Proceedings of the 4th International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/2026.36624

Advances in Reinforcement Learning for Retrieval-Augmented Generation in Large Language Model

Zunlong Hong

School of Computer Science and Technology, Xidian University, Xi'an, China
jjl040915@gmail.com

Abstract. Retrieval-augmented generation (RAG) enhances large language models (LLMs) by incorporating external information, but traditional fixed retrieval processes struggle to adapt to complex task requirements. In recent years, reinforcement learning (RL) has been increasingly applied to train LLMs to autonomously invoke search tools, driving RAG to evolve from the passive information acquisition of a fixed pipeline to a trustworthy retrieval system with autonomous decision-making capabilities. This paper reviews the representative studies on the combination of LLMs, RAG and RL in recent years. It focuses on analyzing the role of RL in dynamic retrieval, process rewards, query optimization, etc., and compares the connections and evolutionary relationships among different methods. The research findings show that RL has gradually expanded from simply improving the accuracy of the final answer to optimizing queries, multi-round search, process decision-making and trustworthy screening, providing new ideas for enhancing the active retrieval ability of RAG and improving the credibility of information.

Keywords: Large Language Models, Reinforcement Learning, Retrieval-Augmented Generation, Agent

1. Introduction

In recent years, large language models (LLMs), due to their strong language understanding and content generation capabilities, have been widely applied to tasks such as open-ended question answering, knowledge retrieval, and text generation. However, since the knowledge base of LLMs is mostly derived from the data used during the training phase, when external information changes or when users pose cutting-edge or highly specialized questions, the models may encounter problems such as outdated knowledge and factual errors. To address these issues, the Retrieval-Augmented Generation (RAG) technology is introduced. It retrieves relevant information from external knowledge bases or the network environment and provides the retrieved results as the context to the large language model.

Traditional RAG typically follows a fixed process of "user question - document retrieval - answer generation", and its main function is to supplement external knowledge for the model. However, due to the possibility of outdated content, low-quality information and misleading sources in the knowledge base or real web pages, the retrieved results may not all be relevant to the question. Therefore, the problems faced by the RAG research are not merely "whether information can be

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.36624

retrieved", but also include deciding when to conduct the retrieval, whether multiple rounds of retrieval are necessary, and judging the credibility of the information source.

To address these issues, researchers began to incorporate reinforcement learning (RL) into the RAG and search agents. Through training, the retrieval behavior of the model was optimized based on feedback signals such as query quality, search process, and final answer. The relevant studies respectively focus on aspects such as dynamic retrieval, multi-round web search, search process rewards, query optimization, and result evaluation. Early studies mainly focused on the selection of dynamic retrieval strategies, such as Adaptive-RAG which selects different retrieval modes based on the complexity of the question, thereby enhancing the adaptability of RAG to various tasks [1]. Self-RAG enhances the reliability of the RAG system by introducing a self-reflection mechanism, enabling the model to evaluate the retrieval requirements, document relevance, and generation quality [2]. Based on these, with the development of RL in large language model Agents, researchers further utilize reward signals to optimize the retrieval process. For instance, DeepResearcher and WebAgent-R1 use RL to train the model to complete search planning and multiple rounds of interaction in a real web environment [3, 4]. In addition, works such as StepSearch, DecoupleSearch and WebFilter further focus on the design of fine-grained rewards during the search process. They optimize the model's retrieval behavior by evaluating query quality, search trajectories, and the effectiveness of retrieval results [5-7]. However, the existing research still has certain limitations. On one hand, different methods mainly focus on optimizing a single aspect of the RAG process, lacking a systematic summary of the relationship among autonomous retrieval, process evaluation, and result optimization; on the other hand, RL in RAG still faces challenges in reward design, training cost, and generalization ability.

Regarding the aforementioned issue, this paper reviews the research on the combination of LLMs, RL, and retrieval-enhanced and agent in recent years. It compares the characteristics of different methods in terms of retrieval timing, search process, reward design, and trustworthy screening, and summarizes the evolution relationships and existing problems in each direction.

2. The application of reinforcement learning in RAG

Although the existing studies have respectively covered dynamic retrieval, web search, reward design and result evaluation, they essentially all aim to solve the same problem, that is, to achieve the transformation of retrieval from fixed tool invocation to a learnable decision-making process. The related work can be classified into three major directions: retrieval, process evaluation, and post-retrieval processing.

Traditional RAG usually follows a fixed "retrieval-generation" process, where a single retrieval is conducted before the generation stage. This makes it difficult to dynamically adjust the retrieval behavior according to the complexity of the task and the information requirements during the reasoning process. The arrival of agent-based methods changed this. ReAct gave LLMs the ability to call external tools and adapt their behavior to environmental feedback, and it became the basis for later autonomous retrieval agents [8]. Adaptive-RAG then explored choosing the retrieval strategy dynamically: it predicts question complexity and decides between direct generation, single-round retrieval, and multi-round retrieval, which gives the system more flexibility across different tasks [1]. Unlike methods that rely on supervised classification for dynamic retrieval, MBA-RAG frames the strategy choice as a multi-armed bandit problem and uses reinforcement learning to balance answer quality against retrieval cost [9]. However, these methods mainly rely on supervised learning or rule design, and it is difficult to continuously optimize the retrieval strategy based on the long-term task benefits. On this basis, researchers began to explore the use of RL to train RAG Agents

DOI: 10.54254/2755-2721/2026.36624

with autonomous search capabilities. Search-o1, as a typical dynamic retrieval framework, addresses the shortcomings of traditional static RAG, which can only conduct a one-time pre-search and is unable to respond to knowledge gaps during the long thinking chain reasoning process. By embedding Agentic RAG into the reasoning process of the large language model, the model can dynamically call the retrieval tool when encountering knowledge gaps, achieving an interactive process of alternating reasoning and retrieval [10]. It does not directly use RL to train the retrieval strategy, but has established an autonomous retrieval paradigm of "reasoning - retrieval - re-reasoning". Search-o1 mainly introduces two innovative modules. First, Intelligent search mechanism. It realizes a cycle of reasoning, retrieval, and re-reasoning, with the entire process being autonomously decided by the model and without the need for manual intervention in the retrieval timing. Second, the Reason-in-Documents module. This module directly purifies the information for the original content of the web pages returned by the search engine, which is lengthy and has a lot of noise. Compared with Search-o1 which directly focuses on lightweight retrieval during the middle stage of the thinking chain, DeepResearcher further extends this process to the real network. It trains the model through RL to complete search planning, web page access, and information verification [3]. In the real Web environment, DeepResearcher, in response to the difficulties of fixed corpus in reflecting web noise and dynamic changes, directly carried out end-to-end RL in real search engines [3]. By using RL to train the model, it completed web search, content extraction, and information verification, significantly enhancing the factual credibility of complex research tasks [3]. WebAgent-R1 pays more attention to the multi-round decision-making in continuous web interactions, enabling the model to adjust subsequent operations based on the current web state [4]. The core change in this stage is not an increase in the number of retrievals, but the transfer of retrieval control from manual processes to the model.

Evaluating the search process is also an important direction in this field. Before applying RL to optimize RAG, Self-RAG enhanced the model's ability to evaluate the retrieval process and generated results through a self-reflection mechanism, providing inspiration for subsequently using the reinforcement learning's reward signal to optimize the retrieval behavior [2]. StepSearch evaluates the intermediate results during the search process through distributed rewards [5]. It scores each search and each retrieval query separately, decomposes the complete multi-hop trajectory into independent and optimizable sub-steps, and evaluates different search steps using information gain and redundancy penalty. DecoupleSearch separates the search planning from the specific execution, and evaluates the two stages separately through hierarchical rewards [6]. WebFilter also focuses on reward design, but places greater emphasis on query accuracy and source credibility [7]. This method sets Source-restricting Reward (SR) to encourage the model to use advanced search methods such as source restriction, time restriction, and exact matching; at the same time, it sets Retrieval-precision Reward (RR) to evaluate whether these operations truly improve the retrieval quality [7]. Unlike WebFilter which mainly focuses on optimizing search behavior, R3-RAG further integrates retrieval decisions into the reasoning chain and optimizes the multi-step reasoning-retrieval interaction process through RL [11]. Furthermore, S3 also pays close attention to the issue of training costs in RL. By designing an effective reward mechanism, it reduces the reliance on large-scale search trajectory data and improves the training efficiency of the Search Agent [12].

RL is also widely applied in the post-processing stage after retrieval. REARANK models the re-ranking task as an MDP reinforcement learning problem, designs a reward mechanism based on ranking metrics, and optimizes the sorting of candidate documents [13]; DynamicRAG further utilizes the generation quality as feedback, enabling the re-ranker to simultaneously adjust the order and quantity of candidate documents, avoiding the problem of fixed Top-k that leads to the omission

DOI: 10.54254/2755-2721/2026.36624

of key information or excessive context noise [14]. RAG-Zeval builds a quality discrimination model using RL to improve the discrimination accuracy of RAG-generated answers [15]. The former two control which information enters the generation model, while the latter checks whether the generated content is supported by evidence, adding a quality control step beyond the optimization of the retrieval strategy.

3. Three development trends of RL-driven RAG

Reinforcement learning has now been applied at several stages of the RAG pipeline, and the work in this area can be grouped under three main trends.

The first trend concerns the search strategy. Traditional RAG runs a fixed, one-time retrieval pass, so the model cannot easily adjust its retrieval behavior to the task. Early studies treated retrieval as a decision problem and used RL to learn a good retrieval policy for different tasks; MBA-RAG, for example, balances retrieval effectiveness against computational cost through reward feedback [9]. With the rise of agent-based methods, the focus moved from choosing a retrieval policy to controlling the autonomous search process itself. Search-o1 embeds retrieval in the model's reasoning cycle, so the model decides on its own, based on its current reasoning state, whether and when to trigger a search [10]. DeepResearcher and WebAgent-R1 go further and move search into real web environments, training end to end so the model can issue multiple queries, collect information, and interact with the environment [3, 4]. Retrieval decisions in RAG have therefore grown from a single policy choice into the optimization of a complete search trajectory.

The second trend concerns the feedback signal, which has become more fine-grained over time. Classical RL settings reward the correctness of the final answer, but a RAG run is made up of many steps, and the final answer alone cannot tell us whether an individual query or retrieval action was effective. Later work has therefore designed finer-grained reward mechanisms. StepSearch assigns step-level rewards that score the information value of each single search action; DecoupleSearch separates search planning from search execution and rewards the two stages differently; and WebFilter takes both query quality and retrieval result quality into account, using multi-dimensional rewards to make search behavior more effective and more reliable [5-7]. In other words, the goal of RL has moved from obtaining a correct answer to learning the process that reliably produces a correct answer.

The third trend concerns the optimization objective, which has widened from question-answering accuracy to the credibility of the retrieved results. As RAG moves from closed knowledge bases to the open web, the retrieved results carry more noise, errors, and stale information, and these become the main constraints on system reliability. The central question has accordingly shifted from whether relevant information can be retrieved to whether the retrieved information can be trusted and whether the generated content is supported by evidence. REARANK uses RL training to re-rank the candidate documents after retrieval, DynamicRAG adjusts the selected context on the fly based on model feedback, and RAG-Zeval checks how well the generated answers agree with the retrieved evidence [13-15]. The overall direction is from autonomous retrieval toward trustworthy retrieval.

Taken together, the three trends point the same way. Retrieval changes from a fixed one-time pass to dynamic multi-round search; feedback changes from a reward on the final answer to process rewards that can evaluate intermediate queries and retrieval results; and the optimization objective changes from improving question-answering accuracy to ensuring source reliability and evidence fidelity. The result is a RAG system that moves from autonomous retrieval to trustworthy retrieval.

DOI: 10.54254/2755-2721/2026.36624

4. Challenges and prospects

4.1. Challenges

As RL is increasingly applied to RAG systems, the model's autonomous retrieval ability and information utilization ability have been enhanced. However, there are still several issues in the current research.

Firstly, the multimodal adaptation capability is insufficient. Currently, the methods driven by RL for RAG mainly focus on text knowledge retrieval. The state representation, action space, and reward function are mostly designed for text queries and text evidence. However, in practical applications, whether it is a private knowledge base or the real Internet web environment, the data usually appears in the form of text, images, and tables. How to enable the RL model to understand the relationships between different modalities of information and learn effective cross-modal retrieval strategies remain an important challenge in current research.

Secondly, there is still a deficiency in the transfer ability of the RAG model driven by RL. Currently, many studies mainly conduct reinforcement learning training based on small-scale language models to reduce training difficulty. However, whether the retrieval strategies and reward preferences learned in small models can effectively transfer to larger-scale models still lacks systematic verification. Due to the differences in reasoning ability, tool invocation ability, and context understanding ability among different-scale models, direct transfer may lead to strategy failure or performance degradation. Therefore, how to learn universal retrieval strategies independent of model size and improve the cross-model adaptability of RL optimized RAG frameworks is an important issue that needs to be focused on in the future.

Finally, the contradiction between performance improvement and computational cost. From the existing research, it can be observed that a reasonable architecture design and reward mechanism can reduce the reliance of RAG systems on large-scale language models, enabling some small-scale models (such as 3B, 7B) to achieve results close to those of larger models on specific tasks. However, the performance improvement brought by RL is usually accompanied by additional computational costs. During the training and actual deployment of RL, multiple rounds of search interactions, reward evaluation based on large models, and network latency caused by real web page access all introduce additional computational costs. Therefore, how to strike a balance between performance improvement in retrieval and system cost control is an important issue that needs to be addressed for the further application of reinforcement learning-driven RAG.

4.2. Prospects

In the future, the development of RAG driven by RL requires further improvement in the model's adaptability, generalization ability, and practical deployment efficiency. In the multi-modal direction, technologies such as visual language models can be combined to enable the RAG system to handle more diverse forms of knowledge and achieve cross-modal information retrieval and reasoning. In terms of model transfer, more general RL strategies need to be explored to ensure that the retrieval capabilities learned from training can adapt to different-sized models, different fields, and different application scenarios. In terms of efficiency optimization, future research should focus on the training and inference costs of RL, by designing more efficient reward mechanisms, reducing ineffective search processes, and exploring the collaborative methods of large and small models, in order to improve the retrieval quality while reducing system overhead. Overall, RL will drive RAG

DOI: 10.54254/2755-2721/2026.36624

to gradually evolve from single-task retrieval optimization to an intelligent retrieval system with autonomous decision-making, multi-modal understanding, and efficient deployment capabilities.

5. Conclusion

This paper reviews the research on the integration of LLMs, RL, and RAG in recent years. Current studies show that RAG is evolving from fixed, single-round document retrieval to dynamic retrieval, multi-round search, and real web page interaction. The role of RL has also expanded from optimizing the final answer to aspects such as search process, planning decisions, result re-ranking, and answer evaluation. At the same time, architectural innovations and specialized rewards have enabled small and medium-sized models to achieve task performance close to that of large models. However, existing methods still face limitations in adapting to multi-modal scenarios, unknown model transfer capabilities, and high application costs. In the future, with continuous breakthroughs in issues such as multimodal understanding, model transfer, and efficiency optimization, RL will further drive RAG towards a trustworthy, efficient, and universal intelligent retrieval system.

References

[1] Jeong, S., Baek, J., Cho, S., Hwang, S. J., & Park, J. C. (2024). Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)* (pp. 7036-7050).

[2] Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2024). Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations* (Vol. 2024, pp. 9112-9141).

[3] Zheng, Y., Fu, D., Hu, X., Cai, X., Ye, L., Lu, P., & Liu, P. (2025). Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 414-431).

[4] Wei, Z., Yao, W., Liu, Y., Zhang, W., Lu, Q., Qiu, L., ... & Li, L. (2025). Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 7920-7939).

[5] Zheng, X., An, K., Wang, Z., Wang, Y., & Wu, Y. (2025). StepSearch: Igniting LLMs search ability via step-wise proximal policy optimization. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 21816-21841).

[6] Sun, H., Qiao, Z., Wang, B., Chen, G., Hou, Y., Jiang, Y., ... & Zhang, Y. (2025). DecoupleSearch: Decouple planning and search via hierarchical reward modeling. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 4465-4478).

[7] Dai, Y., Yang, S., Wang, G., Deng, Y., Zhang, Z., Yin, J., ... & Lu, S. (2026). Careful queries, credible results: Teaching rag models advanced web search tools with reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 40, No. 36, pp. 30458-30466).

[8] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2022). React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*.

[9] Tang, X., Gao, Q., Li, J., Du, N., Li, Q., & Xie, S. (2025). Mba-rag: A bandit approach for adaptive retrieval-augmented generation through question complexity. In *Proceedings of the 31st International Conference on Computational Linguistics* (pp. 3248-3254).

[10] Li, X., Dong, G., Jin, J., Zhang, Y., Zhou, Y., Zhu, Y., ... & Dou, Z. (2025). Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 5420-5438).

[11] Li, Y., Luo, Q., Li, X., Li, B., Cheng, Q., Wang, B., ... & Qiu, X. (2025). R3-RAG: Learning step-by-step reasoning and retrieval for LLMs via reinforcement learning. In *EMNLP (Findings)* (pp. 10491-10507).

[12] Jiang, P., Xu, X., Lin, J., Xiao, J., Wang, Z., Sun, J., & Han, J. (2025). S3: You don't need that much data to train a search agent via RL. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 21610-21628).

DOI: 10.54254/2755-2721/2026.36624

[13] Zhang, L., Wang, B., Qiu, X., Reddy, S., & Agrawal, A. (2025). Rearank: Reasoning re-ranking agent via reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 2458-2471).

[14] Sun, J., Zhong, X., Zhou, S., & Han, J. (2026). DynamicRAG: Leveraging outputs of large language model as feedback for dynamic reranking in retrieval-augmented generation. *Advances in Neural Information Processing Systems, 38*, 168661-168692.

[15] Li, K., Li, Y., Zhang, T., Luo, H., Wu, X., Glass, J., & Meng, H. (2025). RAG-Zeval: Enhancing RAG responses evaluator through end-to-end reasoning and ranking-based reinforcement learning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing* (pp. 24936-24954).