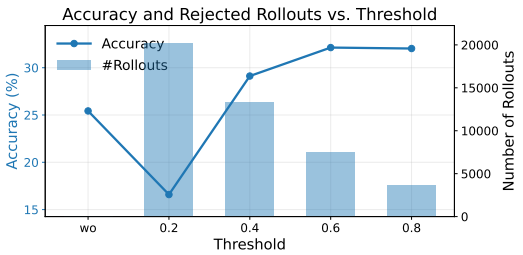


Figure 4: Sensitivity analysis of Rejection threshold on Qwen2.5-3B. The line plot shows F1-Score, and the bar chart depicts the number of rejected training samples.

for **RQ2**. Removing memory-related actions (*w/o MemAction*) substantially degrades performance, e.g., 2Wiki drops from 32.92% to 27.54% and HotpotQA from 44.00% to 28.77%. Removing information-aware rejection (*w/o Reject*) also reduces the average score from 33.55% to 25.43%, while removing both components further lowers it to 24.24%, confirming their complementary contributions.

❷ **Process-level Reward Ablation.** Table 2 also analyzes reward design for **RQ3**. Removing the RAG reward (*w/o* $r_{rag}$) hurts 2Wiki (32.92% to 23.70%) and HotpotQA (44.00% to 34.71%), while removing the memory reward (*w/o* $r_{mem}$) weakens out-of-domain performance, e.g., MusiQue drops from 16.48% to 7.63%. Removing both rewards yields the largest drop (Avg 19.94%), showing that both signals are necessary for effective reasoning and memory utilization.

❸ **Sensitivity to Rejection Threshold.** For **RQ4**, we analyze the effect of the rejection threshold in Figure 4. F1 reaches its lowest value at threshold 0.2 and then steadily improves as the threshold increases, peaking at 0.6 and 0.8. Meanwhile, the number of rejected samples decreases from over 20,000 at 0.2 to the fewest at 0.8, suggesting that moderate-to-high thresholds better balance answer generation and rejection.

## 4 Related Works

**Reinforcement Learning for LLMs** Reinforcement learning has recently shown strong effectiveness in improving LLM reasoning and tool-use capabilities, as demonstrated by OpenAI-O1 (OpenAI, 2025) and DeepSeek-R1 (Guo et al., 2025). While policy gradient methods such as PPO (Schulman et al., 2017) are widely used, they incur high computational costs due to repeated on-policy updates. Alternative approaches, including DPO (Rafailov et al., 2023) and GRPO (Shao et al., 2024), improve training efficiency but face challenges such as off-policy bias or limited exploration. Recent advances target multi-step reasoning and tool interaction, proposing techniques to stabilize training and improve exploration (Dong et al., 2025a,b; Zhou et al., 2025). Nevertheless, existing RL methods remain limited by coarse actions, sparse rewards, and error accumulation in long-horizon reasoning.

## 5 Conclusion and Future Work

We presented AGENTICRAG-R1, a reinforcement learning framework for agentic retrieval-augmented reasoning that integrates fine-grained agentic actions, structured memory control, and action-aware rewards. Experiments across multiple reasoning benchmarks show that AGENTICRAG-R1 consistently outperforms strong RAG and agentic baselines while supporting more effective long-horizon reasoning. Several challenges remain for future work. In particular, jointly optimizing high-level actions and their associated arguments, as well as resolving conflicts among different agentic actions, remains difficult and calls for more principled coordination and credit assignment mechanisms. We also plan to evaluate AGENTICRAG-R1 in broader vertical domains and more open-ended agentic scenarios, such as deep research and long-horizon analytical tasks.

## Limitations

Despite the promising results, our framework does have some limitations that need to be addressed. First, the computational overhead introduced by the memory stack system and dynamic retrieval processes can be relatively high. While these components are essential for managing complex queries and minimizing error accumulation, they do contribute to increased computational costs. In large-scale or real-time applications, this could present challenges related to processing speed and efficiency. Although we have proposed solutions like speculative sampling to alleviate some of this overhead, further optimization is still needed to enhance performance and scalability in time-sensitive environments. Second, our training strategy relies on a rejection-based sampling mechanism that prioritizes more challenging examples to enhance reasoning ability. While effective, this approach requires careful tuning of several hyperparameters, as different configurations can lead to varying performance outcomes. This additional tuning effort may complicate practical deployment and affect performance when transferring the framework to new datasets or tasks. Developing more adaptive or self-calibrating sampling strategies to reduce reliance on manual hyperparameter selection is therefore an important direction for future work.

## Ethical considerations

To evaluate the performance of the proposed method, we carried out experiments exclusively on publicly accessible benchmark datasets, such as GAIA, HotpotQA, MusiQue, and other widely used resources, in accordance with their respective licenses and usage policies. We further confirm that no personally identifiable information was used in this study, and that no human or animal subjects were involved in the research.

## References


Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *ICLR*.

Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. 2020. Factual error correction for abstractive summarization models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6251–6258, Online. Association for Computational Linguistics.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, and 1 others. 2025. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*.

Tianzhe Chu, Yuexiang Zhai, Jihan Yang, Shengbang Tong, Saining Xie, Dale Schuurmans, Quoc V Le, Sergey Levine, and Yi Ma. 2025. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*.

Guanting Dong, Licheng Bao, Zhongyuan Wang, Kangzhi Zhao, Xiaoxi Li, Jiajie Jin, Jinghan Yang, Hangyu Mao, Fuzheng Zhang, Kun Gai, and 1 others. 2025a. Agentic entropy-balanced policy optimization. *arXiv preprint arXiv:2510.14545*.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, and 1 others. 2025b. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Xiaorui Wang, and Zhendong Mao. 2025. Deepresearch bench: A comprehensive benchmark for deep research agents. *arXiv preprint arXiv:2506.11763*.

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

Yuchen Fan, Kaiyan Zhang, Heng Zhou, Yuxin Zuo, Yanxu Chen, Yu Fu, Xinwei Long, Xuekai Zhu, Che Jiang, Yuchen Zhang, and 1 others. 2025. Ssrl: Self-search reinforcement learning. *arXiv preprint arXiv:2508.10874*.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. 2025. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl. *Preprint*, arXiv:2508.07976.

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

Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. 2023. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*.

Hangfeng He, Hongming Zhang, and Dan Roth. 2022. Rethinking with retrieval: Faithful large language model inference. *Preprint*, arXiv:2301.00303.

Xanh Ho, Anh-Khoa Duong, Quoc-Huy Nguyen, and Suong Nguyen. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *COLING*.

Jakub Hoscilowicz, Bartosz Maj, Bartosz Kozakiewicz, Oleksii Tymoshchuk, and Artur Janicki. 2024. Clickagent: Enhancing UI location capabilities of autonomous agents. *CoRR*, abs/2410.11872.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403*.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023a. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023b. Survey of hallucination in natural language generation. *ACM Comput. Surv.*, 55(12).

Xinke Jiang, Yue Fang, Rihong Qiu, Haoyu Zhang, Yongxin Xu, Hao Chen, Wentao Zhang, Ruizhe Zhang, Yuchen Fang, Xu Chu, and 1 others. 2024a. Tc-rag: Turing-complete rag's case study on medical llm systems. *arXiv preprint arXiv:2408.09199*.

Xinke Jiang, Ruizhe Zhang, Yongxin Xu, Rihong Qiu, Yue Fang, Zhiyuan Wang, Jinyi Tang, Hongxin Ding, Xu Chu, Junfeng Zhao, and Yasha Wang. 2024b. Hykge: A hypothesis knowledge graph enhanced framework for accurate and reliable medical llms responses. *Preprint*, arXiv:2312.15883.

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*.

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2020. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *arXiv preprint arXiv:2009.13081*.

Mandar Joshi and 1 others. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Nikhil Kandpal, Haikang Deng, Adam Roberts, Eric Wallace, and Colin Raffel. 2023. Large language models struggle to learn long-tail knowledge. In *International Conference on Machine Learning*, pages 15696–15707. PMLR.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *Preprint*, arXiv:2001.08361.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. *Preprint*, arXiv:1911.00172.

Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqui. 2025. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4745–4759.

Tom Kwiatkowski and 1 others. 2019. Natural questions: A benchmark for question answering research. *TACL*.

Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, and 1 others. 2024. Autowebglm: A large language model-based web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5295–5306.

Myeonghwa Lee, Seonho An, and Min-Soo Kim. 2024. PlanRAG: A plan-then-retrieval augmented generation for generative large language models as decision makers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 6537–6555, Mexico City, Mexico. Association for Computational Linguistics.

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. *arXiv preprint arXiv:2501.05366*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts. *Preprint*, arXiv:2307.03172.

Tianqi Liu, Yao Zhao, Rishabh Joshi, Misha Khalman, Mohammad Saleh, Peter J Liu, and Jialu Liu. Statistical rejection sampling improves preference optimization. In *The Twelfth International Conference on Learning Representations*.

Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, and 1 others. 2025. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6140–6150.

OpenAI. 2025. Openai-o1. Accessed: 2025-05-16.

Akshara Prabhakar, Roshan Ram, Zixiang Chen, Silvio Savarese, Frank Wang, Caiming Xiong, Huan Wang, and Weiran Yao. 2025. Enterprise deep research: Steerable multi-agent deep research for enterprise analytics. *arXiv preprint arXiv:2510.17797*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741.

Daniel Russo and Benjamin Van Roy. 2014. Learning to optimize via posterior sampling. *Preprint*, arXiv:1301.2609.

Daniel Russo and Benjamin Van Roy. 2013. Eluder dimension and the sample complexity of optimistic exploration. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*.

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

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: Language agents with verbal reinforcement learning. *Advances in neural information processing systems*, 36:8634–8652.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Harsh Trivedi and 1 others. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *ACL*.

Minh Duc Vu, Han Wang, Jieshan Chen, Zhuang Li, Shengdong Zhao, Zhenchang Xing, and Chunyang Chen. 2024. Gptvoicetasker: Advancing multi-step mobile task efficiency through dynamic interface exploration and learning. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*, pages 1–17.

Chenglong Wang, Yang Gan, Yifu Huo, Yongyu Mu, Qiaozhi He, Murun Yang, Bei Li, Tong Xiao, Chunliang Zhang, Tongran Liu, and Jingbo Zhu. 2025a. Gram: A generative foundation reward model for reward generalization. *Preprint*, arXiv:2506.14175.

Liang Wang, Haonan Chen, Nan Yang, Xiaolong Huang, Zhicheng Dou, and Furu Wei. 2025b. Chain-of-retrieval augmented generation. *arXiv preprint arXiv:2501.14342*.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, Yuqiong Liu, An Yang, Andrew Zhao, Yang Yue, Shiji Song, Bowen Yu, Gao Huang, and Junyang Lin. 2025c. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *Preprint*, arXiv:2506.01939.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022a. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022b. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Wei Xiong, Jiarui Yao, Yuhui Xu, Bo Pang, Lei Wang, Doyen Sahoo, Junnan Li, Nan Jiang, Tong Zhang, Caiming Xiong, and 1 others. 2025. A minimalist approach to llm reasoning: from rejection sampling to reinforce. *arXiv preprint arXiv:2504.11343*.

Weimin Xiong, Yifan Song, Xiutian Zhao, Wenhao Wu, Xun Wang, Ke Wang, Cheng Li, Wei Peng, and Sujian Li. 2024. Watch every step! llm agent learning via iterative step-level process refinement. *arXiv preprint arXiv:2406.11176*.

Reda Yacouby and Dustin Axman. 2020. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In *Proceedings of the first workshop on evaluation and comparison of NLP systems*, pages 79–91.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, and 25 others. 2024a. Qwen2.5 technical report. *ArXiv*, abs/2412.15115.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024b. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Songhua Yang, Xinke Jiang, Hanjie Zhao, Wenxuan Zeng, Hongde Liu, and Yuxiang Jia. 2024c. Faima: Feature-aware in-context learning for multi-domain aspect-based sentiment analysis. In *COLING*.

Zhilin Yang, Peng Qi, Saizheng Zhang, and 1 others. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

Zijian Zhou, Ao Qu, Zhaoxuan Wu, Sunghwan Kim, Alok Prakash, Daniela Rus, Jinhua Zhao, Bryan Kian Hsiang Low, and Paul Pu Liang. 2025. MEM1: Learning to synergize memory and reasoning for efficient long-horizon agents. *arXiv preprint arXiv:2506.15841*.

# Appendix

## Contents

**1 Introduction** **1**

**2 AGENTICRAG-R1** **3**
2.1 Action Modeling with Hierarchical Rewards . . . . . 3
2.2 Information-Aware Trajectories Rejection . . . . . 5

**3 Experiments** **5**
3.1 Experimental Setup . . . . . 6
3.2 Main Result Analysis . . . . . 6
3.3 Component Analysis . . . . . 7

**4 Related Works** **8**

**5 Conclusion and Future Work** **8**

**A More Related Work** **13**
A.1 Reinforcement Learning for LLMs 13

**B Training Algorithm** **13**

**C Notations Table** **13**

**D Experiment Datasets** **13**
D.1 Training Dataset. . . . . 13
D.2 Testing set. . . . . 14

**E Baseline Implementation Details** **15**

**F Prompt** **16**

**G Implementation Details** **17**

**H Reward Analysis** **17**

**I Additional Experiments** **17**

**J Case Study** **22**

**K Agentic Hacking Theoretical Analysis** **22**
K.1 Outcome-Only Reward Bias in Multi-Step Reasoning . . . . . 22
K.2 Bias Isolation via Pop-Based Stack and Context Masking . . . . . 24

**L More Method Details: Pop-based Attention Mask** **25**

**M Report Generation** **25**

**N Computational Resources and Software Environment** 27

**O The Use of Large Language Model** 27

# A More Related Work

## A.1 Reinforcement Learning for LLMs

Reinforcement learning (RL) has emerged as a powerful paradigm for improving LLM alignment and reasoning, with recent work such as OpenAI-O1 (OpenAI, 2025) and DeepSeek-R1 (Guo et al., 2025) demonstrating strong performance on complex tasks. Methods like PPO (Schulman et al., 2017) have been widely adopted, but require multiple rounds of on-policy optimization, making them computationally expensive and challenging to scale. To improve efficiency, alternatives such as Direct Preference Optimization (DPO) (Rafailov et al., 2023) bypass reward modeling by directly optimizing preference loss, but suffer from off-policy bias. Group Relative Policy Optimization (GRPO) (Shao et al., 2024) offers a compromise by avoiding learned value networks and leveraging intra-group comparisons for relative rewards. Building on these, recent RL methods specifically designed for multi-step reasoning and tool interaction include AEPO (Dong et al., 2025a), which addresses entropy imbalance and rollout collapse in multi-round agents; ARPO (Dong et al., 2025b), which guides exploration of uncertain high-entropy tool calls using advantage attribution; and MEM1 (Zhou et al., 2025), which synergizes memory and reasoning to support long-horizon multi-step interactions. Despite these advances, existing RL approaches still face key challenges: **coarse-grained action modeling, trajectory-level scalar rewards, limited exploration, cumulative errors in multi-step reasoning, and the sparsity of training samples that require diverse reasoning behaviors**.

# B Training Algorithm

Algorithm 1 details the full rollout engine with a maximum action budget $B$.

**Algorithm 1** Dynamic Rollout with Multi-Action Reasoning and Stack-Based Memory

**Require:** query $q$, policy $\pi_\theta$, retriever $\mathcal{R}$, budget $T$.
**Ensure:** Generated reasoning trajectory $y$

```
Initialize trajectory y ← ∅ and memory stack M ← ∅
while t < T do
    Initialize action segment token x_t ← ∅
    while x is not an action delimiter do
        Sample token x ~ π_θ(· | q, y, M)
        Append x to x_t
    end while
    Append x_t to trajectory y
    Parse composed action a_t ← parse(x_t)
    if a_t ∈ {Plan, Think, Search, Conclusion} then
        push(M, content of x_t)
    end if
    if a_t == Search then
        Extract query q_t^s ← parse(x_t)
        Retrieve observation o_t ← R(q_t)
        Append ⟨Observation⟩o_t⟨/Observation⟩ to y
        push(M,⟨Observation⟩o_t⟨/Observation⟩)
        Modify Retrieval Loss Mask
    else if a_t == Backtrack then
        pop(M), push(M, revised reasoning from x_t)
        Modify Pop-based Attention Mask
    else if a_t == Summary then
        pop(M), push(M, summarized content)
        Modify Pop-based Attention Mask
    else if a_t == Conclusion then
        Return y
    end if
    t ← t + 1
end while
Return y
```

# C Notations Table

Table 3 presents a comprehensive list of key notations and symbols used in the AGENTICRAG-R1 framework.

# D Experiment Datasets

## D.1 Training Dataset.

A central challenge in RL-based agent training lies in the limited difficulty of existing QA datasets. Effective supervision requires questions **that demand non-trivial reasoning and long-horizon, multi-step retrieval**. However, widely used multi-hop benchmarks such as HotpotQA and 2WikiMultiHopQA can often be solved with shallow reasoning and zero or only one retrieval step, making them insufficient for learning complex search and planning behaviors. As a result, models trained on such data **tend to overfit short-horizon, template-like reasoning patterns and generalize poorly to challenging agentic scenarios**.

To address this limitation, we construct a curated training dataset following (Gao et al., 2025) by filtering instances from HotpotQA and 2Wiki that are

| Symbol | Description |
|---|---|
| $\mathcal{S}$ | System state capturing the current reasoning context |
| $\mathcal{A}$ | Set of reasoning actions, consisting of primitive actions (push/pop) and composed actions (e.g., Plan, Search) |
| $\mathcal{M}$ | Stack-based short-term memory of reasoning units |
| $\delta$ | State transition function over system state and memory |
| $s_t$ | System state at reasoning step $t$ |
| $a_t$ | Reasoning action taken at step $t$ |
| $\mathcal{M}_t$ | Memory stack at step $t$ |
| $q$ | Input query sampled from the training dataset $\mathcal{D}$ |
| $y$ | Generated reasoning trajectory $(a_1, \ldots, a_T)$ |
| $\pi_\theta$ | Policy model parameterized by $\theta$ |
| $\pi_{\text{ref}}$ | Frozen reference policy for KL regularization |
| $K$ | Number of rollouts generated per input |
| $\mathcal{R}$ | External retrieval module or search engine |
| $r_\phi(q, y)$ | Scalar reward function evaluating trajectory $y$ given input $q$ |
| $x_i^{(k)}$ | The $i$-th generated token in the $k$-th trajectory $y^{(k)}$. |
| $R_G$ | The set of token-level rewards across all trajectories in the current rollout group |
| $r_i^{(k)}$ | The token-level reward assigned to the token $x_i^{(k)}$. |
| $\hat{A}_{i,t}$ | Group-relative advantage for trajectory $i$ at step $t$ |
| $\mathcal{J}(\theta)$ | The Group Relative Policy Optimization (GRPO) objective. |
| $\tilde{z}_{i,t}$ | Clipped importance sampling ratio for trajectory $i$ at step $t$ |
| $\mathbb{D}_{\text{KL}}(\cdot\|\cdot)$ | Kullback–Leibler divergence between policies |
| $\beta$ | KL regularization coefficient |
| $r_{\text{out}}^{(k)}$ | Outcome-level reward evaluating final answer |
| $r_{\text{proc}}^{(k)}$ | Process-level reward in the $k$-th trajectory |
| $\mathbb{I}_{\text{mask},i}^{(k)}$ | Indicator function that equals 1 if the token $x_k^{(i)}$ is masked from policy optimization, and 0 otherwise. |
| $\mathbb{I}_{\text{act},i}^{(k)}$ | Indicator function that equals 1 if the token $x_k^{(i)}$ corresponds to an action invocation, and 0 otherwise. |
| $a_{\text{conclusion}}^{(k)}$ | The predicted final answer extracted from the <Conclusion> action in the $k$-th trajectory $y^{(k)}$ |
| $a_{\text{gold}}$ | The ground-truth answer used for evaluation |
| $r_{\text{fmt},i}^{(k)}$ | Action format reward enforcing syntactic correctness at action level |
| $r_{\text{rag},i}^{(k)}$ | Search-based reward evaluating retrieval relevance |
| $r_{\text{mem},i}^{(k)}$ | Memory-aware reward for backtracking or summarization |
| $\mathbb{I}_{\text{syntax}}(a_i)$ | Indicator for whether $a_i$ has valid <ACTION> tags (1 = valid, 0 = invalid). |
| $\alpha_{\text{type}}(a_i)$ | Action weight: 1.0 for CONCLUSION, 0.5 for other valid actions, 0 if invalid. |
| $q_i^s$ | Search query issued at step $i$ |
| $o_i$ | Retrieved observation at step $i$ |
| $\mathcal{B}$ | Mini-batch size (number of input prompts) |
| $\mathcal{Y}_\mathcal{B}$ | Set of all rollouts in a batch |
| $\mathcal{Y}_\mathcal{B}^{\text{keep}}$ | Rollout subset retained after global selection |
| $\mu^{(b)}$ | Mean reward over rollouts for input $x^{(b)}$ |
| $(\sigma^{(b)})^2$ | Reward variance measuring input-level uncertainty |
| $\tilde{r}_i^{(b)}$ | Information-aware score for rollout $\tau_i^{(b)}$ |
| $p$ | Fraction of top-ranked rollouts retained |

Table 3: Key Notations Used in AGENTICRAG-R1

both solvable and intrinsically require multi-step search. Specifically, for each candidate question, we prompt the training-free TC-RAG (Jiang et al., 2024a) to generate 10 independent trajectories under identical conditions, and discard questions that ❶ are answered incorrectly in all trials, ❷ achieve a correctness rate of 50% or higher, or ❸ can be answered correctly with at most a single retrieval step.

### D.2 Testing set.

In our experiments, we evaluate our method on 9 widely used datasets spanning multi-hop reasoning, open-domain question answering, and real-world agent evaluation, including HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), Bamboogle (Press et al., 2022), FRAMES (Krishna et al., 2025), MusiQue (Trivedi et al., 2022), NQ (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017)) . Dataset statistics, including the number of training, development, and test instances, are provided in Table 4:

- **2WikiMultiHopQA** (Ho et al., 2020) is a large-scale multi-hop question answering dataset constructed from Wikipedia and Wikidata. It contains 192,606 question–answer pairs grounded in a single database, with 154,878 training examples, 12,576 development examples, and 12,576 test examples. The dataset is designed to evaluate multi-hop reasoning that requires aggregating evidence across multiple documents.

- **HotpotQA** (Yang et al., 2018) is a large-scale dataset with 112,779 Wikipedia-based question-answer pairs requiring multi-hop reasoning over multiple documents. The dataset is divided into several subsets for training and evaluation. Specifically, the training set includes 18,089 single-hop questions (*train-easy*), 56,814 multi-hop questions (*train-medium*), and 15,661 hard multi-hop questions (*train-hard*). The development set contains 7,405 hard multi-hop questions, while the test set is further divided into two settings: *test-distractor* with 7,405 hard multi-hop questions and *test-fullwiki* with another 7,405 hard multi-hop questions. This division allows for a comprehensive evaluation of models under different conditions. HotpotQA also provides sentence-level supporting facts required for reasoning, enabling models to improve performance and make explainable predictions.

- **Bamboogle** (Press et al., 2022) is a small, hand-crafted dataset with 125 questions designed to measure the extent to which a question answering system can answer varied compositional questions. It covers many different types of questions on various areas.

- **FRAMES** (Krishna et al., 2025) is a high-quality dataset designed to test LLMs'factual responses, retrieval capabilities, and reasoning in generating final answers. It comprises 824 challenging multi-hop questions requiring integration of information from multiple sources.

- **MuSiQue** (Trivedi et al., 2022) is a multihop question answering dataset constructed from Wikipedia, focusing on questions that require connected reasoning across multiple steps. It contains 19,938 training examples, 2,417 development examples, and 2,459 test examples. MuSiQue is designed to enforce proper multihop reasoning by ensuring that each reasoning step critically relies on the output of the previous step.

- **Natural Questions (NQ)** (Kwiatkowski et al., 2019) is a large-scale question answering dataset based on real user queries issued. Each example pairs a question with a Wikipedia page and includes annotated long and short answers. The dataset contains 307,373 training examples, 7,830 development examples, and 7,842 test examples.

- **TriviaQA** (Joshi et al., 2017) is a large-scale reading comprehension dataset containing over 650K question-answer-evidence triples derived from 95K question-answer pairs authored by trivia enthusiasts. It includes evidence documents from Wikipedia and the Web, with an average of six documents per question. The dataset is divided into training (61,888 for Wiki, 76,496 for Web), development (7,993 for Wiki, 9,951 for Web), and test sets (7,701 for Wiki, 9,509 for Web). TriviaQA is designed to test models' ability to handle complex questions with significant lexical and syntactic variability, requiring multi-sentence reasoning.

## E Baseline Implementation Details

To comprehensively evaluate the effectiveness of the proposed method, we compare it against a diverse set of strong baselines covering different paradigms of reasoning, retrieval, and agentic interaction. The implementation details are described below.

| Dataset | Train | Dev | Test |
|---|---|---|---|
| 2Wiki | 154,878 | 12,576 | 12,576 |
| HotpotQA | 90,564 | 7,405 | 14,810 |
| Bamboogle | 0 | 0 | 125 |
| FRAMES | 0 | 0 | 824 |
| MuSiQue | 19,938 | 2,417 | 2,459 |
| NQ | 307,373 | 7,830 | 7,842 |
| TriviaQA | 138,384 | 17,944 | 17,210 |

Table 4: Overview of datasets used in experiments.

**❶ No-RAG.** This category evaluates the model's capability when no retrieval mechanism is involved, serving to assess how well the LLM performs using only its internal parametric knowledge.

- **No-RAG.** A non-retrieval baseline where the LLM directly generates responses based solely on its parametric knowledge, without access to any external information.

- **CoT (Chain-of-Thought) (Wei et al., 2022b).** Following prior work, this baseline prompts the model to generate explicit intermediate reasoning steps before producing the final answer.

**❷ Naive RAG.** Naive RAG methods incorporate retrieval in a heuristic or fixed manner without explicit reasoning–retrieval coordination, aiming to provide external knowledge through simple retrieval strategies.

- **FS-RAG (Fix Sentence RAG) (Trivedi et al., 2023).** FS-RAG performs retrieval at the sentence level, where each sentence in the input context is independently used as a retrieval query to obtain relevant external knowledge.

- **FL-RAG (Fix Length RAG) (Khandelwal et al., 2020).** FL-RAG invokes retrieval at fixed token intervals, triggering the retrieval module every (n) generated tokens using the preceding token window as the query.

**❸ Agentic RAG.** Agentic RAG methods rely on manually designed prompting strategies to interleave reasoning and retrieval, enabling the model to decide when and how to invoke external tools during inference. knowledge through simple retrieval strategies.

- **ReAct (Yao et al., 2022).** ReAct interleaves reasoning and action steps, enabling the model to interact with external tools or environments during inference.
- **IRCOT (Trivedi et al., 2023).** IRCOT alternates between retrieval and Chain-of-Thought generation, where intermediate reasoning guides retrieval and retrieved evidence is used to refine reasoning.
- **TC-RAG (Jiang et al., 2024a).** TC-RAG introduces a Turing-complete RAG framework with a memory stack, allowing the model to dynamically control retrieval through push and pop.

❹ **RL-based Agentic RAG.** RL-based Agentic RAG approaches apply RL to retrieval-augmented settings, training models to learn when and how to invoke external retrieval tools during multi-step reasoning.

- **ReSearch (Chen et al., 2025).** ReSearch is trained using RL to jointly perform reasoning and search for multi-hop question answering, without requiring supervision over intermediate reasoning steps. We implement this baseline using Proximal Policy Optimization (PPO) (Schulman et al., 2017).
- **Search-R1 (Jin et al., 2025).** Search-R1 trains the LLM via RL to autonomously interact with a search engine during reasoning. It generates multi-round search queries and integrates retrieved information for problem solving. We train this baseline using Group Relative Policy Optimization (GRPO)(Shao et al., 2024).
- **AEPO (Dong et al., 2025a).** AEPO adopts an entropy-balanced policy optimization framework to stabilize training in high-entropy agentic settings by balancing entropy during rollout generation and policy updates.
- **ARPO (Dong et al., 2025b).** ARPO introduces an entropy-aware adaptive rollout strategy that dynamically adjusts sampling at high-entropy decision points, encouraging diverse tool-use behaviors.
- **Mem1 (Zhou et al., 2025).** Mem1 is an end-to-end RL framework for long-horizon multi-turn tasks under constant memory constraints, maintaining a compact internal state for reasoning and memory consolidation.

# F Prompt

In this section, we provide the prompts used in our framework:

**AGENTICRAG-R1 PROMPT TEMPLATE**

**Answer the following question to the best of your ability. You have access to the tools listed below.**

*[Insert* `tool_descriptions` *here]*

The reasoning process and final answer should be wrapped in the <ACTION> and </ACTION> tags, respectively:
1. That is, "<Think> Write the reasoning process here </Think>";
2. "<Conclusion> Write the final answer here </Conclusion>";
3. Use the <Backtrack> tag to wrap your reflection results;
4. Use the <Summary> tag to wrap your summary results;
5. During the thinking process, **if necessary, the assistant can perform a search** to find uncertain knowledge, formatted as "<Search> search query. The search system will then provide the assistant with the retrieved information in the format "<Observation> ...search results... </Observation>".

Please think strictly according to the described process and use the following **Action Format**:

```
User_Query: ...
<Think> ... </Think>
<Search> ... </Search>
<Summary> ... </Summary>
<Backtrack> ... </Backtrack>
...(repeat <Think> / <Search> / <Summary> /
<Backtrack> as needed)
<Think> I now know the final answer. </Think>
<Conclusion> ... </Conclusion>
```

**Begin!**

**GRM Relevance Evaluation Prompt**

You are a relevance evaluator. Your task is to assess the semantic relevance of the retrieved documents with respect to the given question and task context, and output a relevance score between **0.0** and **1.0**, reported to **one decimal place** (e.g., 0.7).

**Question:** `{Question}`
**Search Arguments:** `{Search Args}`
**Retrieved Documents:** `{Search Docs}`

***Scoring Guidelines:***

- **1.0**: Highly relevant; directly answers the question and strongly supports reasoning.
- **0.7–0.9**: Mostly relevant and useful, with minor noise or redundancy.
- **0.4–0.6**: Partially relevant; indirect, incomplete, or weakly aligned.
- **0.1–0.3**: Largely irrelevant or misleading, with only marginal relevance.
- **0.0**: Completely irrelevant to the question.

**Response Format:**
`Relevance Score:`

## G Implementation Details

We implement our method using a GRPO-based reinforcement learning framework. Training is performed with a batch size of 16 and a learning rate of $1 \times 10^{-6}$, without quantization or KV caching. We train for a single GRPO epoch consisting of 300 optimization steps. For each prompt, four stochastic generations are sampled with a temperature of 0.7, diverse sampling enabled, and a diversity penalty of 1.0. Each generation is limited to 2048 new tokens, with a maximum aggregated context length of 4096 tokens and up to ten iterative generation steps per episode. Rewards are computed at the token level. The GRPO optimizer follows a clipped objective with $\beta = 0.04$, $\mu = 2$, and $\epsilon = 0.1$. All hyperparameters are fixed across experiments, as training is conducted on a single curated dataset (Table 5).

## H Reward Analysis

Our process rewards ($r_{\text{rag}}$ and $r_{\text{mem}}$) are scored by a general-purpose LLM judge (GRM) via a fixed evaluation prompt, requiring no additional reward model training. To validate the reliability of this design, we analyze cross-judge consistency, sensitivity to judge choice, and downstream retrieval quality.

**Cross-Judge Consistency.** We evaluate four LLM judges on 35,583 multi-hop samples and compute pairwise Pearson correlation coefficients for the assigned reward scores. As shown in Table 6, all pairs exhibit correlations $\geq 0.85$, indicating strong agreement across judges.

| Hyperparameter | Value |
|---|---|
| Batch Size | 16 |
| Learning Rate | $1 \times 10^{-6}$ |
| GRPO Epochs | 1 |
| Steps per Epoch | 300 |
| Number of Generations | 10 |
| Max New Tokens | 2,048 |
| Max Context Length | 4,096 |
| Max Generate Iterations | 10 |
| Temperature | 0.7 |
| $\beta$ (GRPO) | 0.04 |
| $\mu$ (GRPO) | 2 |
| $\epsilon$ (GRPO) | 0.1 |

Table 5: Implementation details and hyperparameter settings used in all experiments.

**Ablation under Different Reward Models.** We further train AGENTICRAG-R1 on Qwen2.5-1.5B using each individual LLM as the GRM judge, and additionally test a multi-RM ensemble. As shown in Table 7, performance varies minimally across single judges (26.5-26.8 average), while the multi-RM ensemble yields further improvement (27.7), confirming that our framework is robust to the choice of reward model.

**Retrieval Quality Analysis.** Beyond final answer accuracy, we evaluate retrieval-augmented answer quality before and after RL training using RAGAS metrics and an LLM-based judger. As shown in Table 8, training with AGENTICRAG-R1 improves all retrieval quality dimensions, indicating that the framework learns to retrieve more relevant documents and integrate them more faithfully.

## I Additional Experiments

**❶ Statistical Analysis of Reasoning and Tool Usage.** Table 9 compares the reasoning and tool usage statistics of AGENTICRAG-R1 with TC-RAG on Qwen2.5-3B. Overall, AGENTICRAG-R1 demonstrates more efficient and structured agentic behavior. Specifically, it produces slightly longer reasoning paths (Avg. Actions increased by $\sim$12.2%), indicating deeper multi-step reasoning. At the same time, AGENTICRAG-R1 maintains a notably shorter response length (increased by $\sim$5.1%), suggesting improved reasoning efficiency rather than verbosity. Moreover, it achieves

| Judge | GPT-4o | DeepSeek-V3.2 | Qwen2.5-72B | DouBao |
|---|---|---|---|---|
| GPT-4o | 1.00 | 0.90 | 0.88 | 0.87 |
| DeepSeek-V3.2 | 0.90 | 1.00 | 0.89 | 0.86 |
| Qwen2.5-72B | 0.88 | 0.89 | 1.00 | 0.85 |
| DouBao | 0.87 | 0.86 | 0.85 | 1.00 |

Table 6: Pearson correlation of GRM scores across four LLM judges on 35,583 samples.

| GRM Judge | 2Wiki | HotpotQA | Bamboogle | FRAMES | MusiQue | NQ | TriviaQA | Avg |
|---|---|---|---|---|---|---|---|---|
| GPT-4o | 28.1 | 29.8 | 19.0 | 13.0 | 15.2 | 30.4 | 49.9 | 26.5 |
| DeepSeek-V3.2 | 28.8 | 30.0 | 18.8 | 12.9 | 15.7 | 31.0 | 50.1 | 26.8 |
| Qwen2.5-72B | 28.5 | 29.9 | 18.7 | 13.0 | 15.3 | 30.7 | 50.0 | 26.6 |
| DouBao | 28.4 | 29.8 | 18.9 | 12.8 | 15.3 | 30.6 | 49.8 | 26.5 |
| Multi-RM Ensemble | **29.5** | **30.9** | **20.0** | **13.5** | **16.5** | **31.4** | **51.8** | **27.7** |

Table 7: Performance on Qwen2.5-1.5B with different GRM judges.

| Metric | Before Training | After Training |
|---|---|---|
| Faithfulness | 62 | 78 |
| Answer Relevancy | 75 | 82 |
| Context Precision | 68 | 76 |
| Context Recall | 61 | 79 |
| Judger Reward | 0.62 | 0.83 |

Table 8: Retrieval quality metrics before and after training.

| Metric | TC-RAG | AGENTICRAG-R1 |
|---|---|---|
| Avg.Time (s) | 11.86 | 10.15 |
| Avg.ResponseLen | 838.14 | 880.89 |
| Avg.Actions | 10.14 | 11.38 |
| Tool-Call Ratio (%) | 91.70 | 97.50 |

Table 9: Statistical comparison of inference latency, response complexity, and tool-use behaviors between TC-RAG and AGENTICRAG-R1 on Qwen2.5-3B.

a higher tool usage ratio (97.5% vs 91.7%, ∼5.8% increase), reflecting more effective and purposeful interaction with external tools. These results indicate that our agentic training framework encourages deeper reasoning while avoiding redundant tool calls.

**❷ Medical Experiment under High-Noise Settings.** To investigate the robustness of AGENTICRAG-R1in high-noise and domain-specific scenarios, we further conduct experiments on the MedQA dataset (Jin et al., 2020). Table 10 reports the performance of different methods evaluated using Qwen2.5-72B as the judge. Compared to the base model, ReAct improves format accuracy but suffers from a noticeable drop in answer accuracy, indicating increased sensitivity to noisy intermediate reasoning. In contrast, AGENTICRAG-R1 achieves substantial gains in both format accuracy (92%) and answer accuracy (87%), demonstrating that our method remains effective even under noisy medical settings. These results suggest that our R1-based training framework exhibits strong robustness to noise and generalizes well to challenging vertical domains such as medicine.

| Method | Format Acc (%) | Answer Acc (%) |
|---|---|---|
| Base | 39% | 84% |
| ReAct | 56% | 79% |
| AGENTICRAG-R1 | 92% | 87% |

Table 10: Results on the MedQA test set. The Qwen2.5-7B-Instruct model is evaluated using Qwen2.5-72B as the judge.

**❸ Action-Specific Importance Analysis** To analyze how different behaviors engage model parameters, we define a layer-wise importance score based on neuron activations. For each action token (including both the action token and its argument tokens), we extract its hidden representation at every transformer layer and count the number of neurons with positive activations after the ReLU nonlinearity:

$$\text{Importance}(l) = \sum \mathbf{ReLU}(\text{embedding}_l > 0), \tag{9}$$

which measures intensity of neuron participation at layer $l$. Each embedding is 2048-dimensional and is visualized by averaging every 100 dimensions.

Importance scores are normalized within each action, as shown in the upper panels of Fig. 6. All statistics are computed on a challenging reasoning

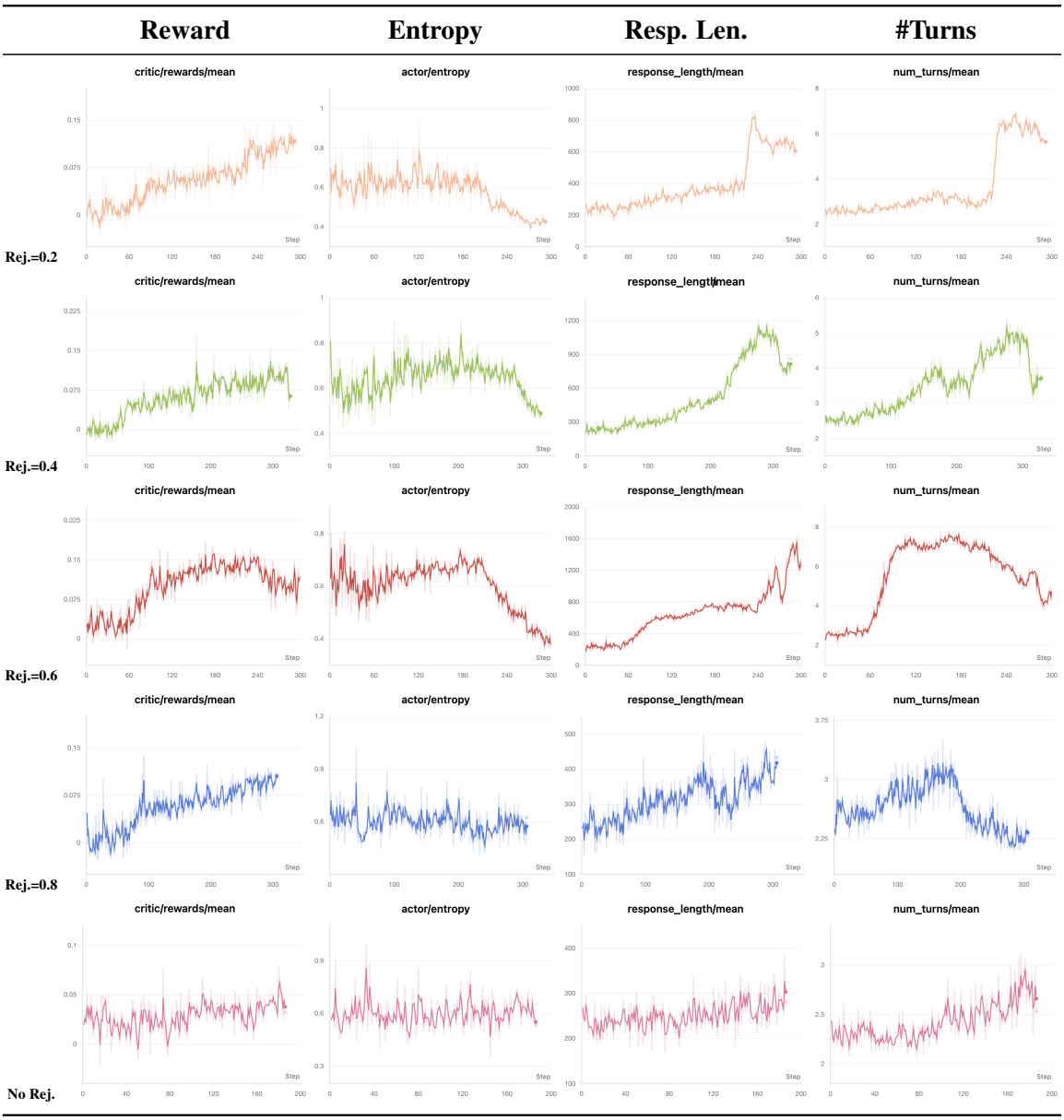


Figure 5: Training dynamics of AGENTICRAG-R1 during GRPO optimization on Qwen2.5-3B under different rejection ratios.

dataset collected via rejection sampling, retaining only correct-answer trajectories to avoid spurious activations from erroneous reasoning. The top six heatmaps present the *normalized* importance profiles of six actions. At a coarse level, these actions exhibit broadly similar activation patterns, suggesting shared use of the model's representation space. However, their importance distributions are not identical and encode action-specific differences in both magnitude and layer-wise allocation that are difficult to discern without direct comparison.

To make these distinctions explicit, the bottom row shows pairwise **differences computed from unnormalized importance scores**. Clear contrasts emerge between <Search> and <Think>, as well as between <Backtrack> and <Think>, indicating substantial divergence in their underlying neuron activations. In contrast, the difference between <Summary> and <Backtrack> is relatively small, suggesting that these memory-related actions rely on highly overlapping neural resources. Overall, the results indicate that tool invocation (<Search>), internal reasoning (<Think>), and memory manipulation (<Backtrack>, <Summary>) correspond to distinct and largely decoupled activation patterns. **This finding supports explicit action-level decomposition and fine-grained optimization, rather than treating all behaviors as a single**

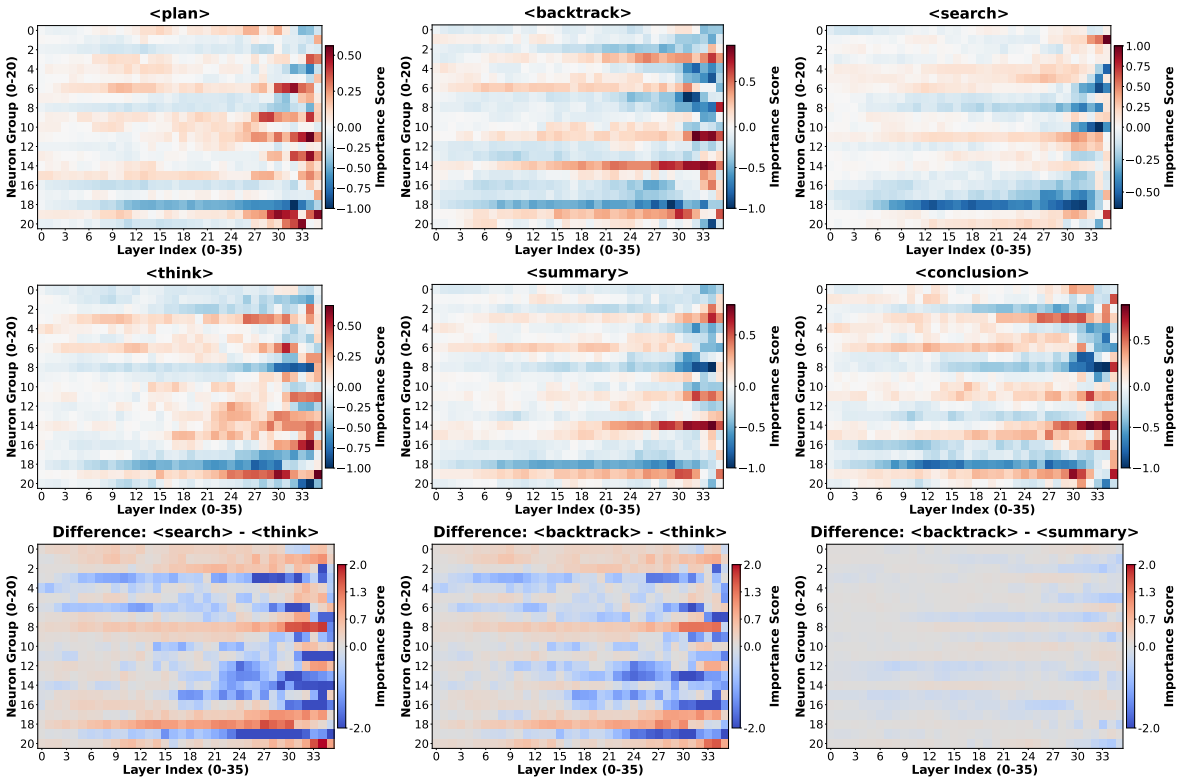


Figure 6: Neuron-wise importance analysis of different actions. Top two rows: normalized importance scores for individual actions. Bottom row: pairwise differences of unnormalized importance scores between actions.

**homogeneous reasoning process.**

Building on the action-level importance analysis above, we further examine activation patterns at a finer action–argument granularity. As shown in Fig. 7, we separately compute layer-wise importance scores for action tokens and their associated argument tokens using the same activation-sum metric as before, but aggregating activations across layers instead of tokens. While the overall trends remain consistent with the action-level results, clear divergences emerge across several contiguous layer ranges (highlighted in red), where different actions exhibit distinct layer-wise distributions for both their action tags and argument tokens. Importantly, although both components are consistently involved across actions, their relative contributions vary across layers in an action-dependent manner. This suggests that different actions are characterized by distinct layer allocation patterns between action tags and arguments, rather than differences in overall activation magnitude alone.

**❹ Action-Specific Optimization with Self-Search and Decoupling.** Motivated by the observation that different actions rely on distinct and only partially overlapping neural resources, we investigate action-specific parameter optimization guided by model Shapley analysis. Specifically, we identify action-relevant parameters using SP-R1 and decouple the optimization by assigning different subsets of parameters to the args layer and action layer, allowing each action to be updated independently. We evaluate this approach on MedQA under two search paradigms: External Search (ES), where both training and testing rely on external Wiki retrieval, and Self-Search (SS) (Fan et al., 2025), which generates candidate retrievals during training without external knowledge and uses external Wiki search only at test time. This self-supervised search encourages better generalization to unseen questions. Table 11 reports the results: incorporating SS alone improves accuracy over ES (from 65.5% to 67.5%), demonstrating its strong effect on generalization, while decoupled, action-specific optimization (Decoup) further boosts performance to 72.2% under SS and 68.5% under ES. These results highlight that SS significantly enhances generalization by exposing the model to diverse self-generated retrievals during training, and that Decoup effectively leverages action-specific parameter structure, enabling targeted optimization per action and yielding consistent gains across paradigms.

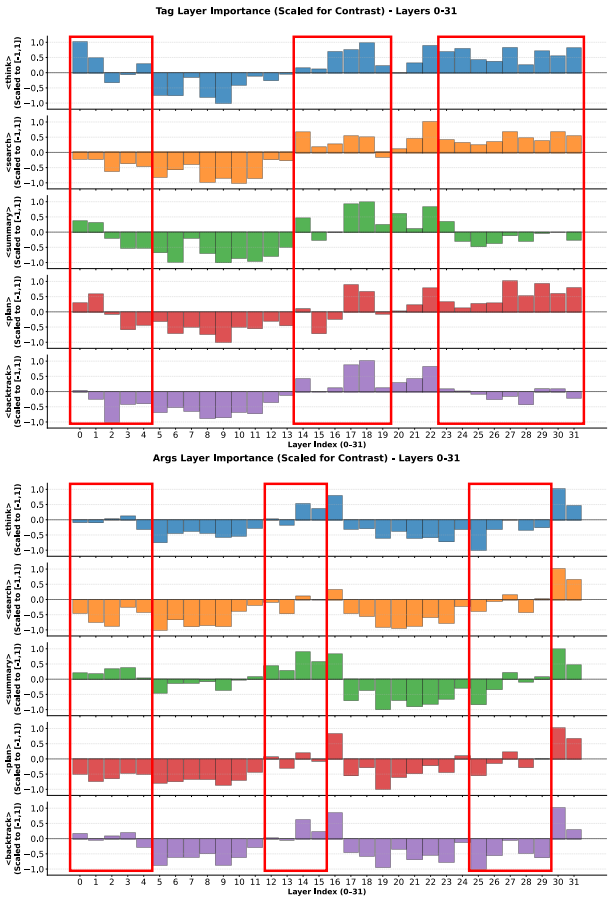


Figure 7: Layer-wise importance profiles of action tokens and their argument tokens.

| Paradigm | Method | Acc (%) |
|---|---|---|
| Raw | Base | 63.5 |
| | React | 65.5 |
| ES | AGENTICRAG-R1+ ES | 66.3 |
| | AGENTICRAG-R1+ ES + Decoup | **68.5** |
| SS | AGENTICRAG-R1+ SS | 67.5 |
| | AGENTICRAG-R1+ SS+ Decoup | **72.2** |

Table 11: Effect of action-specific (decoupled) parameter optimization under different search paradigms.

| Method | Max Steps | | | | |
|---|---|---|---|---|---|
| | 10 | 15 | 20 | 25 | 30 |
| *2Wiki* | | | | | |
| ReAct | 18.23 | 22.45 | 25.17 | 27.89 | 29.03 |
| Search-R1 | 24.12 | 38.02 | 37.51 | 33.08 | 44.27 |
| TC-RAG | 22.78 | 37.35 | 36.92 | 32.41 | 43.10 |
| AGENTICRAG-R1 | 23.91 | 31.81 | 32.53 | 38.61 | 47.45 |
| *Bamboogle* | | | | | |
| ReAct | 15.67 | 18.92 | 20.34 | 21.76 | 22.19 |
| Search-R1 | 37.05 | 34.11 | 37.02 | 28.03 | 28.19 |
| TC-RAG | 36.42 | 33.34 | 36.36 | 27.27 | 27.23 |
| AGENTICRAG-R1 | 10.03 | 31.81 | 14.29 | 21.43 | 35.47 |
| *TriviaQA* | | | | | |
| ReAct | 28.76 | 30.12 | 31.05 | 31.42 | 31.68 |
| Search-R1 | 34.21 | 36.12 | 33.57 | 33.09 | 32.28 |
| TC-RAG | 33.57 | 35.48 | 32.91 | 32.44 | 31.62 |
| AGENTICRAG-R1 | 37.53 | 32.53 | 39.45 | 43.50 | 55.28 |

Table 12: Long-horizon performance under different maximum step budgets across multiple datasets.

**❺Long-Horizon Experiments.** To address **RQ5**, we progressively increase the maximum number of allowed steps from 10 to 15, 20, 25, and 30, thereby evaluating its capability to perform long-horizon reasoning effectively.

As shown in Table 12, our model exhibits clear and consistent gains as the maximum step budget increases, demonstrating strong scalability in long-horizon settings. This trend holds across all evaluated datasets, indicating that the model can effectively leverage extended interaction horizons to perform deeper multi-step reasoning and retrieval. In contrast, TC-RAG shows unstable or even degraded performance when the horizon grows, suggesting limited ability to accumulate and utilize information over long trajectories. These results highlight that simply increasing the allowed steps is insufficient; rather, the proposed design is crucial for unlocking effective long-horizon reasoning capabilities. Furthermore, compared to ReAct and Search-R1, which either plateau early or fluctuate inconsistently across step budgets, AGENTICRAG-R1 exhibits monotonically improving trends on most benchmarks, further validating its effectiveness in long-horizon settings.

**❻ Multi-Run Statistical Analysis.** To verify the consistency of our improvements, we report mean and standard deviation over multiple independent runs on Qwen2.5-3B across all seven benchmarks. As shown in Table 13, AGENTICRAG-R1 consistently outperforms baselines with non-overlapping confidence intervals on most benchmarks.

**❼ Controlled Comparison under PPO.** To eliminate confounding from optimizer differences, we train both SearchR1 and AGENTICRAG-R1 under PPO with aligned backbone, training data, and compute budget. As shown in Table 14, AGENTICRAG-R1 consistently outperforms SearchR1+PPO (32.53 vs. 29.33 average), confirming that the gains stem from the framework design rather than the choice of optimizer.

**❽ Robustness under Retrieval Noise.** To evaluate robustness when retrieval quality degrades, we inject two types of noise following TC-RAG (Jiang et al., 2024a): **Structural noise** (completely replac-

| Method | 2Wiki | HotpotQA | Bamboogle | FRAMES | MusiQue | NQ | TriviaQA | Avg |
|---|---|---|---|---|---|---|---|---|
| ReSearch | 27.23±0.40 | 33.96±0.48 | 15.09±0.32 | 10.00±0.28 | 9.47±0.26 | 34.61±0.52 | 53.93±0.61 | 26.33±0.46 |
| Search-R1 | 29.90±0.45 | 37.24±0.55 | 29.90±0.46 | 10.76±0.30 | 13.53±0.33 | 34.73±0.51 | 55.08±0.63 | 30.16±0.46 |
| Ours | **32.92**±0.48 | **44.00**±0.38 | **31.48**±0.35 | **16.23**±0.36 | **16.48**±0.37 | **37.15**±0.37 | **56.62**±0.65 | **33.55**±0.42 |

Table 13: Mean ± std over multiple runs on Qwen2.5-3B.

| Method | 2Wiki | HotpotQA | Bamboogle | FRAMES | MusiQue | NQ | TriviaQA | Avg |
|---|---|---|---|---|---|---|---|---|
| SearchR1+PPO | 29.34 | 36.27 | 27.14 | 13.19 | 13.00 | 33.02 | 53.33 | 29.33 |
| Ours+PPO | **31.82** | **42.63** | **30.51** | **15.26** | **15.63** | **36.01** | **55.85** | **32.53** |

Table 14: Controlled comparison under PPO optimization on Qwen2.5-3B.

ing retrieved documents with irrelevant ones) and **Partial noise** (mixing relevant and irrelevant documents). As shown in Table 15, under noise injection the model adaptively increases <Backtrack> usage, demonstrating self-corrective behavior.

| Attack Type | Summary (%) | Backtrack (%) | EM |
|---|---|---|---|
| Structural | 18 | 66 | 34 |
| Partial | 48 | 36 | 29 |

Table 15: Model behavior under retrieval noise attacks.

**❾ Cross-Domain Rollout Reward Statistics.** To examine whether the trajectory rejection strategy remains effective across different domains and model scales, we report the mean and variance of rollout rewards. As shown in Table 16, meaningful rollout variance is preserved across all domains, confirming that the reward distributions do not collapse into indistinguishable ranges.

| Domain | Qwen2.5-3B | | Qwen2.5-7B | |
|---|---|---|---|---|
| | Mean | Var | Mean | Var |
| Math | 0.44 | 0.07 | 0.46 | 0.09 |
| Medical | 0.37 | 0.10 | 0.53 | 0.12 |
| General | 0.41 | 0.12 | 0.65 | 0.11 |

Table 16: Rollout reward statistics across domains and model scales.

**❿ Scaling Analysis across Model Sizes.** We compare average performance gains of AGENTICRAG-R1 over TC-RAG across three model scales. Smaller models benefit proportionally more from the framework, while larger models achieve the best absolute performance:

- Qwen2.5-1.5B: 17.9 → 27.6 (+54%)
- Qwen2.5-3B: 22.2 → 33.5 (+50%)
- Qwen2.5-7B: 30.5 → 36.6 (+20%)

This pattern suggests that the structured action space and memory mechanisms are particularly effective at compensating for limited parametric capacity in smaller models.

## J Case Study

We present a representative case study to illustrate the model's ability to perform adaptive retrieval refinement and error correction. As shown in Table 17, the model initially issues broad retrieval queries that return irrelevant or noisy information, including unrelated medical content and off-topic encyclopedic entries. Instead of committing such noise to its internal state, the model explicitly invokes backtracking to revise retrieval keys and progressively narrow the search scope. Through multiple iterations of retrieval, inspection, and memory revision, the model correctly determines that *Battleship* is the 2012 military science fiction action film directed by Peter Berg in which Jason Snell worked. These cases highlight the effectiveness of fine-grained memory control and retrieval refinement in mitigating error accumulation during multi-step reasoning.

## K Agentic Hacking Theoretical Analysis

### K.1 Outcome-Only Reward Bias in Multi-Step Reasoning

[Error Amplification under Outcome-Only Rewards] Consider a multi-step reasoning trajectory $y = (a_1, a_2, \ldots, a_T)$ generated by a policy $\pi_{\boldsymbol{\theta}}(a_t \mid s_t)$ conditioned on an initial query $q_0$. Let $r_{\text{out}}$ denote an outcome-level reward assigned solely based on the correctness of the final answer. Under **outcome-only reward assignment without action-level reward decomposition**, each reasoning step $a_t$ receives an identical implicit reward

| | |
|---|---|
| **Question** | On which 2012 American military science fiction action film, directed by Peter Berg, did Jason Snell work? |
| **Ground Truth** | Battleship |
| **Output** | <Plan> First, I need to find the 2012 American military science fiction action film directed by Peter Berg. Then, I will search for the actors who worked on the film to identify Jason Snell. </Plan><br><Search> [Wiki_RAG]: **Peter Berg's** films </Search><br><Observation> '1. Gerald T. Olson is an American film producer best known for films such as House Party, Bloody Birthday and Dumb & Dumber. The San Francisco State alum relocated...'</Observation><br><Backtrack> While the tool did provide some information about films directed by Peter Berg, **it didn't specify the 2012 military** science fiction action film I was looking for... Let me search for the 2012 military science fiction action film directed by Peter Berg. </Backtrack><br><Search>[Wiki_RAG]: Search for detailed information about the **2012 military** sci-fi action film **Peter Berg**</Search><br><Observation> 'Wu Junyong (; born 1978) is a Chinese artist born in Putian, a city in Fujian province, China...'</Observation><br><Backtrack> The tool didn't provide the information about the 2012 military science fiction action film directed by Peter Berg. I will now try searching for that specific film. </Backtrack><br><Search> [Wiki_RAG]: **Peter Berg movies 2012** </Search><br><Observation> '1. House Party -1991 - drama 2. Bloody Birthday -1992 - horror 3. Dumb & Dumber -1994 - comedy 4. The Rundown -2003 - action 5. Patriots Day -2016 - drama The Hidden -1999 - horror Fast & Furious 6 -2013 - action' </Observation><br><Backtrack> From the previous searches, **no information about a 2012 military science fiction action film** directed by Peter Berg and which Jason Snell worked on was retrieved. This information needs to be carefully re-evaluated and corrected. </Backtrack><br><Search> [Wiki_RAG]: 2012 military science fiction action film directed by Peter Berg Jason Snell worked</Search><br><Observation> '1. Peter Berg (born March 11, 1964) is an American director, producer, writer, and actor... **the military science fiction war film Battleship (2012)**...' </Observation><br><Think> From the search result, **I see that the 2012 movie directed by Peter Berg is "Battleship"**. However, the query mentions Jason Snell worked in it. Let me **verify** if Jason Snell indeed played a role in this film. </Think><br><Search> [Wiki_RAG]: Jason Snell in Battleship</Search><br><Observation> '1. Jason H. Snell is an American special... Indiana Jones and the Kingdom of the **Battleship (2012)**...'</Observation><br><Summary> **After verifying with the search results, it is clear that "Battleship" (2012)** is the 2012 military science fiction action film directed by Peter Berg and in which Jason Snell worked. He played a role in this film.</Summary><br><Conclusion> The 2012 military science fiction action film directed by Peter Berg in which Jason Snell worked is **"Battleship"**.</Conclusion> |

Table 17: A Case Study of a Wiki Problem on AGENTICRAG-R1.

$\tilde{r}_t = \frac{r_{\text{out}}}{T}$, regardless of its semantic correctness[1]. For any **erroneous or noisy step** $a_e$ with true marginal contribution $\phi_e < 0$, the policy model parameter $\boldsymbol{\theta}_k$ at step $k$ update satisfies that:

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \mu \cdot \frac{r_{\text{out}}}{T} \cdot \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_e \mid s_e), \quad (10)$$

where $\mu > 0$ is the learning rate, $\log \pi_{\boldsymbol{\theta}}(a_e \mid s_e)$ is the output logits. Consequently, the generation probability of $a_e$ increases whenever $r_{\text{out}} > 0$, irrespective of its negative contribution.

*Proof.* Under the RL estimator with outcome-only rewards, the policy gradient for a trajectory $y$ is:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}[R_y] = \mathbb{E}\left[ r_{\text{out}} \sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t) \right]. \quad (11)$$

By linearity of expectation, this is equivalent to assigning each step an identical implicit reward $\tilde{r}_t = r_{\text{out}}/T$:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}[R_y] = T \sum_{t=1}^{T} \mathbb{E}[\tilde{r}_t \cdot \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t)]. \quad (12)$$

This formulation implicitly assumes uniform contribution across steps. In practice, however, reasoning steps exhibit heterogeneous marginal contributions $\{\phi_t\}_{t=1}^{T}$, where $\sum_{t=1}^{T} \phi_t = r_{\text{out}}$, and some $\phi_t$ **may be negative due to hallucination, spurious inference, or retrieval noise**. Thus, consider an erroneous step $a_e$ with $\phi_e < 0$. The induced parameter update is

$$\Delta\boldsymbol{\theta} = \mu \cdot \frac{r_{\text{out}}}{T} \cdot \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a_e \mid s_e). \quad (13)$$

Applying first-order Taylor expansion to step update:

$$\begin{aligned} \pi_{\boldsymbol{\theta}_{k+1}}(a_e) &= \pi_{\boldsymbol{\theta}_k}(a_e) + \nabla_{\boldsymbol{\theta}} \pi_{\boldsymbol{\theta}_k}(a_e) \cdot \Delta\boldsymbol{\theta} + \mathcal{O}(\mu^2) \\ &= \pi_{\boldsymbol{\theta}_k}(a_e) \left( 1 + \mu \cdot \frac{r_{\text{out}}}{T} \left\| \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_k}(a_e \mid s_e) \right\|_2^2 \right) + \mathcal{O}(\mu^2). \end{aligned}$$

where we used the log-derivative identity $\nabla_{\boldsymbol{\theta}} \pi = \pi \nabla_{\boldsymbol{\theta}} \log \pi$. Since $r_{\text{out}} > 0$ for correct outcomes and gradient norms are non-negative, the bracketed term exceeds unity, implying

$$\pi_{\boldsymbol{\theta}_{k+1}}(a_e) > \pi_{\boldsymbol{\theta}_k}(a_e) \quad \text{regardless of } \phi_e < 0. \quad (14)$$

[1]Here we ignore the group-relative advantage normalization for simplicity.

To characterize long-term accumulation, define the **step-level reward allocation error** as:

$$\varepsilon_e = \frac{r_{\text{out}}}{T} - \phi_e. \quad (15)$$

For erroneous steps, $\varepsilon_e > 0$ whenever $r_{\text{out}} > 0$. Over $K$ training iterations, the cumulative bias satisfies

$$\mathbb{E}\left[ \|\boldsymbol{B}_e^{(K)}\|_2 \right] \simeq \varepsilon_e \cdot K \cdot \mathbb{E}\left[ \left\| \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_e \mid s_e) \right\|_2 \right], \quad (16)$$

demonstrating monotonic amplification of erroneous reasoning steps (**memory-aware actions**) in the absence of backtracking or summarizing. □

### K.2 Bias Isolation via Pop-Based Stack and Context Masking

[Structural Isolation via Pop-Based Backtracking] Consider a multi-step reasoning trajectory $y = (a_1, a_2, \ldots, a_T)$ generated by a policy $\pi_{\boldsymbol{\theta}}(a_t \mid s_t)$. Suppose that at step $e$, an erroneous action $a_e$ is detected and removed via a pop-based stack mechanism, producing a pruned trajectory $\tilde{y}$. The removal is implemented by masking $a_e$ from the conditioning context of all subsequent actions. Under this counterfactual trajectory $\tilde{y}$, the expected policy gradient update contains no contribution from $a_e$:

$$\mathbb{E}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_e \mid s_e) \cdot r_{\text{out}}] = 0. \quad (17)$$

Thus, erroneous reasoning steps do not receive outcome-level rewards and are not systematically reinforced.

*Proof.* Under outcome-only rewards, the standard advantage estimator is

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}[R_y] = \mathbb{E}\left[ r_{\text{out}} \sum_{t=1}^{T} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t) \right]. \quad (18)$$

**Pop-based backtracking and context masking.** When $a_e$ is removed, all subsequent actions are generated or evaluated under a masked context in which $a_e$ is excluded. The resulting trajectory is

$$\tilde{y} = (a_1, \ldots, a_{e-1}, a'_{e+1}, \ldots, a'_T). \quad (19)$$

The attention mask ensures that $a_e$ does not contribute to hidden states or logits involved in producing the final answer, effectively addressing the problem of reward hacking from outcome reward to be allocated the harmless popped actions.

**Counterfactual independence.** Under the masked context, the outcome distribution satisfies

$$p_{\boldsymbol{\theta}}(a_T \mid a_1, \ldots, a_{e-1}, \cancel{a_e}, \ldots) = p_{\boldsymbol{\theta}}(a_T \mid \tilde{y}), \tag{20}$$

implying that $r_{\text{out}}$ depends only on $\tilde{y}$ and is conditionally independent of $a_e$.

**Vanishing gradient contribution.** The policy gradient under $\tilde{y}$ becomes

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}[R_{\tilde{y}}] = \mathbb{E}\left[ r_{\text{out}} \sum_{t \in \tilde{y}} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_t \mid s_t) \right]. \tag{21}$$

Since $a_e \notin \tilde{y}$, the corresponding score-function term never appears. Equivalently,

$$\mathbb{E}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_e \mid s_e) \cdot r_{\text{out}}] = 0. \tag{22}$$

Therefore, erroneous reasoning steps are structurally isolated from outcome-level rewards. □

**Discussion.** Rather than numerically redistributing rewards, the pop-based stack mechanism corrects outcome-only reward bias through *structural isolation*. By explicitly removing erroneous actions from the conditioning context via attention masking, policy gradients are evaluated on a counterfactual trajectory in which these actions never occurred, eliminating the error amplification effect identified in Section K.1.

## L More Method Details: Pop-based Attention Mask

We extend the standard causal attention mechanism with a pop-based attention mask to handle dynamic context updates induced by `<summary>` and `<backtrack>` operations (`<pop>`). In autoregressive Transformers, causal attention ensures that each token at position ( $i$ ) can only attend to tokens at positions ( $j \leq i$ ). However, in our setting, causal ordering alone is insufficient, since tokens that appear earlier in the sequence may have been explicitly **removed from the active context** via a pop operation and should no longer be accessible.

Consider the example shown in Figur 2 Part 4, where the sequence follows an **Action–Summary** pattern: **Act. 1** → **Act. 2** → **Sum. 3** → **Act. 4**. The summary token **Sum. 3** aggregates the information of **Act. 2**, after which **Act. 2** is popped from the active memory. As a result, when generating **Act. 4**, the model should be allowed to attend to **Sum. 3**, but must be prevented from directly attending to **Act. 2**, even though **Act. 2** appears earlier in the sequence. Thus, the attention mask between **Act. 2** and **Act. 4** should be False. This constraint is indicated as **Mask after POP** in the attention matrix. Formally, let ( $M^{\text{causal}}$ ) denote the standard causal mask:

$$M^{\text{causal}}_{i,j} = \begin{cases} 0, & j \leq i \\ -\infty, & j > i \end{cases} \tag{23}$$

We introduce an additional pop-based mask ( $M^{\text{pop}}$ ), where a token ( j ) is masked if it has been popped before step ( i ):

$$M^{\text{pop}}_{i,j} = \begin{cases} -\infty, & \text{if} j \text{ is popped before } i \\ 0, & \text{otherwise} \end{cases} \tag{24}$$

The final attention mask is defined as:

$$M_{i,j} = M^{\text{causal}}_{i,j} + M^{\text{pop}}_{i,j}. \tag{25}$$

This formulation ensures that once an action has been summarized and popped, its information is only accessible through the corresponding summary token in all subsequent steps.

## M Report Generation

Figure 8 presents the report-generation performance of AGENTICRAG-R1 on the 7B model, inferred with the query: *"Give me a report of topic <RAG's Future in Medical Domain>."* As shown, after multiple rounds of retrieval, the model produces a high-quality report accompanied by well-aligned citations. This result further demonstrates the strong generalization capability of our reinforcement learning–based agent training strategy.

To quantitatively evaluate the generalization of AGENTICRAG-R1 to long-horizon agentic scenarios beyond QA, we benchmark on DeepResearch Bench (Du et al., 2025), a multi-document report generation task that requires sustained multi-step retrieval, information aggregation, and structured composition. We evaluate using RACE (report quality) and FACT (factual consistency, including effective citation accuracy and claim-level accuracy). As shown in Table 18, AGENTICRAG-R1 substantially outperforms all baselines, including RAG, ReACT, and EDR (Prabhakar et al., 2025), achieving the highest RACE score (43.60) and a significant margin on FACT metrics (+8.14 on Eff.c. and +8.56 on C.acc. over the best baseline).

## Retrieval-Augmented Generation (RAG) in the Medical Field: Applications, Benefits, and Challenges

### Key Points

- **Real-time Clinical Decision-Making:** RAG provides evidence-based recommendations, crucial for quick and accurate clinical decisions.
- **Medical Information Extraction:** Extracts and synthesizes medical data from patient records, literature, and guidelines, improving data accuracy.
- **Patient Diagnosis:** Assists in diagnosing patients by integrating relevant medical data with AI models.
- **Enhanced Accuracy and Transparency:** Combines model knowledge with external retrieval, reducing AI-generated misinformation.
- **Ethical and Privacy Concerns:** Significant challenges include privacy, bias, and data security issues that need to be addressed for responsible deployment.

### Overview

Retrieval-Augmented Generation (RAG) is a cutting-edge technology that combines information retrieval and generative AI to enhance various aspects of healthcare. By leveraging both model knowledge and external data sources, RAG aims to provide more accurate, transparent, and up-to-date information, thereby improving clinical decision-making, patient diagnosis, and data management.

### Overview

Retrieval-Augmented Generation (RAG) is a cutting-edge technology that combines information retrieval and generative AI to enhance various aspects of healthcare. By leveraging both model knowledge and external data sources, RAG aims to provide more accurate, transparent, and up-to-date information, thereby improving clinical decision-making, patient diagnosis, and data management.

### Detailed Analysis

#### Applications of RAG in Medicine

1. **Real-time Clinical Decision-Making**
   - **Use Case:** RAG systems can provide healthcare professionals with evidence-based recommendations, enabling them to make quick and accurate decisions.
   - **Source:** A Scoping Review of Retrieval-Augmented Generation—Ethics, Bias
2. **Medical Information Extraction**
   - **Use Case:** RAG can extract and synthesize medical data from various sources, such as patient records, research literature, and clinical guidelines, improving the accuracy and completeness of medical information.
   - **Source:** Enhancing medical AI with retrieval-augmented generation: A mini
3. **Patient Diagnosis**
   - **Use Case:** RAG can assist in diagnosing patients by integrating relevant medical data with AI models, providing a more comprehensive and accurate diagnosis.
   - **Source:** Enhancing medical AI with retrieval-augmented generation: A mini
4. **Data Accessibility**
   - **Use Case:** RAG on FHIR (Fast Healthcare Interoperability Resources) makes healthcare data more accessible by retrieving real-time data based on specific questions.

(a) Report Part (1/2)

   - **Source:** A Scoping Review of Retrieval-Augmented Generation—Ethics, Bias
3. **Reduction of Hallucinations**
   - **Benefit:** Grounding AI responses in external knowledge significantly reduces the occurrence of incorrect or irrelevant information.
   - **Source:** Retrieval augmented generation for large language models and...
4. **Up-to-date Information**
   - **Benefit:** RAG can incorporate the latest clinical data and guidelines, ensuring that the information provided is current and relevant.
   - **Source:** Retrieval augmented generation for large language models and...

#### Challenges of RAG in Medicine

1. **Lack of Standardized Evaluation Frameworks**
   - **Challenge:** There is a need for standardized frameworks to assess the performance and reliability of RAG systems.
   - **Source:** A Scoping Review of Retrieval-Augmented Generation—Ethics, Bias
2. **Ethical Concerns**
   - **Challenge:** Privacy and bias must be addressed to ensure the responsible deployment of RAG in healthcare.
   - **Source:** A Scoping Review of Retrieval-Augmented Generation—Ethics, Bias
3. **Complexity of Retrieval Mechanisms**
   - **Challenge:** Optimizing the retrieval mechanism to minimize irrelevant information is complex and requires advanced techniques.
   - **Source:** A Scoping Review of Retrieval-Augmented Generation—Ethics, Bias
4. **Missing Content and Hallucinations**
   - **Challenge:** RAG may still provide incorrect answers if relevant information is not available in the knowledge base.
   - **Source:** Top 7 Challenges with Retrieval-Augmented Generation - Valprovia
5. **Performance and Scalability**
   - **Challenge:** Handling large datasets and high query volumes efficiently is a significant challenge.
   - **Source:** Top Problems with RAG systems and ways to mitigate them
6. **Security Concerns**
   - **Challenge:** Data leakage and privacy risks are significant concerns that need to be addressed.
   - **Source:** Retrieval-augmented generation for generative artificial intelligence...

### Key Citations

- A Scoping Review of Retrieval-Augmented Generation—Ethics, Bias
- Enhancing medical AI with retrieval-augmented generation: A mini
- Generative AI in Healthcare: Use Cases, Benefits, and Challenges
- Retrieval augmented generation for large language models and...
- Top 7 Challenges with Retrieval-Augmented Generation - Valprovia
- Top Problems with RAG systems and ways to mitigate them
- Retrieval-augmented generation for generative artificial intelligence...

(b) Report Part (2/2)

Figure 8: Report Generation for query "*Give me a report of topic <RAG's Future in Medical Domain>*"

| Method | RACE | FACT (Eff.c.) | FACT (C.acc.) |
|---|---|---|---|
| RAG | 36.68 | 2.93 | 10.86 |
| ReACT | 42.21 | 4.69 | 23.78 |
| EDR | 40.99 | 12.28 | 23.70 |
| Ours | **43.60** | **20.42** | **32.34** |

Table 18: Results on DeepResearch Bench for multi-document report generation.

## N Computational Resources and Software Environment

Experiments were carried out on a server equipped with two Intel Xeon Platinum 8488C processors, comprising 96 physical cores and 192 hardware threads in total. The system was configured with two NVIDIA A800 GPUs, each providing 80 GB of GPU memory, and 503 GB of system RAM. All experiments were conducted on Ubuntu 22.04.5 LTS. The software environment was based on Python 3.11.11, with package management handled through Conda (version 23.5.2). Model implementation and training were performed using PyTorch 2.6.0 with CUDA support, together with HuggingFace Transformers 4.51.3 and SpaCy 3.8.4. Unless otherwise specified, all models and libraries were used with their default parameter settings. The total training time for our method was approximately 21,582 seconds under the above hardware configuration. During inference, each sample required an average of 50.91 seconds to process, with an average input length of 458.82 tokens per sample. The computational cost of training, data preprocessing, and inference varied depending on the experimental configuration and dataset scale, and all reported results were obtained under the same hardware and software setup.

## O The Use of Large Language Model

In this study, generative AI software tools were used solely as auxiliary support for language refinement, programming-related assistance, and figure visualization. Specifically, large language models were employed to improve grammatical accuracy, readability, and overall writing quality, as well as to provide general coding suggestions or debugging guidance. In addition, AI-based image generation tools were used to enhance the visual appearance of figure 1; however, the figure design, initial draft, and final verification were entirely performed by the authors. All AI-assisted outputs were carefully reviewed and validated by the authors prior to inclusion. The research conception, methodological design, and analysis of experimental results were conducted exclusively by the authors, and generative AI tools did not contribute to the formulation of research ideas or the derivation of conclusions.