RAG-R1: Incentivizing the Search and Reasoning Capabilities of LLMs Through Multi-query Parallelism

Zhiwen Tan[1], **Jiaming Huang**[1,2], **Qintong Wu**[1], **Hongxuan Zhang**[1,3], **Chenyi Zhuang**[1*], **Jinjie Gu**[1]
[1]Ant Group
[2]Zhejiang University
[3]Nanjing University
ender.tzw@antgroup.com, chenyi.zcy@antgroup.com

Abstract

Large Language Models (LLMs), despite their remarkable capabilities, are prone to generating hallucinated or outdated content due to their static internal knowledge. While Retrieval-Augmented Generation (RAG) integrated with Reinforcement Learning (RL) offers a solution, these methods are fundamentally constrained by a single-query mode, leading to prohibitive latency and inherent brittleness. To overcome these limitations, we introduce RAG-R1, a novel two-stage training framework centered around multi-query parallelism. Our framework enables LLMs to adaptively leverage internal and external knowledge during the reasoning process while transitioning from the single-query mode to multi-query parallelism. This architectural shift bolsters reasoning robustness while significantly reducing inference latency. Extensive experiments on seven question-answering benchmarks confirm the superiority of our method, which outperforms the strongest baseline by up to 13.7% and decreases inference time by 11.1%.

Code — https://github.com/inclusionAI/AWorld-RL/tree/main/RAG-R1

1 Introduction

Large Language Models (LLMs) (Zhao et al. 2023; Guo et al. 2025) have demonstrated remarkable capabilities in various domains, including mathematical reasoning, question answering, and code generation. However, the knowledge encoded in these models is static, limiting their adaptability. As a result, LLMs are susceptible to producing hallucinated or outdated responses (Ji et al. 2023; Shuster et al. 2021; Asai et al. 2024) when dealing with complex or real-time issues, compromising their reliability. Therefore, it is essential to equip LLMs with access to external knowledge to ensure more accurate and grounded responses.

Retrieval-Augmented Generation (RAG) (Gao et al. 2023; Fan et al. 2024) is a widely adopted approach to address this issue, broadening the model's capability boundaries by integrating external knowledge into the generation process. Early efforts (Shao et al. 2023; Ram et al. 2023a) in this area concentrated on prompt-based strategies that guide LLMs through question decomposition, query rewriting and multi-turn retrieval. While effective, these approaches are constrained by the limitations inherent in prompt engineering. Recent advances (Trivedi et al. 2022a; Li et al. 2025; Asai et al. 2024; Guan et al. 2025; Wang et al. 2025) have increasingly emphasized the integration of reasoning capabilities within RAG. These approaches combine RAG with Chain of Thought (CoT) for step-by-step retrieval, or automatically generate intermediate retrieval chains that incorporate reasoning and employ Supervised Fine-Tuning (SFT) to enable learning of retrieval and reasoning. However, new insights (Chu et al. 2025) indicate that these methods may cause models to memorize solution paths, thus constraining their generalization to novel scenarios.

Recently, reinforcement learning (RL) (Schulman et al. 2017) has demonstrated great potential in improving LLM performance by enhancing the reasoning capability. Reasoning models such as OpenAI-o1 (Jaech et al. 2024) and Deepseek-R1 (Guo et al. 2025) indicate that utilizing RL with outcome-based rewards can enhance the model's performance in mathematical and logical reasoning tasks. Within this paradigm, several works have explored enhancing the model's search ability during reasoning through RL. R1-Searcher (Song et al. 2025) proposes a novel two-stage outcome-based RL approach designed to enhance the search capability of LLMs. Search-R1 (Jin et al. 2025) introduces a novel RL framework that enables LLMs to interact with search engines in an interleaved manner with their own reasoning. Despite significant improvements, these RL-based methods struggle with stable training due to the restricted capabilities of cold-start models (Guo et al. 2025). Furthermore, existing methods generate only a single search query whenever external retrieval is required, which presents two significant challenges: (1) **Prohibitive Latency from Serial Execution.** The single-query architecture mandates a serial execution model for multi-hop reasoning, where each step must await the completion of the preceding retrieval-inference cycle. Consequently, latency accumulates with each turn, rendering the model impractical for latency-sensitive, real-world applications. (2) **Inherent Brittleness and Knowledge Confinement.** The single-query approach is inherently brittle, as its entire reasoning trajectory depends on the success of each sequential step. An early, suboptimal query can lock the model into an unrecoverable path—a

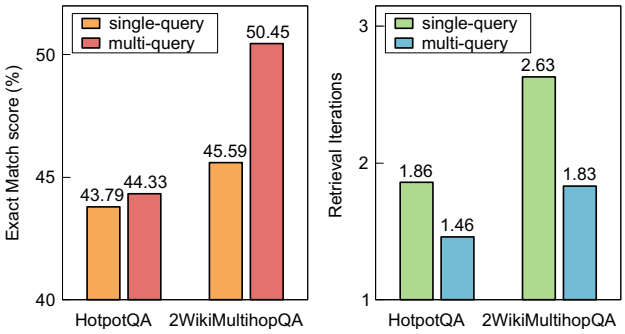
*Corresponding author.

critical failure mode. This single-threaded process confines the model to a narrow slice of external knowledge, leaving it unable to navigate user ambiguity, explore alternative reasoning paths, or recover from an initial misstep. This fragility ultimately compromises its reasoning robustness and final performance. We conducted a straightforward experiment based on Qwen2.5-72B-Instruct (Team 2024) following Jin et al. (2025) to validate the aforementioned challenges. As illustrated in Figure 1, we evaluated the model's performance and average retrieval count on two Multi-Hop Question Answering benchmarks: HotpotQA (Yang et al. 2018) and 2WikiMultiHopQA (Ho et al. 2020), comparing the generation of a single query to multiple queries in scenarios necessitating retrieval. The findings indicate that the multi-query method enhances the model's performance and decreases retrieval iterations compared to the single-query approach, highlighting the limitations inherent in the single-query mode.

To address the aforementioned challenges, we propose RAG-R1, a novel training framework that enables LLMs to adaptively leverage internal and external knowledge during the reasoning process and enhances their reasoning capabilities. We further expand the single-query mode to multi-query parallelism, an architectural shift that directly addresses prohibitive latency and inherent brittleness, thereby bolstering reasoning robustness and reducing inference time. Specifically, the training framework contains two stages, i.e., Format Learning Supervised Fine-Tuning and Retrieval-Augmented Reinforcement Learning. In the first stage, we thoughtfully generate samples that integrate reasoning and search to equip LLMs with the ability to adaptively leverage internal and external knowledge during the reasoning process and respond in a think-then-search format. In the second stage, we employ outcome-based RL with a retrieval environment to enhance the model's ability to reason effectively and dynamically access external knowledge for accurate question answering. Building upon the training framework, we transition from the single-query mode to multi-query parallelism. By minimizing serial retrieval iterations, this approach drastically reduces overall inference time. Simultaneously, it arms the model with comprehensive evidence from multiple perspectives, directly bolstering its robustness against reasoning failures.

We conduct extensive experiments based on Qwen2.5-7B-Instruct (Team 2024) to verify the effectiveness of our proposed method. The results demonstrate its effectiveness, which achieves state-of-the-art performance on seven question-answering benchmarks. In particular, our method utilizing multiple-query parallelism outperforms the strongest baseline by up to 13.7% and decreases inference time by 11.1%. The contributions of this work are summarized as follows:

- We propose RAG-R1, a novel training framework that empowers LLMs to adaptively leverage internal and external knowledge during the reasoning process and enhances their reasoning capabilities.
- We transition from the single-query mode to multi-query parallelism to directly address prohibitive latency and inherent brittleness. This architectural shift bolsters reasoning robustness while significantly reducing inference time.
- Extensive experiments demonstrate the effectiveness of our proposed method, which achieves state-of-the-art performance on seven question-answering benchmarks and significantly decreases retrieval counts and inference time.



(a) Exact Match score (b) Retrieval Iterations

Figure 1: Comparison of single-query and multi-query methods on Multi-Hop QA benchmarks based on Qwen2.5-72B-Instruct: (a) Model performance evaluated by Exact Match metric; (b) Average retrieval iterations during inference. The multi-query approach achieves a higher Exact Match score with fewer retrieval iterations.

2 Related Work

Retrieval-Augmented Generation RAG enhances LLMs outputs by integrating external information with internal knowledge. Initial RAG methods employed static pipelines, coupling retrieved documents with LLMs via prompt engineering (Shi et al. 2023; Ram et al. 2023b; Lin et al. 2023; Korikov et al. 2024) to control generation. While effective for information-sourcing tasks, these approaches struggle with complex, multi-faceted queries requiring iterative retrieval. Subsequent research developed dynamic RAG frameworks (Jeong et al. 2024) that utilize LLM feedback for adaptive retrieval and generation control. Complementary efforts have focused on improving knowledge representation of retrieved documents (Edge et al. 2024; Guo et al. 2024) from augmentation perspectives. Most recently, advances (Jin et al. 2025; Li et al. 2025; Song et al. 2025; Sun et al. 2025; Wu et al. 2025) demonstrate significant scalability potential by training interaction trajectories to optimize LLM's use of search engines for complex problem-solving queries. Following these advances, our method further enhances the interaction between LLMs and retrievers by leveraging LLM's reasoning abilities.

Learning to Search Recent advances in Learning to Search have evolved from static query and generation template via SFT to dynamic and reward-driven search using RL. SFT-based approaches have demonstrated success in improving LLM capabilities in instruction following (Dong et al. 2024b,a; Li et al. 2024), robustness (Dong et al. 2025),

and domain-specific adaptation (Zhang et al. 2024). However, these methods are limited by a tendency to memorize solution paths, which constrains generalization and scalability. RL-based methods (Jaech et al. 2024; Guo et al. 2025; Shao et al. 2024; Yang et al. 2025) offer a promising alternative by enabling autonomous reasoning and decision-making. Recent works (Li et al. 2025; Song et al. 2025; Jin et al. 2025) further integrate LLMs into real-time search workflows, allowing interaction with live search engines. Although these methods support autonomous search engine invocation, the efficacy of search-as-a-tool remains underexplored. To address this, our approach introduces a two-stage training framework that enables parallel query generation, with the aim of retrieving more comprehensive and diverse document contexts.

3 Training Framework

In this section, we introduce the RAG-R1 training framework, which aims to empower LLMs with the ability to adaptively leverage internal and external knowledge during reasoning through two stages, i.e., Format Learning Supervised Fine-Tuning and Retrieval-Augmented Reinforcement Learning. Specifically, in the first stage, we thoughtfully generate samples that integrate reasoning and search, segmenting them into four categories. We then apply SFT based on these segmented samples to equip LLMs with the ability to generate responses in a think-then-search format and leverage internal and external knowledge adaptively. In the second stage, we conduct data selection and employ outcome-based RL with a retrieval environment to enhance the model's ability to reason and dynamically retrieve external knowledge to answer questions correctly. The overall framework is shown in Figure 2.

3.1 Format Learning Supervised Fine-Tuning

SFT Data Generation To equip LLMs with the ability to adaptively leverage internal and external knowledge during reasoning and respond in a think-then-search format, we first generate samples that integrate reasoning and search. Specifically, the system instruction guides the LLM to perform reasoning between <**think**> and </**think**> whenever it receives new information. After completing the reasoning process, the LLM can generate search query between the designated search call tokens, <**search**> and </**search**>, whenever external retrieval is necessary. The system will subsequently extract the search query and request an external search engine to retrieve relevant documents. The retrieved information is then appended to the existing sequence, enclosed within special retrieval tokens, <**information**> and </**information**>, providing additional context for the next generation step. This process continues iteratively until reaching the maximum number of retrieval or the model generates a final answer enclosed within the special answer tokens, <**answer**> and </**answer**>. The system instruction follows Jin et al. (2025) and is shown in Table 1.

We use Qwen2.5-72B-Instruct as the generation model and employ a portion of the HotpotQA (Yang et al. 2018) training dataset for generation tasks.

Answer the given question. You must conduct reasoning inside <think> and </think> first every time you get new information. After reasoning, if you find you lack some knowledge, you can call a search engine by <search> query </search> and it will return the top searched results between <information> and </information>. You can search as many times as your want. If you find no further external knowledge needed, you can directly provide the answer inside <answer> and </answer>, without detailed illustrations. For example, <answer> Beijing </answer>. Question: {question}

Table 1: System instruction of SFT data generation.

SFT Data Segmentation and Training After generating the samples, we selected those with correct answers and segment them into four categories according to specific segmentation points. As illustrated in Figure 2, we segment the samples at corresponding points whenever the model needs to perform reasoning or retrieval, thereby preventing the model from generating retrieval documents. In this manner, a complete sample might be split into multiple smaller samples, which can then be classified into four distinct categories.

By employing the first and second categories of samples, we train the model to respond by adaptively leveraging internal and external knowledge, respectively. In contrast, by utilizing the third and fourth categories of samples, we expect the model to perform reasoning and generate subsequent steps primarily based on external knowledge. Notably, the output section of the samples is designed to exclude retrieved documents, which aids in preventing the model from generating hallucinations.

After segmentation, we apply SFT to these samples to train the model to generate responses in a think-then-search format and leverage internal and external knowledge adaptively. This stage aims to develop a highly capable model that can respond in a think-then-search format, serving as a cold-start model to enhance the stability of the subsequent RL training stage.

3.2 Retrieval-Augmented Reinforcement Learning

RL Data selection After Format Learning SFT, we obtain a model that can adaptively leverage internal and external knowledge during the reasoning process and respond in a think-then-search format. To further enhance the model's reasoning ability and enable it to answer questions accurately, we begin with data selection to identify challenging yet answerable questions suitable for RL.

Specifically, we select the samples that generate incorrect answers in Section 3.1. The questions of these samples present a relative challenge to the model. We subsequently filter out samples that are inherently unanswerable due to incomplete data retrieval or model limitations. We implement stochastic sampling for Qwen2.5-72B-Instruct with the sampling temperature of 1.2 and the maximum number of retrieval to 10. For each question, we conduct up to 10 rollouts

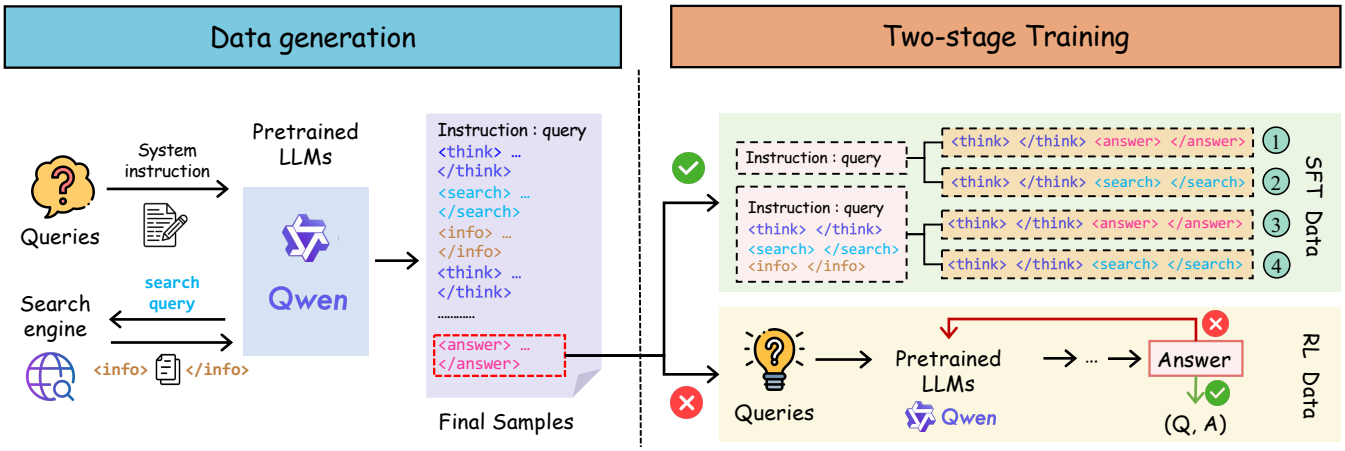


Figure 2: Overview of the RAG-R1 training framework, consisting of two stages: Format Learning Supervised Fine-Tuning (Section 3.1) and Retrieval-Augmented Reinforcement Learning (Section 3.2), along with the data generation details.

and retain only samples containing at least one correct solution. Finally, we obtain 2488 samples which are challenging yet answerable and randomly selected 25% of the correctly answered samples in Section 3.1 to construct the final training dataset.

RL with Retrieval We extend reinforcement learning to utilize an external retrieval system. The RL objective function utilizing an external retrieval system $\mathcal{R}$ can be represented as follows:

$$\max_{\pi_\theta} \mathbb{E}_{x\sim D, y\sim\pi_\theta(\cdot|x;\mathcal{R})}[r_\phi(x,y)] - \beta\mathbb{D}_{KL}[\pi_\theta(y|x;\mathcal{R})||\pi_{ref}(y|x;\mathcal{R})]$$

where π_θ and π_{ref} represent the policy model and the reference model, respectively, both of which are initialized from the SFT model, r_ϕ is the reward function and $\mathbb{D}_{KL}$ is KL-divergence. x denote input samples drawn from the dataset D, and y represent the generated outputs which is sampled from the policy model π_θ and retrieved from the retrieval system $\mathcal{R}$. The rollout process follows the same procedure detailed in Section 3.1. We employ the Proximal Policy Optimization (PPO) (Schulman et al. 2017) algorithm to optimize the policy model, which is widely used in reinforcement learning due to its efficiency and reliability.

Retrieval Masked Loss. In our framework, the rollout sequence consists of both LLM-generated tokens and retrieved tokens from the external retrieval system. To prevent retrieved tokens from interfering with the inherent reasoning and generation capabilities of the model, we implement a masked loss for these tokens. By computing the policy gradient objective exclusively on the LLM-generated tokens and excluding the retrieved content from optimization, this approach stabilizes training while preserving the adaptability and benefits of retrieval-augmented generation.

Reward Modeling. The reward function serves as the primary training signal, which decides the optimization direction of RL. Inspired by Guo et al. (2025), we adopt a rule-based reward system that includes final answer rewards to assess the accuracy of responses. For instance, in factual reasoning tasks, we adopt rule-based criteria such as exact string matching (EM) to evaluate correctness:

$$r_\phi(x,y) = EM(a_{pre}, a_{gold})$$

where a_{pre} represents the extracted final answer from response y and a_{gold} denotes the ground truth answer. This method focuses on ensuring that the final answer matches the expected truth precisely. We avoid incorporating format rewards because the SFT model has already learned to respond in a think-then-search format. This strategy enables us to concentrate on the factual accuracy of the responses, leveraging the inherent structured approach of the model without imposing additional format-based conditions. Furthermore, following Guo et al. (2025), we do not apply neural reward models to avoid reward hacking and additional computational cost.

Utilizing the SFT model, which can adaptively leverage internal and external knowledge, as the cold-start model improves the stability of RL training. Furthermore, by incorporating retrieval into outcome-based RL, we improve the model's reasoning capability and its ability to dynamically access external knowledge to accurately answer questions.

4 Multi-query Parallelism

Enabling the model to adaptively leverage internal and external knowledge during reasoning significantly extends its capability boundaries. However, existing methods are constrained to a single-query approach, which presents two fundamental weaknesses: prohibitive latency from serial retrieval iterations, and inherent brittleness due to knowledge confinement. To overcome these limitations, we replace this fragile, sequential process with multi-query parallelism. This architectural shift is designed not merely to enhance capabilities, but to directly tackle the root causes of failure—it reduces latency by parallelizing retrieval and fortifies robustness by gathering diverse evidence simultaneously.

Specifically, we guide the model to generate multiple search queries whenever external retrieval is required. The

external retrieval system then performs parallel searches with these queries and returns the results to the model in a JSON format, ensuring clear alignment between the search queries and the retrieved documents. The adoption of multi-query parallelism directly confronts the two fundamental weaknesses of the single-query approach: prohibitive latency and inherent brittleness. First, by executing retrievals in parallel, it dismantles the serial dependency that causes prohibitive latency, leading to a substantial reduction in overall inference time. Second, it shatters the knowledge confinement of a single query by gathering diverse evidence simultaneously. This enriched information base fortifies the model's reasoning process, dramatically enhancing its robustness against suboptimal queries and unrecoverable paths. This dual improvement is particularly salient for complex tasks like comparison-type questions, where synthesizing multiple perspectives is crucial for accuracy.

Table 2 illustrates the system instructions for generating multiple search queries. We restrict the model to generate at most three parallel search queries simultaneously. The retrieval system reorganized the search results in a JSON format which contains:

- **query**: A list of search queries generated by the model.
- **documents**: A list containing the at most three retrieved documents corresponding to the search queries.

Utilizing the above format allows the model to recognize the alignment between difference search queries and corresponding retrieved documents. The remaining aspects of the training process are consistent with those outlined in Section 3.

Answer the given question. You must conduct reasoning inside <think> and </think> first every time you get new information. After reasoning, if you find you lack some knowledge, you can call a search engine by <search> query_1,query_2 </search> and it will return the top searched results for each query between <information> and </information>. You can search as many times as your want, using up to three queries each time. If you find no further external knowledge needed, you can directly provide the answer inside <answer> and </answer>, without detailed illustrations. For example, <answer> Beijing </answer>. Question: {question}

Table 2: System instruction for generating multiple search queries.

5 Experiments

5.1 Experimental Settings

Datasets and Evaluation Metrics. We evaluate our models on seven benchmark datasets: (1) **General Question Answering**: NQ (Kwiatkowski et al. 2019), TriviaQA (Joshi et al. 2017), and PopQA (Mallen et al. 2022). (2) **Multi-Hop Question Answering**: HotpotQA (Yang et al. 2018), 2WikiMultiHopQA (Ho et al. 2020), Musique (Trivedi et al. 2022b), and Bamboogle (Press et al. 2022). HotpotQA is in-domain benchmark since a portion of its training sets is used for training while others serve as out-of-domain benchmarks to assess our model's generalization capabilities. For evaluation metric, we utilize Exact Match (EM) score following Yu et al. (2024) and Retrieval Count (RC).

Baselines. We compare our method against the following baselines: (1) **Direct Inference**: Generate answers based on the model's internal knowledge. (2) **Standard RAG**: Traditional retrieval-augmented generation systems. (3)**RAG-CoT Methods**: Integration of retrieval-augmented generation with chain-of-thought prompts, such as IRCoT (Trivedi et al. 2022a) and Search-o1 (Li et al. 2025). (4) **RAG-RL Methods**: Utilizing reinforcement learning to allow the model to autonomously perform retrieval during inference, such as Search-R1 (Jin et al. 2025) and R1-Searcher (Song et al. 2025).

Implementation Details. We adopt Qwen-2.5-7B-Instruct as the base model for both our method and all baseline approaches, ensuring a fair comparison under the same architectural backbone. For retrieval, we use the English Wikipedia provided by KILT (Petroni et al. 2020) as retrieval corpus, segmented into 100-word passages with appended titles, totaling 29 million passages. We employ BGE-large-en-v1.5 (Chen et al. 2024) as the text retriever and set the number of retrieved passages to 3 across all retrieval-based methods to ensure fair comparison. For evaluation, the number of retrieval iterations was left uncapped.

In Format Learning SFT, we collect 18994 samples for single-query training and 19303 samples for multi-query parallelism training. We perform full-parameter SFT for 5 epochs and chose the checkpoint with the lowest validation loss for evaluation and subsequent RL training.

In Retrieval-Augmented RL, we collect 5022 samples for single-query training and 6015 samples for multi-query parallelism training. We split 95% of all the samples into a training set and used the remaining as a validation set. For the PPO variant, we set the learning rate of the policy LLM to 1e-6 and that of the value LLM to 1e-5. The training step is 500, with warm-up ratios of 0.285 and 0.015 for the policy and value models, respectively. We use Generalized Advantage Estimation (GAE) (Schulman et al. 2015) with parameters $\lambda = 1$ and $\gamma = 1$. All the trainings are conducted on a single node with 8 A100 GPUs. We configure the batch settings as follows: a total batch size of 512, a mini-batch size of 128, and a micro-batch size of 32.

5.2 Main results

The main results comparing RAG-R1 with baseline methods across the seven datasets are presented in Table 3. *RAG-R1-sq* denotes our method operating in single-query mode while *RAG-R1-mq* represents the overall framework incorporating multi-query parallelism. From the results, we can obtain the following key observations:

- **Significant performance improvements across all datasets.** Our method achieves significant improvements compared to all baseline methods across both General QA and mutil-hop QA benchmarks, including both

Methods	General QA			Multi-Hop QA				
	NQ	PopQA	TriviaQA	HotpotQA	2Wiki	Musique	Bamboogle	Avg.
Direct Inference	0.132	0.148	0.360	0.183	0.236	0.031	0.080	0.167
Standard RAG	0.328	0.353	0.476	0.284	0.253	0.048	0.152	0.271
IRCoT	0.183	0.328	0.434	0.276	0.356	0.060	0.144	0.254
Search-o1	0.277	0.294	0.474	0.348	0.384	0.107	0.296	0.311
Search-R1	0.387	0.422	0.531	0.377	0.351	0.135	0.376	0.368
R1-Searcher	0.404	0.410	0.522	0.442	0.513	0.158	0.368	0.402
RAG-R1-sq	**0.429**	0.477	0.599	0.492	0.520	0.187	**0.440**	0.449
RAG-R1-mq	0.423	**0.479**	**0.608**	**0.495**	**0.563**	**0.192**	**0.440**	**0.457**

Table 3: Performance comparisons on QA benchmarks under the Exact Match metric. The best and second best results are **bold** and underlined, respectively. Our models, RAG-R1-sq and RAG-R1-mq, achieve substantial improvements over all baselines

CoT-based methods and RL-based methods. Specifically, RAG-R1-mq outperforms the best RL-based method, R1-Searcher, by 13.7% across all datasets. These results demonstrate that our approach enables the model to effectively utilize both internal and external knowledge throughout the reasoning process.

- **Effectiveness of Multi-query Parallelism.** RAG-R1-mq consistently outperforms all single-query RL-based methods, notably achieving a 1.8% average gain over our own single-query counterpart, RAG-R1-sq. This result reveals a key insight: multi-query parallelism is not merely an efficiency optimization, but a direct remedy for the inherent brittleness of single-query methods. By shattering knowledge confinement, it empowers the model to synthesize diverse evidence, fundamentally bolstering its reasoning robustness against failures. Further details are available in Section 6.2.
- **Excellent generalization Ability.** Despite utilizing only a subset of the HotpotQA training data, our models achieve significant improvements on in-domain datasets and exhibit excellent generalization capabilities across out-of-domain datasets, such as 2WikiMultiHopQA and Musique. This demonstrates that our models have effectively learned to reason and retrieve information for diverse questions, which proves the effectiveness of our approach across various scenarios requiring retrieval. Moreover, it can effectively extend to online search, as detailed in Section 6.3.

6 Further Analysis

6.1 Ablation Study

To validate the effectiveness of our proposed training framework, we conduct a comprehensive ablation analysis of its key design elements.

SFT and RL As shown in Table 4, *w/o SFT* removes the initial format learning SFT while *w/o RL* removes the entire RL training stage. The results demonstrate the necessity and effectiveness of both SFT and RL in our training framework, which together improve the model's performance. Specifically, *w/o SFT* struggles to leverage internal and external knowledge and respond in a think-then-search format, resulting in decreased performance. In contrast, *w/o RL* restricts the model's capability to correctly answer questions through reasoning. The considerable improvement achieved through RL highlight its ability to significantly enhance the model's capability.

RL Data Selection We further validate the effectiveness of data selection during the RL process. We trained the models using all samples with incorrect answers and 25% of the samples with correct answers in Section 3.1, denoted as RAG-R1-sq *w/o Filter* and RAG-R1-mq *w/o Filter* respectively. The decline in performance indicates that careful data selection plays a crucial role in enhancing the effectiveness of RL training. Unanswerable samples do not facilitate the model's improvement and may even detrimentally impact the training process.

6.2 Effects of Multi-query Parallelism

As shown in Table 5, we record the average inference time (in seconds) and average retrieval iterations for different methods on HotpotQA and 2WikiMultiHopQA using A100 GPUs without employing any inference acceleration technique. The results show that RAG-R1-mq achieves the lowest inference time and requires the fewest retrieval iterations, indicating that multi-query parallelism can significantly enhance the efficiency of the model. As shown in Table 3, our multi-query approach (RAG-R1-mq) confirms its superior effectiveness. A detailed analysis against our single-query baseline (RAG-R1-sq) reveals this performance gain stems from two fundamental advantages: (1) **More Efficient and Decisive Exploration**: The MQ model's parallel retrieval acts as a wide-ranging initial search, quickly identifying promising information pathways. This prevents the fruitless retrieval loops or premature terminations that plague the SQ model, which often lead to unrecoverable errors. (2) **Adaptive Reasoning and Self-Correction**: When initial retrievals fail, the MQ model demonstrates a crucial capability for self-correction. It synthesizes the complete context from all parallel queries to intelligently formulate a more informed new query. In contrast, the SQ model, locked into its linear path, lacks this adaptive mechanism, rendering it brittle to an initially suboptimal query. In essence, multi-query parallelism fosters a more robust and adaptive reason-

Methods	HotpotQA	2Wiki	Musique	Bamboogle	Avg.
RAG-R1-sq	0.492	0.520	0.187	**0.440**	0.410
RAG-R1-sq *w/o SFT*	0.415	0.406	0.138	0.312	0.318
RAG-R1-sq *w/o RL*	0.425	0.433	0.150	0.368	0.344
RAG-R1-sq *w/o Filter*	0.452	0.462	0.159	0.408	0.370
RAG-R1-mq	**0.495**	**0.563**	**0.192**	**0.440**	**0.423**
RAG-R1-mq *w/o SFT*	0.413	0.423	0.129	0.392	0.339
RAG-R1-mq *w/o RL*	0.427	0.490	0.143	0.424	0.371
RAG-R1-mq *w/o Filter*	0.491	0.543	0.186	0.424	0.411

Table 4: Ablation study on SFT, RL and RL Data Selection. The RL-only method (w/o SFT) struggles to respond in a think-then-search format, while the SFT-only method (w/o RL) fails to enhance model performance through reasoning. Moreover, the performance decline observed in the w/o Filter method emphasizes the necessity of RL data selection.

Methods	HotpotQA		2Wiki		Avg.	
	Time	RI	Time	RI	Time	RI
Search-R1	7.79	2.44	8.90	3.01	8.35	2.73
R1-Searcher	10.98	2.31	10.93	**2.40**	10.96	2.36
RAG-R1-sq	7.69	2.14	8.72	2.72	8.21	2.43
RAG-R1-mq	**6.72**	**1.89**	**8.11**	2.43	**7.42**	**2.16**

Table 5: Average time and retrieval iterations of different methods on two multi-hop datasets. RAG-R1-mq reduces the inference time by 9.6% and the average retrieval iterations by 0.27 compared to RAG-R1-sq, demonstrating improved efficiency in multi-hop reasoning.

ing process, expanding the model's problem-solving capabilities beyond mere efficiency gains.

6.3 Generalization to Online Search

Considering training efficiency and cost, we employ a local dense embedding retrieval system utilizing Wikipedia as a static external retrieval corpus throughout the training process. This contrasts with most real-world applications, which depend on online web search. To demonstrate the generalization capability of RAG-R1 within this scenario, we assess our models' performance on two benchmarks: Bamboogle and a randomly selected set of 500 samples from HotpotQA, utilizing online web search—a setting not encountered during training.

Specifically, during inference, whenever retrieval is necessary, we leverage the Google API to perform real-time web searches and obtain relevant web pages. Subsequently, we utilize BeautifulSoup4 to scrape information from these pages and employ GPT-4o-mini to generate concise summaries, which are then incorporated into the reasoning process. As illustrated in Figure 3, RAG-R1-mq achieves the best EM score among all compared methods, highlighting its strong adaptability to online search scenario. These findings suggest that our approach equips the model to dynamically retrieve information during the reasoning process, rather than merely memorizing response formats.

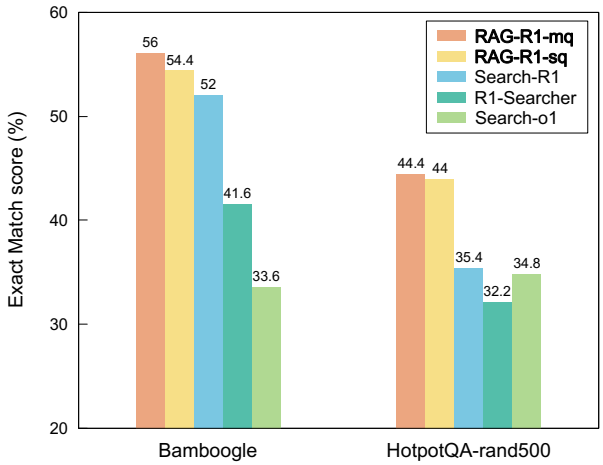


Figure 3: Performance comparison of RAG-R1 and three baselines within the online search scenario. Our models consistently deliver robust results across both offline and online settings, highlighting the strong generalization capabilities of our approach.

7 Conclusion

In this paper, we introduced RAG-R1, a novel training framework that enables LLMs to adaptively leverage internal and external knowledge, significantly enhancing their reasoning capabilities. The cornerstone of our work is the integration of multi-query parallelism, an architectural innovation designed to directly address the prohibitive latency and inherent brittleness of conventional single-query methods, thereby bolstering reasoning robustness and reducing inference time. Extensive experiments on seven QA benchmarks demonstrate the effectiveness of our method, which outperforms the strongest baseline by up to 13.7% and decreases inference time by 11.1%. This dual advancement confirms that our method achieves a superior trade-off, simultaneously boosting the model's reasoning robustness and inference efficiency.

References

Asai, A.; Wu, Z.; Wang, Y.; Sil, A.; and Hajishirzi, H. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection.

Chen, J.; Xiao, S.; Zhang, P.; Luo, K.; Lian, D.; and Liu, Z. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Chu, T.; Zhai, Y.; Yang, J.; Tong, S.; Xie, S.; Schuurmans, D.; Le, Q. V.; Levine, S.; and Ma, Y. 2025. Sft memorizes, rl generalizes: A comparative study of foundation model post-training. *arXiv preprint arXiv:2501.17161*.

Dong, G.; Lu, K.; Li, C.; Xia, T.; Yu, B.; Zhou, C.; and Zhou, J. 2024a. Self-play with execution feedback: Improving instruction-following capabilities of large language models. *arXiv preprint arXiv:2406.13542*.

Dong, G.; Song, X.; Zhu, Y.; Qiao, R.; Dou, Z.; and Wen, J.-R. 2024b. Toward general instruction-following alignment for retrieval-augmented generation. *arXiv preprint arXiv:2410.09584*.

Dong, G.; Zhu, Y.; Zhang, C.; Wang, Z.; Wen, J.-R.; and Dou, Z. 2025. Understand what LLM needs: Dual preference alignment for retrieval-augmented generation. In *Proceedings of the ACM on Web Conference 2025*, 4206–4225.

Edge, D.; Trinh, H.; Cheng, N.; Bradley, J.; Chao, A.; Mody, A.; Truitt, S.; Metropolitansky, D.; Ness, R. O.; and Larson, J. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Fan, W.; Ding, Y.; Ning, L.; Wang, S.; Li, H.; Yin, D.; Chua, T.-S.; and Li, Q. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, 6491–6501.

Gao, Y.; Xiong, Y.; Gao, X.; Jia, K.; Pan, J.; Bi, Y.; Dai, Y.; Sun, J.; Wang, H.; and Wang, H. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2(1).

Guan, X.; Zeng, J.; Meng, F.; Xin, C.; Lu, Y.; Lin, H.; Han, X.; Sun, L.; and Zhou, J. 2025. DeepRAG: Thinking to Retrieve Step by Step for Large Language Models. *arXiv preprint arXiv:2502.01142*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Guo, Z.; Xia, L.; Yu, Y.; Ao, T.; and Huang, C. 2024. Lightrag: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*.

Ho, X.; Nguyen, A.-K. D.; Sugawara, S.; and Aizawa, A. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Jaech, A.; Kalai, A.; Lerer, A.; Richardson, A.; El-Kishky, A.; Low, A.; Helyar, A.; Madry, A.; Beutel, A.; Carney, A.; et al. 2024. Openai o1 system card. *arXiv preprint arXiv:2412.16720*.

Jeong, S.; Baek, J.; Cho, S.; Hwang, S. J.; and Park, J. C. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. *arXiv preprint arXiv:2403.14403*.

Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y. J.; Madotto, A.; and Fung, P. 2023. Survey of hallucination in natural language generation. *ACM computing surveys*, 55(12): 1–38.

Jin, B.; Zeng, H.; Yue, Z.; Yoon, J.; Arik, S.; Wang, D.; Zamani, H.; and Han, J. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Joshi, M.; Choi, E.; Weld, D. S.; and Zettlemoyer, L. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Korikov, A.; Saad, G.; Baron, E.; Khan, M.; Shah, M.; and Sanner, S. 2024. Multi-aspect reviewed-item retrieval via LLM query decomposition and aspect fusion. *arXiv preprint arXiv:2408.00878*.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466.

Li, X.; Dong, G.; Jin, J.; Zhang, Y.; Zhou, Y.; Zhu, Y.; Zhang, P.; and Dou, Z. 2025. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*.

Li, X.; Jin, J.; Zhou, Y.; Wu, Y.; Li, Z.; Ye, Q.; and Dou, Z. 2024. Retrollm: Empowering large language models to retrieve fine-grained evidence within generation. *arXiv preprint arXiv:2412.11919*.

Lin, K.; Lo, K.; Gonzalez, J. E.; and Klein, D. 2023. Decomposing complex queries for tip-of-the-tongue retrieval. *arXiv preprint arXiv:2305.15053*.

Mallen, A.; Asai, A.; Zhong, V.; Das, R.; Khashabi, D.; and Hajishirzi, H. 2022. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. *arXiv preprint arXiv:2212.10511*.

Petroni, F.; Piktus, A.; Fan, A.; Lewis, P.; Yazdani, M.; De Cao, N.; Thorne, J.; Jernite, Y.; Karpukhin, V.; Maillard, J.; et al. 2020. KILT: a benchmark for knowledge intensive language tasks. *arXiv preprint arXiv:2009.02252*.

Press, O.; Zhang, M.; Min, S.; Schmidt, L.; Smith, N. A.; and Lewis, M. 2022. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.

Ram, O.; Levine, Y.; Dalmedigos, I.; Muhlgay, D.; Shashua, A.; Leyton-Brown, K.; and Shoham, Y. 2023a. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11: 1316–1331.

Ram, O.; Levine, Y.; Dalmedigos, I.; Muhlgay, D.; Shashua, A.; Leyton-Brown, K.; and Shoham, Y. 2023b. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11: 1316–1331.

Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; and Abbeel, P. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shao, Z.; Gong, Y.; Shen, Y.; Huang, M.; Duan, N.; and Chen, W. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Shi, W.; Min, S.; Yasunaga, M.; Seo, M.; James, R.; Lewis, M.; Zettlemoyer, L.; and Yih, W.-t. 2023. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Shuster, K.; Poff, S.; Chen, M.; Kiela, D.; and Weston, J. 2021. Retrieval augmentation reduces hallucination in conversation. *arXiv preprint arXiv:2104.07567*.

Song, H.; Jiang, J.; Min, Y.; Chen, J.; Chen, Z.; Zhao, W. X.; Fang, L.; and Wen, J.-R. 2025. R1-searcher: Incentivizing the search capability in llms via reinforcement learning. *arXiv preprint arXiv:2503.05592*.

Sun, H.; Qiao, Z.; Guo, J.; Fan, X.; Hou, Y.; Jiang, Y.; Xie, P.; Zhang, Y.; Huang, F.; and Zhou, J. 2025. Zerosearch: Incentivize the search capability of llms without searching. *arXiv preprint arXiv:2505.04588*.

Team, Q. 2024. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2022a. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. *arXiv preprint arXiv:2212.10509*.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2022b. MuSiQue: Multihop Questions via Single-hop Question Composition. *Transactions of the Association for Computational Linguistics*, 10: 539–554.

Wang, L.; Chen, H.; Yang, N.; Huang, X.; Dou, Z.; and Wei, F. 2025. Chain-of-Retrieval Augmented Generation. *arXiv preprint arXiv:2501.14342*.

Wu, W.; Guan, X.; Huang, S.; Jiang, Y.; Xie, P.; Huang, F.; Cao, J.; Zhao, H.; and Zhou, J. 2025. MASKSEARCH: A Universal Pre-Training Framework to Enhance Agentic Search Capability. *arXiv preprint arXiv:2505.20285*.

Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Yu, Y.; Ping, W.; Liu, Z.; Wang, B.; You, J.; Zhang, C.; Shoeybi, M.; and Catanzaro, B. 2024. Rankrag: Unifying context ranking with retrieval-augmented generation in llms. *Advances in Neural Information Processing Systems*, 37: 121156–121184.

Zhang, T.; Patil, S. G.; Jain, N.; Shen, S.; Zaharia, M.; Stoica, I.; and Gonzalez, J. E. 2024. Raft: Adapting language model to domain specific rag. *arXiv preprint arXiv:2403.10131*.

Zhao, W. X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).