Improving Retrieval-Augmented Generation through Multi-Agent Reinforcement Learning

Yiqun Chen[1] **Lingyong Yan**[2] **Weiwei Sun**[3] **Xinyu Ma**[2] **Yi Zhang**[2]
Shuaiqiang Wang[2] **Dawei Yin**[2] **Yiming Yang**[3] **Jiaxin Mao**[1*]
[1]Renmin University of China [2]Baidu Inc.
[3]Carnegie Mellon University
chenyiqun990321@ruc.edu.cn, maojiaxin@gmail.com

Abstract

Retrieval-augmented generation (RAG) is widely utilized to incorporate external knowledge into large language models, thereby enhancing factuality and reducing hallucinations in question-answering (QA) tasks. A standard RAG pipeline consists of several components, such as query rewriting, document retrieval, document filtering, and answer generation. However, these components are typically optimized separately through supervised fine-tuning, which can lead to misalignments between the objectives of individual components and the overarching aim of generating accurate answers. Although recent efforts have explored using reinforcement learning (RL) to optimize specific RAG components, these approaches often focus on simple pipelines with only two components or do not adequately address the complex interdependencies and collaborative interactions among the modules. To overcome these limitations, we propose treating the complex RAG pipeline with multiple components as a multi-agent cooperative task, in which each component can be regarded as an RL agent. Specifically, we present MMOA-RAG[2], **M**ulti-**M**odule joint **O**ptimization **A**lgorithm for **RAG**, which employs multi-agent reinforcement learning to harmonize all agents' goals toward a unified reward, such as the F1 score of the final answer. Experiments conducted on various QA benchmarks demonstrate that MMOA-RAG effectively boost the overall performance of the pipeline and outperforms existing baselines. Furthermore, comprehensive ablation studies validate the contributions of individual components and demonstrate MMOA-RAG can be adapted to different RAG pipelines and benchmarks.

1 Introduction

Large Language Models (LLMs) have been widely applied to tasks such as question answering [1, 23], information retrieval [45, 2], various forms of reasoning [15, 13], and evaluation [11, 8]. However, since LLMs cannot promptly update their internal knowledge after pre-training, they are still prone to generating outdated or fabricated responses [58]. To address these challenges, Retrieval-Augmented Generation (RAG) enhances the generative capabilities of LLMs by retrieving relevant information from external knowledge sources. Recent RAG systems are often built as complex pipelines comprising multiple interconnected modules [10], including query rewriting [30, 19], first-stage retrieval [25, 40], re-ranking [36, 37], document preprocessing [22, 27], and answer generation [40, 44].

The complexity of RAG systems makes their optimization particularly challenging. Standard supervised fine-tuning (SFT) optimizes each module independently using human-annotated data. However,

*Corresponding author.

[2]The code of MMOA-RAG is on https://github.com/chenyiqun/MMOA-RAG.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

 https://doi.org/10.52202/085713-4047

this often results in misalignment between the objectives of individual components and the overarching goal of the system of generating high-quality results. For example, retrieval modules are frequently trained on human-labeled relevance data to optimize metrics such as nDCG[18]. However, this process does not address the disconnect between document relevance and response quality - documents with high relevance scores do not always contribute to generating accurate answers [6].

To address this issue, existing work on end-to-end optimization for RAG, such as [24, 12, 25, 37] aims to propagate rewards from the final output to intermediate modules using techniques such as attention distributions [17], generation probability [25, 56], and expectation maximization (EM) iterations [42, 36]. However, earlier approaches primarily focus on simplified pipelines with only two components-a retriever and a generator-and fail to provide a generalizable framework for jointly optimizing complex systems with multiple components and richer interdependencies. More recent methods attempt to eliminate the need for module-specific rewards by leveraging algorithms like Direct Preference Optimization (DPO) [33] and Proximal Policy Optimization (PPO)[39]. Nonetheless, these methods still concentrate on optimizing individual RAG modules in isolation, without adequately modeling the collaborative dynamics between interacting components [27, 30, 22]. Effectively capturing interdependencies among multiple modules and jointly optimizing complex RAG architectures remains an open research challenge.

In this paper, we propose a novel approach called the **M**ulti-**M**odule joint **O**ptimization **A**lgorithm (MMOA-RAG) to enable joint optimization across multiple modules in a RAG system. Our framework treats each intermediate component in the RAG pipeline as an agent and formulates the optimization process as a **Co**operative **M**ulti-**A**gent **R**einforcement **L**earning (Co-MARL) problem, where the agents (i.e., modules) work together to maximize a shared reward for the final outcome.

Specifically, we focus on applying MMOA-RAG to a RAG pipeline that includes four key modules: a query rewriter, a fixed document retriever, a document selector, and an answer generator. Our primary objective is to optimize these modules by defining the final reward as the correctness of the generated response, measured by the F1 score against the ground-truth answer. To achieve this, we leverage the Multi-Agent PPO (MAPPO) algorithm [55], which facilitates collaborative optimization in a fully cooperative setting. This means that all modules work in a cooperative way, with their optimization goals aligned toward producing high-quality answers. Unlike previous methods that rely on DPO [62, 61] or PPO [30, 22], MMOA-RAG offers greater flexibility for different pipeline designs and excels at promoting collaboration among multiple modules. This end-to-end optimization ensures that each module's objectives are consistently aligned with the overarching goal of generating accurate responses.

To demonstrate the effectiveness of the MMOA-RAG modeling and optimization approach, we conducted experiments on three publicly available QA datasets, HotpotQA [53], 2WikiMultihopQA [14] and AmbigQA [31], based on Llama-3-8B-Instruct [7]. The experimental results indicate that MMOA-RAG achieves better performance than a series of existing optimization methods for RAG. Additionally, we performed extensive ablation studies to investigate the effectiveness and advantages of jointly optimizing mulitple modules in the RAG system and the generalizability of MMOA-RAG across different RAG pipelines.

Our main contributions are as follows:

- We innovatively model RAG as a multi-agent collaborative task, treating multiple modules within the RAG pipeline as individual agents.
- We employ a multi-agent reinforcement learning algorithm to jointly optimize a sophisticated RAG system with four key modules: a query writer, a fixed document retriever, a document selector, and an answer generator.
- We conduct extensive experiments to verify and demonstrate the effectiveness and generalizability of the proposed framework.

2 Related Works

End-to-end Optimization in OpenQA Lewis et al. [25] introduces Retrieval-Augmented Generation as RAG, which combines pre-trained language models with non-parametric memory for improved performance on knowledge-intensive NLP tasks. And some other methods [24, 12, 17, 56] proposed end-to-end optimizing framework for OpenQA system.

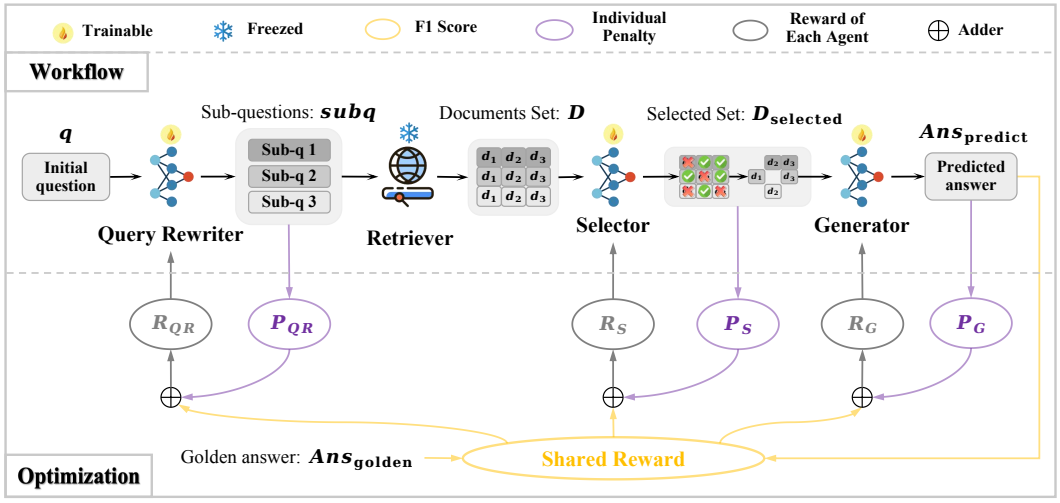


Figure 1: The overall framework of MMOA-RAG.

RAG without Parameters Updating Some works [20, 50, 44, 46] design a novel RAG mechanism without parameters updating, enhancing the performance of LLMs on question answering tasks.

RAG with Parameters Updating Some works [51, 57, 48] optimize RAG with supervised fine-tuning. PPO [39] is used by some works [39, 30, 22, 9, 19] to fine-tune LLMs. Specifically, Search-r1 [21] and R1-Searcher [43] both use answer-based reward to improve the reasoning in RAG. Additionally, DPO [33] or similar alignment algorithms are used to optimize LLMs in RAG task [33, 62, 61, 59, 27]. More detailed related works can be seen in Appendix A.

3 Method

3.1 Modeling RAG as Co-MARL

In this work, we conceptualize the RAG procedure within a cooperative multi-agent reinforcement learning (Co-MARL) framework. Within this framework, each module of the RAG pipeline functions as an individual RL agent. The overarching objective of this multi-agent system is to produce high-quality answers, which aligns with the individual goals of each module.

We define the tuple $\langle \mathcal{G}, \mathcal{O}, \mathcal{A}, \mathcal{R} \rangle$, where $\mathcal{G}$ denotes the set of agents in the Co-MARL system, $\mathcal{O}$ represents the observation information available to each agent, $\mathcal{A}$ constitutes the action space accessible to each agent, and $\mathcal{R}$ is the reward shared among all agents. The ultimate aim is to maximize this shared reward, thereby achieving higher evaluation metrics and enhancing the overall performance of the RAG system.

In this paper we utilizes Multi-Agent PPO (MAPPO) [55], which is an extension of the PPO algorithm [39] for multi-agent environments, to optimize the policy for each agent in the Co-MARL framework. In fully cooperative settings, unlike PPO, which focuses on single-agent scenarios with individual reward, MAPPO employs a shared global reward to promote cooperation among all agents.

3.2 Overall of MMOA-RAG

RAG systems typically follow a modular architecture composed of multiple interconnected components. Figure 1 illustrates the architecture of our MMOA-RAG framework, which consists of four primary modules: the Query Rewriter, the Retriever, the Selector, and the Generator:

- **Query Rewriter** reformulates the initial question q, which may be too complex or ambiguous to resolve with a single retrieval, into a set of sub-questions denoted as $subq$.
- **Retriever** retrieves relevant documents from the corpus for each sub-questions, respectively, and outputs a set of candidate document D.

https://doi.org/10.52202/085713-4047

- **Selector** further filters D to obtain a subset of documents D_{selected} that is useful for generating the final answer to the initial query q.
- **Generator** leverages D_{selected} to generate the predicted answer Ans_{predict} to the initial question.

Since the Query Rewriter, Selector, and Generator modules can all be implemented using LLMs, they can be treated as RL agents [32], enabling parameter updates through reward signals. To optimize computational efficiency, these three modules can share the same LLM. Additionally, given the difficulty of modeling the Retriever module as an RL agent, we use a fixed Retriever and regard it as a part of the environment [3].

The focus of the MMOA-RAG framework is on the collaborative optimization of multiple modules to align their individual objectives with the ultimate goal of generating high-quality answers. We use metrics from the Generator's predicted answer Ans_{predict}, such as F1 score, as a shared reward R_{shared}. Given the fully cooperative nature of the modules in the RAG system, R_{shared} can be used to train all agents, a common approach in existing MARL literature [35, 55, 4]. Additionally, to ensure training stability and accelerate convergence in the multi-agent system, we design penalty terms P_{QR}, P_S, and P_G for each agent. A more detailed explanation will be provided in Section 3.3.

3.3 Detailed Configuration for Each Agent

In this section, we will provide a detailed explanation of each element in the tuple $\langle \mathcal{G}, \mathcal{O}, \mathcal{A}, \mathcal{R} \rangle$ mentioned in Section 3.1. Here, $\mathcal{G} = \{\text{Query Rewriter (QR), Selector (S), Generator (G)}\}$ represents all agents. In the following, we introduce the essential elements for each agent $i \in \mathcal{G}$: the observation information $O_i \in \mathcal{O}$, the action space $A_i \in \mathcal{A}$, and the reward function R_i.

3.3.1 Elements of Query Rewriter

Observation of Query Rewriter is defined as Equation (1), which contains prompt of Query Rewriter $Prompt_{QR}$ (as shown in Table 3) and the initial question q.

$$O_{QR} = \{Prompt_{QR}, q\} \tag{1}$$

Action Space of Query Rewriter corresponds to the vocabulary of LLMs, denoted as $\mathcal{V}$, as we prompt the LLM to generate one or more sub-questions based on q.

$$A_{QR} = \mathcal{V} \tag{2}$$

Reward Function of the Query Rewriter is defined as shown in Equation (3). Here, R_{shared} can be the metric for the final answer, depicted as the yellow section in Figure 1. In this paper, we utilize the F1 score of the predicted answer, Ans_{predict}, as the shared reward. The term P_{QR} serves as a penalty to discourage the Query Rewriter from generating an excessive number of sub-questions during training. Specifically, P_{QR} is assigned a value of -0.5 if the number of sub-questions exceeds four, and it is set to 0 if the number of sub-questions is four or fewer.

$$R_{QR} = R_{\text{shared}} + P_{QR} \tag{3}$$

3.3.2 Elements of Selector

Observation of Selector is defined as Equation (4), which contains prompt of Selector $Prompt_S$ (as shown in Table 4), the initial question q and the candidate documents set D with K documents.

$$O_S = \{Prompt_S, q, D\} \tag{4}$$

Action Space of Selector only comprises of several words as Equation (5). Since the function of the Selector is to output the IDs of candidate documents helpful to answering the initial question q, the action space is constrained to this limited set of words. This constraint can significantly reduce the exploration space of the Selector and provide a more stable training process.

$$A_S = \{\text{"}\mathbf{0}\text{"}, \text{"}\mathbf{1}\text{"}, \ldots, \text{"}\mathbf{K\text{-}1}\text{"}, \text{"}\mathbf{Document}\text{"}, \text{","}\} \tag{5}$$

[3] Recent studies in generative IR (see [26] for a survey) have explored using generative models for retrieval. But we choose a more traditional dense retrieval model [16] as the first-stage retriever and leave the optimization of the first-stage retriever for future work.

Reward Function of Selector also contains two terms, which are R_{shared} and P_S. And P_S is a penalty term designed to prevent the Selector from generating duplicate document IDs and from outputting IDs that do not conform to the specified format (e.g., Document0,Document3,Document9). When the Selector outputs duplicate document IDs or fails to adhere to the specified format, P_S is set to -1; otherwise, P_S is set to 0.

$$R_S = R_{\text{shared}} + P_S \tag{6}$$

3.3.3 Elements of Generator

Observation of Generator is in Equation (7), containing prompt of Generator $Prompt_G$ (as shown in Table 5), the initial question q and the selected candidate documents set D_{selected} given by Selector.

$$O_G = \{Prompt_G, q, D_{\text{selected}}\} \tag{7}$$

Action Space of Generator A_G is the same as Query Rewriter.

$$A_G = A_{QR} = \mathcal{V} \tag{8}$$

Reward Function of Generator contains R_{shared} and penalty term P_G, which is used to constrain the model from generating excessively long content. When the generated answer exceeds a certain length, P_G is set to -0.5; otherwise, it is set to 0. In fact, the values of each penalty P_i $(i \in \mathcal{G})$ are mostly 0, and they only become negative when the output does not meet the requirements.

$$R_G = R_{\text{shared}} + P_G \tag{9}$$

3.4 Training Process of MMOA-RAG

3.4.1 Warm Start with SFT

In preparation for joint optimization of multiple modules using Multi-Agent PPO, it is essential to perform a warm start for each trainable module. The warm start enables the model to better adhere to instructions across diverse tasks and reduces the exploration space during MARL joint training, thereby enhancing the efficiency of exploration and exploitation.

Within the MMOA-RAG framework, there are three trainable modules: the Query Rewriter, the Selector, and the Generator. Consequently, we construct the training data for the SFT of each corresponding task and perform the SFT to get the warm-up checkpoints for each trainable modules. The details of constructing training data can be seen in Appendix B.

3.4.2 Multi-Agent Optimization

After undergoing SFT, the LLM demonstrates an improved ability to follow instructions while executing the functions of Query Rewriter, Selector, and Generator. The RAG system also achieves relatively satisfactory warm-start performance. To further enhance the performance of the RAG system, which is modeled as a fully cooperative multi-agent system, it is crucial to conduct joint training of multiple agents to strengthen collaboration among them.

We adopt a setup similar to Multi-Agent PPO [55] in Starcraft II, where multiple agents share a global reward, optimizing $\mathcal{G} = \{\text{QR, S, G}\}$ with R_{shared}. To reduce computational overhead, we apply the parameter-sharing mechanism among agents, allowing QR, S, and G to utilize the same LLM.

In the multi-agent optimization process, there are three models to consider: the Actor model, the Critic model, and the SFT model. The parameters for these models are denoted as θ, ϕ, and θ_{SFT}, respectively. The role of the Actor model is to provide the response $Answer_i$ based on the observation O_i for each agent i. The Critic model is responsible for estimating the state-value function $V_\phi^{i,t}$, which is a classic setup in Actor-Critic architecture within RL algorithms. The SFT model serves as a baseline for the Actor model, similar to InstructGPT [32]. The objective is to update the parameters of both the Actor and Critic models. The overall loss function, $\mathcal{L}(\theta, \phi)$, consists of two terms: $\mathcal{L}_{\text{Actor}}(\theta)$ and $\mathcal{L}_{\text{Critic}}(\phi)$:

$$\mathcal{L}(\theta, \phi) = \mathcal{L}_{\text{Actor}}(\theta) + \alpha * \mathcal{L}_{\text{Critic}}(\phi) \tag{10}$$

The Actor loss function presented in Equation (11) is similar to that used in the typical single-agent PPO [39] algorithm. The primary difference is that multiple agents are being optimized. In Equation

 https://doi.org/10.52202/085713-4047

(11), $i \in \mathcal{G}$ denotes the three agents: Query Rewriter, Selector, and Generator. The term r_t^i in Equation (12) denotes the importance sampling ratio, which measures the difference between the new and old policies. The expression $\hat{A}_{\pi_\theta}^{i,t}$ in Equation (13) is the advantage function, estimated using Generalized Advantage Estimation (GAE) [38]. The variable δ_t^i in Equation (14) is known as the temporal difference (TD) error at time step t.

$$\mathcal{L}_{\text{Actor}}(\theta) = \sum_i \sum_t \min\left(r_t^i \hat{A}_{\pi_\theta}^{i,t},\ \text{clip}\left(r_t^i, 1-\epsilon, 1+\epsilon\right) \hat{A}_{\pi_\theta}^{i,t}\right) \tag{11}$$

$$r_t^i = \frac{\pi_\theta(a_t^i \mid s_t^i)}{\pi_{\theta_{\text{old}}}(a_t^i \mid s_t^i)} \tag{12}$$

$$\hat{A}_{\pi_\theta}^{i,t} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^i \tag{13}$$

$$\delta_t^i = R(s_t^i, a_t^i) + \gamma V_\phi(s_{t+1}^i) - V_\phi(s_t^i) \tag{14}$$

Similar to InstructGPT [32], the final reward function $R(s_t^i, a_t^i)$ is defined in Equation (15). The distinction is that our approach does not require a trained reward model, as we use the evaluation metric (F1 score) of the predicted answers Ans_{predict} of Generator as the shared reward R_{shared} for all agents. The penalty term P_i can also be easily obtained from the output of each agent, as introduced in Section 3.3. The components R_i in Equation (15) are defined in Equations (3), (6), or (9). And $Answer_i$ represents the output generated by each agent i based on its individual observation O_i.

$$R(s_t^i, a_t^i) = \begin{cases} 0, & \text{if } t < T \\ \underbrace{R_{\text{shared}} + P_i}_{R_i,\ \text{and}\ i \in \mathcal{G}} - \beta \log\left(\frac{\pi_\theta(Answer_i|O_i)}{\pi_{\theta_{\text{SFT}}}(Answer_i|O_i)}\right), & \text{if } t = T \end{cases} \tag{15}$$

The loss function of the Critic model, as shown in Equation (16), employs a clipping operation similar to the Actor model. Here, $\Delta V_{i,t} = V_\phi^{i,t} - V_{\text{target}}^{i,t}$, where $V_\phi^{i,t} = V_\phi(s_t^i)$. The term $V_{\text{target}}^{i,t}$ represents the cumulative return and s_t^i is the state-value function.

$$\mathcal{L}_{\text{Critic}}(\phi) = \sum_i \sum_t \max\left[(\Delta V_{i,t})^2, \left(\text{clip}\left(V_\phi^{i,t}, V_{\phi_{\text{old}}}^{i,t} \pm \epsilon\right) - V_{\text{target}}^{i,t}\right)^2\right] \tag{16}$$

The pseudocode for multi-agent optimization based on MAPPO is shown in **Algorithm 1** in Appendix C, which corresponds to the overall framework of MMOA-RAG depicted in Figure 1. For a specific question, the first step is to execute the `Collect Rollout` process. This process involves passing through the Query Rewriter, Retriever, Selector, and Generator, and the computed tuple $\mathcal{T} = ((O_{QR}, subq, R_{QR}), (O_S, IDs, R_S), (O_G, Ans_{\text{predict}}, R_G))$ is stored in the replay buffer $\mathcal{M}$. Next, the `Policy and Value Optimization` process is executed where the GAE is used to estimate the advantage function $\hat{A}_{\pi_\theta}^{i,t}$. Subsequently, the overall loss function $\mathcal{L}(\theta, \phi)$ is calculated, and the parameters of both the Actor and Critic models are updated. Additionally, to accelerate the entire training process, we can run a minibatch in parallel. Ultimately, we obtain a well-trained Actor model used for subsequent inference and evaluation.

4 Experiments

Our experiments mainly aim to explore the following research questions:

RQ1: How does our MMOA-RAG perform compared to existing RAG optimization methods?

RQ2: How does the joint optimization of individual modules in the RAG pipeline contribute to the effectiveness of MMOA-RAG framework?

RQ3: Can MMOA-RAG exhibit generalizability across different RAG systems?

4.1 Experimental Settings

Datasets and Evaluation We conducted experiments using MMOA-RAG alongside various baseline models across three open-domain QA datasets: HotpotQA [53], 2WikiMultihopQA [14], and AmbigQA [31]. The candidate documents are all retrieved from Wikipedia passages for three datasets.

Table 1: Performance for different methods across datasets. All the results in this table are obtained using Contriever [16] as the retrieval model. In each dataset, the highest baseline value is underscored. The symbol Δ displays the improvement of MMOA-RAG over the best baseline.

Methods	HotpotQA			2WikiMultihopQA			AmbigQA		
	Acc	EM	F1	Acc	EM	F1	Acc	EM	F1
LLM w/o RAG	25.08	21.31	31.18	27.78	23.68	29.47	27.21	20.96	33.42
Vanilla RAG w/o train	27.99	20.62	30.67	31.94	13.91	22.84	31.09	22.42	33.56
Vanilla RAG w SFT	36.18	32.30	44.49	39.47	38.28	43.36	34.41	30.74	44.36
SELF-RAG [1]	30.42	27.77	38.93	36.32	35.39	38.86	28.35	25.70	39.04
RetRobust [54]	37.69	34.60	46.49	41.02	39.73	44.51	35.13	32.37	44.78
Rewrite-Retrieve-Read [30]	38.03	33.93	46.32	40.40	39.17	44.17	35.94	31.90	45.92
BGM [22]	36.05	32.76	44.54	39.61	38.61	43.29	36.01	32.53	45.76
RAG-DDR [27]	35.20	32.65	44.26	40.49	39.45	44.18	36.25	32.55	45.83
MMOA-RAG (ours)	**39.15**	**36.15**	**48.29**	**42.73**	**41.52**	**46.40**	**38.85**	**34.75**	**48.59**
Δ	**+1.12**	**+1.55**	**+1.80**	**+1.71**	**+1.79**	**+1.89**	**+2.60**	**+2.20**	**+2.67**

We employ three key evaluation metrics—Accuracy, Exact Match (EM), and F1 score—to assess the performance of the RAG methods.

Implementation Details We utilize Contriever [16] as the Retriever for most experiments. And the Selector consistently receives a fixed set of $K = 10$ documents as input. Besides, we employ Llama-3-8B-Instruct [7] as the foundational LLM for all the baselines and MMOA-RAG.

We compare MMOA-RAG with different baseline methods: **LLM w/o RAG**, **Vanilla RAG w/o train**, **Vanilla RAG w SFT**, **SELF-RAG** [1], **RetRobust** [54], **Rewrite-Retrieve-Read** [30], **BGM** [22], **RAG-DDR** [27].

The detailed introduction of experimental settings and baselines can be seen in Appendix D.

4.2 Comparisons with Other Methods

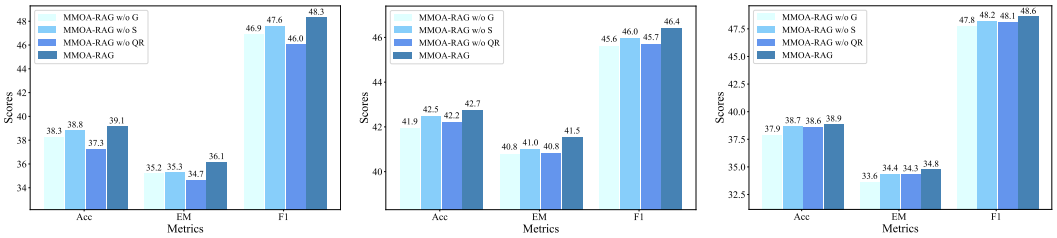
We conducted a comparative analysis of MMOA-RAG against multiple baselines, with the results presented in Table 1. To ensure fairness in comparison, all methods utilized Llama-3-8B-Instruct as the backbone LLM, and all baselines were re-implemented according to the settings delineated in Appendix D.2.

Firstly, as shown in Table 1, MMOA-RAG demonstrates superior performance across all metrics and datasets, highlighting its effectiveness. Additionally, it is noteworthy that Vanilla RAG w/o train achieves comparable results to LLM w/o RAG across various metrics. This observation suggests that the pre-trained Llama-3-8B-Instruct struggles to effectively leverage external knowledge for answer generation, likely due to the absence of RAG-related tasks in its pre-training process, which limits its external knowledge utilization. In contrast, Vanilla RAG w SFT exhibits substantial improvements over Vanilla RAG w/o train across all evaluation metrics. This indicates that the SFT-enhanced Llama-3-8B-Instruct is adept at utilizing external knowledge, successfully extracting valuable information from noisy candidate documents to enhance the quality of generated answers.

The Rewrite-Retrieve-Read and BGM approaches enhance Vanilla RAG by respectively integrating a query rewrite module and a bridge module, each of which is trained using the PPO algorithm. As indicated in Table 1, on the multi-hop datasets HotpotQA and 2WikiMultihopQA, Rewrite-Retrieve-Read surpasses BGM, suggesting that the inclusion of a query rewrite module is more effective than adding a bridge module for these multi-hop datasets. Conversely, on the single-hop dataset AmbigQA, the performance of Rewrite-Retrieve-Read and BGM is relatively similar. Our MMOA-RAG can be conceptualized as augmenting Vanilla RAG by integrating both a Query Rewriter and a Selector, whose roles are akin to the query rewrite module in Rewrite-Retrieve-Read and the bridge module in BGM. The primary advantage of MMOA-RAG lies in its simultaneous optimization of the Query Rewriter, Selector, and Generator modules. This is achieved by aligning the objectives of these modules with the goal of generating higher-quality answers via MAPPO. The experimental results presented in Table 1 further illustrate that MMOA-RAG significantly outperforms Rewrite-Retrieve-Read, BGM, and other baselines.

In addition, we tested various methods based on other retriever, BGE [49] and E5 [47], and the results can be seen in Table 7 in Appendix E. And we also perform out-of-domain experiments, the results can be seen in Table 8 in Appendix F. In summary, The results in Table 1, Table 7, Table 8, and the

 https://doi.org/10.52202/085713-4047



(a) HotpotQA (b) 2WikiMultihopQA (c) AmbigQA

Figure 2: Ablation about Optimizing Different Agents. In this figure, MMOA-RAG w/o i ($i \in \{\text{QR, S, G}\}$) denote the variant where agent i is excluded from the complete optimization process of multi-agent joint optimization.

analysis in this Section 4.2 jointly answer the **RQ1**. And the case study can be found in Appendix I, from which we can intuitively understand the advantages of multi-module joint optimization.

4.3 Ablation Experiments on the Optimization of Different Agents

To demonstrate the necessity of multi-agent joint optimization in RAG systems, we present ablation experiments in this section. The MMOA-RAG framework, depicted in Figure 1, consists of three agents: $i \in \{\text{Query Rewriter (QR), Selector (S), Generator (G)}\}$. In Figure 2, MMOA-RAG w/o i denotes the variant where agent i is excluded from the complete optimization process of multi-agent joint optimization.

As illustrated in Figure 2, the complete version of MMOA-RAG, where all three modules are jointly optimized, delivers the highest performance. This underscores the effectiveness of multi-agent joint optimization within the RAG system and validates the importance of optimizing multiple modules concurrently. Additionally, the MMOA-RAG w/o S variant achieves the best performance among the three ablation configurations. The Selector's primary function is to refine the candidate document set D, yielding a higher-quality subset D_{selected}, which enhances the Generator's ability to produce a superior answer Ans_{predict}. However, through the joint optimization by MAPPO, the Generator acquires some denoising capabilities. Consequently, satisfactory results can be achieved even when the Selector is not optimized during joint optimization.

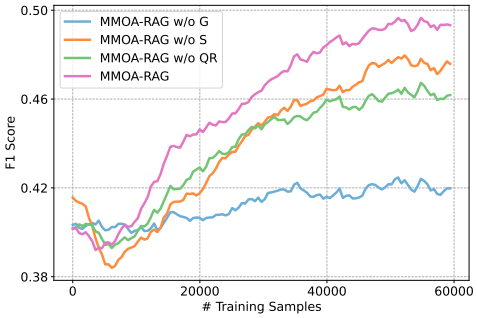


Figure 3: Ablation experiments on AmbigQA dataset. The horizontal axis represents the number of training samples, while the vertical axis denotes the shared reward R_{shared} (F1 score) during the training process.

We also present the trajectory of the shared reward R_{shared} during the training process based on ablation experiments conducted on the AmbigQA dataset, as shown in Figure 3. From Figure 3, it is evident that the reward curve for MMOA-RAG demonstrates the fastest convergence rate and achieves the highest final convergence value. This underscores the effectiveness of joint optimization across multiple modules in significantly and efficiently enhancing the performance of the RAG system. Furthermore, the training curve for MMOA-RAG w/o G in Figure 3 is noticeably slower compared to other algorithms, and the test results for MMOA-RAG w/o G on the AmbigQA dataset, as shown in Figure 2, are the poorest. These findings suggest that the Generator module is the most critical component for the single-hop AmbigQA dataset.

The results in Figure 2 and Figure 3 answer the **RQ.2** that it is more effective to optimize multiple modules in a RAG system simultaneously.

4.4 Generality Experiments on RAG Systems with Varying Module Configurations

In this section, we evaluate the performance of MMOA-RAG in optimizing RAG systems with different numbers of agents, as detailed in Table 2. In Table 2, QR+S+G represents the RAG

Table 2: Generality Experiments on RAG Systems with Varying Module Configurations. In the second column, SFT and MAPPO refer to the current module configuration following the warm start training stage (Section 3.4.1) and the MAPPO joint training stage (Section 3.4.2), respectively. The symbol Δ signifies the enhancement achieved in the MAPPO stage relative to the SFT stage.

Modules	Training Stage & Delta	HotpotQA			2WikiMultihopQA			AmbigQA		
		Acc	EM	F1	Acc	EM	F1	Acc	EM	F1
QR+S+G	SFT	36.00	33.04	44.69	39.54	38.50	42.97	36.55	32.60	46.71
	MAPPO	39.15	36.15	48.29	42.73	41.52	46.40	38.85	34.75	48.59
	Δ	**+3.15**	**+3.11**	**+3.60**	**+3.19**	**+3.02**	**+3.43**	**+2.30**	**+2.15**	**+1.88**
S+G	SFT	34.25	32.18	43.14	38.93	37.97	42.40	35.85	32.35	45.82
	MAPPO	38.23	34.85	47.07	41.79	40.57	45.25	37.60	33.90	47.19
	Δ	**+3.98**	**+2.67**	**+3.93**	**+2.86**	**+2.60**	**+2.85**	**+1.75**	**+1.55**	**+1.37**
QR+G	SFT	36.76	32.78	45.00	39.15	37.89	42.91	35.50	31.50	45.31
	MAPPO	38.90	35.89	47.94	42.43	41.01	46.19	37.65	33.50	47.53
	Δ	**+2.14**	**+3.11**	**+2.94**	**+3.28**	**+3.12**	**+3.28**	**+2.15**	**+2.00**	**+2.22**

framework depicted in Figure 1, illustrating a multi-agent system composed of three agent, Query Rewriter, Selector and Generator. The configuration S+G results from omitting the Query Rewriter agent, relying solely on the initial question q for retrieval, thereby configuring the RAG system as a two-agent (Selector and Generator) system. Conversely, QR+G denotes the exclusion of the Selector agent, forming a RAG pipeline consisting of two agents, Query Rewriter and Generator. The second column of Table 2 specifies that SFT refers to the warm start of all agents in the corresponding RAG system through supervised fine-tuning, while MAPPO refers to the joint optimization of all agents built upon SFT utilizing the MAPPO framework. The notation Δ is used to denote the performance enhancement achieved by MAPPO compared to SFT.

The experimental results in Table 2 reveal that RAG systems optimized using joint MAPPO consistently outperform those using only SFT across all datasets. This finding underscores the robust generalizability of the MMOA-RAG joint optimization approach, yielding significant performance improvements across diverse RAG configurations. Notably, the performance gains from MAPPO over SFT are approximately three percentage points on multi-hop datasets such as HotpotQA and 2WikiMultihopQA, while improvements on the single-hop dataset AmbigQA are around two percentage points. This difference may stem from the greater complexity inherent to multi-hop datasets, which potentially exacerbates misalignment among different modules during the SFT stage. These results further highlight the necessity of multi-module joint optimization, especially in the context of more challenging multi-hop datasets.

The results presented in Table 2 demonstrate the effectiveness of MMOA-RAG in optimizing various RAG systems across different configurations, thereby answering **RQ.3**.

5 Conclusions and Future Works

In this paper, we model the RAG system as a multi-agent collaborative task, wherein we consider the Query Rewriter, Selector, and Generator modules as learnable RL agents. We employ a multi-agent reinforcement learning algorithm to jointly optimize these agents, aligning the optimization goals of multiple modules with the ultimate objective of generating high-quality answers.

Our experiments demonstrate the effectiveness of our modeling approach and joint optimization method. Comprehensive ablation studies confirm the necessity and generality of multi-module joint optimization, establishing MMOA-RAG as an effective approach for optimizing RAG systems.

In future works, we intend to explore the application of MMOA-RAG in more complex workflows. This exploration will include scenarios where the RAG workflows are organized as a directed acyclic graph, as well as situations involving dynamic workflows within agentic RAG. Additionally, it is important to assess the cost and latency associated with specific modules within the RAG system. In this regard, the design of the reward function should not be exclusively based on evaluation metrics, such as the F1 score highlighted in this paper. Instead, it should aim to strike a balance between effectiveness and cost in those partially cooperative RAG scenarios.

 https://doi.org/10.52202/085713-4047

Acknowledgements

This research was supported by the Natural Science Foundation of China (61902209, 62377044), Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China, the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China (22XNKJ15), and Beijing Nova Program.

References

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

[2] Yiqun Chen, Qi Liu, Yi Zhang, Weiwei Sun, Daiting Shi, Jiaxin Mao, and Dawei Yin. Tourrank: Utilizing large language models for documents ranking with a tournament-inspired strategy. *arXiv preprint arXiv:2406.11678*, 2024.

[3] Yiqun Chen, Hangyu Mao, Jiaxin Mao, Shiguang Wu, Tianle Zhang, Bin Zhang, Wei Yang, and Hongxing Chang. Ptde: Personalized training with distilled execution for multi-agent reinforcement learning. *arXiv preprint arXiv:2210.08872*, 2022.

[4] Yiqun Chen, Hangyu Mao, Tianle Zhang, Shiguang Wu, Bin Zhang, Jianye Hao, Dong Li, Bin Wang, and Hongxing Chang. Ptde: Personalized training with distillated execution for multi-agent reinforcement learning. *arXiv preprint arXiv:2210.08872*, 2022.

[5] Yiqun Chen, Erhan Zhang, Lingyong Yan, Shuaiqiang Wang, Jizhou Huang, Dawei Yin, and Jiaxin Mao. Mao-arag: Multi-agent orchestration for adaptive retrieval-augmented generation. *arXiv preprint arXiv:2508.01005*, 2025.

[6] Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. The power of noise: Redefining retrieval for rag systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 719–729, 2024.

[7] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[8] Jinlan Fu, See-Kiong Ng, Zhengbao Jiang, and Pengfei Liu. Gptscore: Evaluate as you desire. *arXiv preprint arXiv:2302.04166*, 2023.

[9] Jingsheng Gao, Linxu Li, Weiyuan Li, Yuzhuo Fu, and Bin Dai. Smartrag: Jointly learn rag-related tasks from the environment feedback. *arXiv preprint arXiv:2410.18141*, 2024.

[10] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023.

[11] Peiyuan Gong and Jiaxin Mao. Coascore: Chain-of-aspects prompting for nlg evaluation. *arXiv preprint arXiv:2312.10355*, 2023.

[12] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020.

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

[14] Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*, 2020.

[15] Jie Huang and Kevin Chen-Chuan Chang. Towards reasoning in large language models: A survey. *arXiv preprint arXiv:2212.10403*, 2022.

[16] Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*, 2021.

[17] Gautier Izacard and Edouard Grave. Distilling knowledge from reader to retriever for question answering. *arXiv preprint arXiv:2012.04584*, 2020.

[18] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.

[19] Jinhao Jiang, Jiayi Chen, Junyi Li, Ruiyang Ren, Shijie Wang, Wayne Xin Zhao, Yang Song, and Tao Zhang. Rag-star: Enhancing deliberative reasoning with retrieval augmented verification and refinement. *arXiv preprint arXiv:2412.12881*, 2024.

[20] Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*, 2023.

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

[22] Zixuan Ke, Weize Kong, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. Bridging the preference gap between retrievers and llms. *arXiv preprint arXiv:2401.06954*, 2024.

[23] Omar Khattab, Keshav Santhanam, Xiang Lisa Li, David Hall, Percy Liang, Christopher Potts, and Matei Zaharia. Demonstrate-search-predict: Composing retrieval and language models for knowledge-intensive nlp. *arXiv preprint arXiv:2212.14024*, 2022.

[24] Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*, 2019.

[25] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[26] Xiaoxi Li, Jiajie Jin, Yujia Zhou, Yuyao Zhang, Peitian Zhang, Yutao Zhu, and Zhicheng Dou. From matching to generation: A survey on generative information retrieval. *arXiv preprint arXiv:2404.14851*, 2024.

[27] Xinze Li, Sen Mei, Zhenghao Liu, Yukun Yan, Shuo Wang, Shi Yu, Zheni Zeng, Hao Chen, Ge Yu, Zhiyuan Liu, et al. Rag-ddr: Optimizing retrieval-augmented generation using differentiable data rewards. *arXiv preprint arXiv:2410.13509*, 2024.

[28] Yuchen Li, Hengyi Cai, Rui Kong, Xinran Chen, Jiamin Chen, Jun Yang, Haojie Zhang, Jiayi Li, Jiayi Wu, Yiqun Chen, et al. Towards ai search paradigm. *arXiv preprint arXiv:2506.17188*, 2025.

[29] Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

[30] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. Query rewriting for retrieval-augmented large language models. *arXiv preprint arXiv:2305.14283*, 2023.

[31] Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. Ambigqa: Answering ambiguous open-domain questions. *arXiv preprint arXiv:2004.10645*, 2020.

 https://doi.org/10.52202/085713-4047

[32] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[33] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[34] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

[35] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *International conference on machine learning*, pages 4295–4304. PMLR, 2018.

[36] Alireza Salemi and Hamed Zamani. Learning to rank for multiple retrieval-augmented models through iterative utility maximization. *arXiv preprint arXiv:2410.09942*, 2024.

[37] Alireza Salemi and Hamed Zamani. Towards a search engine for machines: Unified ranking for multiple retrieval-augmented large language models. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 741–751, 2024.

[38] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

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

[41] Zhengliang Shi, Weiwei Sun, Shen Gao, Pengjie Ren, Zhumin Chen, and Zhaochun Ren. Generate-then-ground in retrieval-augmented generation for multi-hop question answering. *arXiv preprint arXiv:2406.14891*, 2024.

[42] Devendra Singh, Siva Reddy, Will Hamilton, Chris Dyer, and Dani Yogatama. End-to-end training of multi-document reader and retriever for open-domain question answering. *Advances in Neural Information Processing Systems*, 34:25968–25981, 2021.

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

[44] Weihang Su, Yichen Tang, Qingyao Ai, Zhijing Wu, and Yiqun Liu. Dragin: Dynamic retrieval augmented generation based on the real-time information needs of large language models. *arXiv preprint arXiv:2403.10081*, 2024.

[45] Weiwei Sun, Lingyong Yan, Xinyu Ma, Shuaiqiang Wang, Pengjie Ren, Zhumin Chen, Dawei Yin, and Zhaochun Ren. Is chatgpt good at search? investigating large language models as re-ranking agents. *arXiv preprint arXiv:2304.09542*, 2023.

[46] Fei Wang, Xingchen Wan, Ruoxi Sun, Jiefeng Chen, and Sercan Ö Arık. Astute rag: Overcoming imperfect retrieval augmentation and knowledge conflicts for large language models. *arXiv preprint arXiv:2410.07176*, 2024.

[47] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2022.

[48] Zhepei Wei, Wei-Lin Chen, and Yu Meng. Instructrag: Instructing retrieval augmented generation via self-synthesized rationales. In *Adaptive Foundation Models: Evolving AI for Personalized and Efficient Learning*.

[49] Shitao Xiao, Zheng Liu, Peitian Zhang, Niklas Muennighoff, Defu Lian, and Jian-Yun Nie. C-pack: Packed resources for general chinese embeddings. In *Proceedings of the 47th international ACM SIGIR conference on research and development in information retrieval*, pages 641–649, 2024.

[50] Shicheng Xu, Liang Pang, Huawei Shen, Xueqi Cheng, and Tat-Seng Chua. Search-in-the-chain: Interactively enhancing large language models with search for knowledge-intensive tasks. In *Proceedings of the ACM on Web Conference 2024*, pages 1362–1373, 2024.

[51] Shicheng Xu, Liang Pang, Mo Yu, Fandong Meng, Huawei Shen, Xueqi Cheng, and Jie Zhou. Unsupervised information refinement training of large language models for retrieval-augmented generation. *arXiv preprint arXiv:2402.18150*, 2024.

[52] Wei Yang and Jesse Thomason. Learning to deliberate: Meta-policy collaboration for agentic llms with multi-agent reinforcement learning. *arXiv preprint arXiv:2509.03817*, 2025.

[53] Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.

[54] Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. Making retrieval-augmented language models robust to irrelevant context. *arXiv preprint arXiv:2310.01558*, 2023.

[55] Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624, 2022.

[56] Hamed Zamani and Michael Bendersky. Stochastic rag: End-to-end retrieval-augmented generation through expected utility maximization. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2641–2646, 2024.

[57] Qingfei Zhao, Ruobing Wang, Yukuo Cen, Daren Zha, Shicheng Tan, Yuxiao Dong, and Jie Tang. Longrag: A dual-perspective retrieval-augmented generation paradigm for long-context question answering. *arXiv preprint arXiv:2410.18050*, 2024.

[58] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 2023.

[59] Xinping Zhao, Dongfang Li, Yan Zhong, Boren Hu, Yibin Chen, Baotian Hu, and Min Zhang. Seer: Self-aligned evidence extraction for retrieval-augmented generation. *arXiv preprint arXiv:2410.11315*, 2024.

[60] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024.

[61] Junda Zhu, Lingyong Yan, Haibo Shi, Dawei Yin, and Lei Sha. Atm: Adversarial tuning multi-agent system makes a robust retrieval-augmented generator. *arXiv preprint arXiv:2405.18111*, 2024.

[62] Kun Zhu, Xiaocheng Feng, Xiyuan Du, Yuxuan Gu, Weijiang Yu, Haotian Wang, Qianglong Chen, Zheng Chu, Jingchang Chen, and Bing Qin. An information bottleneck perspective for effective noise filtering on retrieval-augmented generation. *arXiv preprint arXiv:2406.01549*, 2024.

 https://doi.org/10.52202/085713-4047

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We claimed the main contribution and the scope in our abstract and introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss some limitations of our proposed method in Appendix G.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all the details and the anonymous code url in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

https://doi.org/10.52202/085713-4047

Answer: [Yes]

Justification: We provide all the details and the anonymous code url in the paper.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide many details of the experiments and the key experimental settings.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We don't do the statistical significance tests.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the computational resource requirements in our anonymous code.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: No ethics problem.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: This discussion is not necessary for our work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate

https://doi.org/10.52202/085713-4047

deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our work poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Our paper meets this requirement.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: We don't release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We only checked for syntax errors using LLM.

Guidelines:

 https://doi.org/10.52202/085713-4047

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

Appendix

A Detailed Related Works

A.1 End-to-end Optimization in OpenQA

ORQA [24] is an open-domain QA system that learns end-to-end evidence retrieval and answer generation using only question-answer pairs, enabled by pretraining with an Inverse Cloze Task. REALM [12] is an end-to-end optimizing framework that enhances language model pre-training with a retrieval-augmented approach. [25] introduces Retrieval-Augmented Generation as RAG, a model that combines pre-trained language models with non-parametric memory for improved performance on knowledge-intensive NLP tasks. [17] propose a knowledge distillation method to train retriever models using synthetic labels derived from reader model attention scores. Stochastic RAG [56] introduces a novel end-to-end optimization framework for RAG through expected utility maximization.

A.2 RAG without Parameters Updating

These methods typically involve designing a novel RAG mechanism to enhance the performance of LLMs on question answering tasks. For example, DSP [23] leverages sophisticated interactions between retrieval and language models to address knowledge-intensive NLP tasks. FLARE [20], an active retrieval augmented generation method, enhances text generation by dynamically retrieving relevant information throughout the process, showing superior performance across various long-form knowledge-intensive tasks. ITER-RETGEN [40] is an iterative retrieval-generation synergy method that enhances retrieval-augmented large language models by synergistically combining retrieval and generation in an iterative manner. Search-in-the-Chain [50] is a framework that interactively enhances Large Language Models with search capabilities to improve performance on complex, knowledge-intensive tasks. SELF-RAG [1] enhances language model quality and factuality through self-reflective retrieval and generation. DRAGIN [44] is a dynamic RAG framework that addresses the real-time information needs of LLMs during text generation, enhancing performance on knowledge-intensive tasks. GenGround [41] synergizes large language model knowledge with external documents to enhance multi-hop question answering through an iterative process of generating answers and grounding them in evidence. Astute RAG [46] is a approach that enhances the robustness of Retrieval-Augmented Generation for Large Language Models by adaptively integrating internal and external knowledge while resolving knowledge conflicts.

A.3 RAG with Parameters Updating

A.3.1 Optimizing RAG with SFT

INFO-RAG [51], an unsupervised training method, enhances the capacity of large language models to integrate and refine information from retrieved texts. LongRAG [57] introduces a dual-perspective retrieval-augmented generation system to enhance understanding of complex long-context knowledge for improved performance in long-context question answering tasks. In INSTRUCTRAG [48], generation accuracy and trustworthiness are enhanced by explicitly denoising retrieved information through self-synthesized rationales, outperforming standard RAG approaches without additional supervision.

A.3.2 Optimizing RAG with RL

Some existing works use PPO [39] algorithm to fine-tune LLMs. In Rewrite-Retrieve-Read framework [30], a small language model is trained with reinforcement learning to rewrite queries for RAG. BGM [22] proposes a novel bridge mechanism between retrieval model and LLMs and uses PPO to optimize the parameters of the bridge to filter for more helpful documents. SMARTRAG [9] optimizes an iterative RAG framework with reward, which includes a decision maker and a policy network. RAG-Star [19] is a reasoning approach that combines Monte Carlo Tree Search (MCTS) to improve the complex reasoning abilities of LLMs. Additionally, concurrent work Search-r1 [21] and R1-Searcher [43] both use answer-based reward to improve the reasoning in RAG. MAO-ARAG [5] proposes a hierarchical RAG framework that achieves a balance between effectiveness and efficiency

 https://doi.org/10.52202/085713-4047

in RAG by dynamically orchestrating executors through optimizing the planner agent with RL, thereby achieving Pareto optimality. SoftRankPO [52] introduces a meta-cognitive multi-agent reinforcement learning framework that enables LLM agents to dynamically deliberate, coordinate, and improve reasoning performance. Additionally, [28] propose an AI search paradigm which can be trained with RL algorithms.

Some other works use DPO [33] or similar alignment algorithms to optimize LLMs. A noise-filtering method [62] is proposed by optimizing mutual information between compressed data and output while minimizing it with the retrieved passage. ATM [61], an Adversarial Tuning Multi-agent system, enhances the robustness and performance of retrieval-augmented generators in question answering by iteratively tuning against an adversarial attacker agent to better discriminate useful documents and resist fabricated content. SEER [59] proposes a novel self-aligned evidence extraction learning framework aimed at enhancing RAG performance by optimizing the extraction of high-quality, concise, and relevant evidence. RAG-DDR [27] optimizes RAG systems by aligning data preferences between modules through DDR, resulting in enhanced performance on knowledge-intensive tasks.

B How to Construct Training Data for the SFT Training Process

Before the multi-module joint training process of MAPPO, the warm-up to each trainable modules is necessary. The warm-up can make LLMs to follow the instructions better and reduce the exploration space for the MAPPO training process to stabilize and accelerate the joint training process.

In this section, we introduce the details of constructing training data for each modules.

B.1 Query Rewriter

Query Rewriter is to reformulate the initial query q, which may be too complex or ambiguous to resolve with a single retrieval, into a set of sub-questions denoted as $subq$. The prompt of Query Rewriter is in Table 3.

In Rewrite-Retrieve-Read [30], a small language model was trained using PPO to effectively rewrite queries for RAG. Building on this approach, we utilize the publicly available query rewriting data from Rewrite-Retrieve-Read as the SFT dataset to warm start the Query Rewriter in MMOA-RAG.

Table 3: The prompt of Query Rewriter agent.

system: You are a professional assistant skilled at rewriting complex or unclear questions into simpler, more searchable subquestions.

assistant: Okay, I will provide the rewritten sub-questions.

user: Please help me rewrite or decompose the given questions into sub-questions, making them easier to search for answers in a search engine. The rewritten sub-questions must have logical connections and dependencies, without being overly repetitive in meaning. Additionally, avoid using vague demonstrative pronouns and similar terms.

assistant: Okay, I will provide the rewritten sub-questions.

user: Original question is {content of Question}. Now rewrite or decompose the original question into sub-questions according to the above requirements, and only output the rewritten subquestions in the format of one subproblem per line without any additional content. Additionally, avoid using vague demonstrative pronouns and similar terms, avoid the duplicate subquestions.

B.2 Selector

The task of the Selector is to choose a subset D_{selected} that are helpful for answering a question from a given set D with K candidate documents.

The output format of the Selector is the IDs of the documents in D_{selected} (e.g., Document0, Document4, Document6, Document7), as shown in the prompt of Selector in Table 4. Therefore, to construct SFT data for the Selector, the ground truth should be the IDs of documents that are truly useful for answering the question. One method to obtain the ground truth is to employ advanced LLMs, such as GPT-4o, to provide the ground truth. However, we have found that this approach does not yield results as good as expected. Additionally, BGM [22] introduced and optimized a bridge module which is similar to the Selector module. They proposed a method called synthesis silver passage sequence (Synthesis SPS) to construct the ground truth for SFT data. However, the Synthesis SPS method requires examining each candidate document $d_{i,j} \in D_i$ (candidate documents of question i), invoking the LLM for each check, and comparing the utility values before and after the check, making it a complex and costly method.

Table 4: The prompt of Selector agent.

system: You are a helpful, respectful and honest assistant. Your task is to output the IDs of the candidate Documents (0,1,2,...,K-1) which are helpful in answering the Question.

assistant: Okay, I will provide the ID of candidate Documents which are helpful in answering the Question.

user: Question: {content of Question}
Document0: {content of Document0}
Document1: {content of Document1}
⋮
Document(K-2): {content of Document(K-2)}
Document(K-1): {content of Document(K-1)}

assistant: OK, I received the Question and the candidate Documents.

user: Now, output the IDs of the candidate Documents (0,1,2,...,K-1) which are helpful in answering the Question: {content of Question}, for example, in the following format: Document0,Document4,Document6,Document7.

We propose a convenient heuristic approach for constructing SFT data, aimed at LLMs to effectively follow instructions and output in the desired format. As illustrated in Figure 4, for a given question q_i and its golden answer, there are K candidate documents denoted as $d_{i,j}$, where $j \in \{0, 1, \cdots, K-1\}$. First, by removing certain insignificant stop words and punctuation marks from q_i and its golden answer, and converting the words to lowercase, we obtain the set Set_{q_i}. Similarly, we perform the same operation on the K candidate documents $d_{i,j}$ to obtain $Set_{d_{i,j}}$. Finally, if any word from Set_{q_i} appears in $Set_{d_{i,j}}$, the ID of corresponding document j is included in the final output as the Label of SFT. With this approach, we can rapidly and cost-effectively construct the Selector's ground truth labels during the SFT stage. Given our focus on the subsequent joint optimization of multiple modules, this straightforward data construction method can adequately meet our requirements.

B.3 Generator

The Generator is responsible for producing the final answer, Ans_{predict}, based on the D_{selected} provided by the Selector. Therefore, the ground truth for the SFT data of Generator is the golden answer Ans_{golden}. And the prompt of Generator is in Table 5.

https://doi.org/10.52202/085713-4047

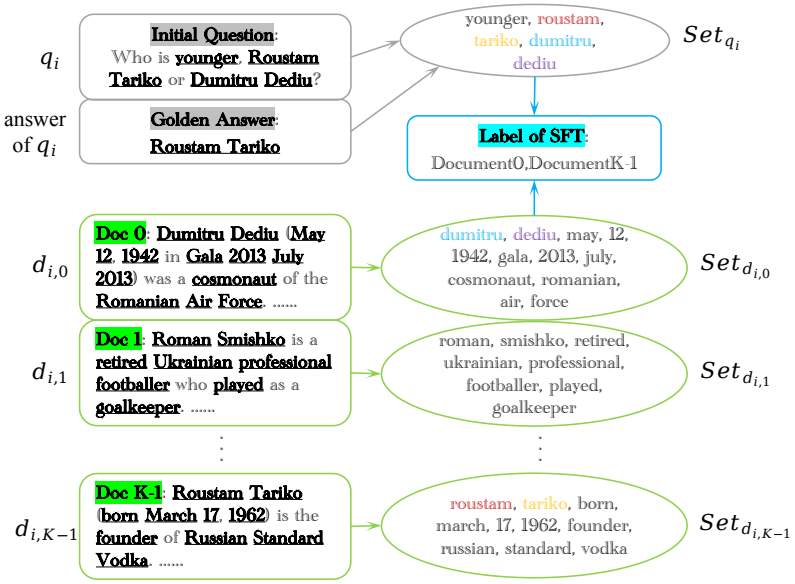


Figure 4: The convenient approach to construct the SFT data for Selector.

Table 5: The prompt of Generator agent.

system: You are a helpful, respectful and honest assistant. Your task is to predict the answer to the question based on the given documents. If you don't know the answer to a question, please don't share false information. Answer the question as accurately as possible.

assistant: Okay, I will provide the answer to the question based on the corresponding documents. Please provide the question and the corresponding documents.

user: Question: {content of Question}
Document0: {content of Document0}
Document3: {content of Document3}
⋮
Document7: {content of Document7}

assistant: OK, I received the Question and the corresponding Documents.

user: Given the Question and the corresponding Documents, predict the answer to the Question as briefly and accurately as possible based on the Documents. Only give the brief and accurate answer with the form of **answer** and nothing else.

With these approaches, the SFT data used for the warm start training of these three modules—Query Rewriter, Selector, and Generator—can be obtained. All modules can be fine-tuned using the typical loss function of SFT presented in Equation (17).

$$\mathcal{L}_{\text{SFT}}(\theta) = -\sum_{n=1}^{N} \log P(Y_i \mid X_i; \theta) \tag{17}$$

In Equation (17), N represents the number of samples in the SFT dataset, while θ denotes the parameters of the LLM. The variable X_i corresponds to the input content of each module. Meanwhile, Y_i signifies the output content of each module.

C The Pseudocode of Multi-Agent Training Process of MMOA-RAG

Algorithm 1 is the pseudocode for multi-agent optimization based on MAPPO.

Algorithm 1: The Training Process of Multi-Agent Optimization

Initialize: The parameters of the Actor model θ, the Critic model ϕ, the SFT model θ_{SFT}, and a replay buffer $\mathcal{M} = \varnothing$.
Inputs: Dataset with initial questions q and corresponding golden answers Ans_{golden}
for $epoch \leftarrow 1$ **to** N_epoch **do**
  for $batch \leftarrow 1$ **to** N_batch **do**
    `// Collect Rollout`
    for *each question* $q \in batch$ **do**
      `// Query Rewriter (QR)`
      Construct observation O_{QR} according to Equation (1)
      Get sub-questions $subq$ for initial question q
      Calculate the penalty term of Query Rewriter P_{QR}
      `// Retriever`
      Retrieve K candidate documents to construct D
      `// Selector (S)`
      Construct observation O_S according to Equation (4)
      Select IDs of helpful documents, and get D_{selected}
      Calculate the penalty term of Selector P_S
      `// Generator (G)`
      Construct observation O_G according to Equation (7)
      Predict the Ans_{predict} to initial question q
      Calculate the penalty term of Generator P_G
      `// Getting Reward and Storing Tuple`
      Calculate the F1 score of Ans_{predict} as the shared reward R_{shared}
      Get reward for each agent $R_i, i \in \{QR, S, G\}$, according Equation (3), (6) and (9)
      Store tuple $\mathcal{T} = ((O_{QR}, subq, R_{QR}), (O_S, IDs, R_S), (O_G, Ans_{\text{predict}}, R_G))$ in the replay buffer $\mathcal{M}$
    `// Policy and Value Optimization`
    for *each question* $q \in batch$ **do**
      Compute the advantage function $\hat{A}^{i,t}_{\pi_\theta}$ using GAE
      Calculate the loss of the Actor $\mathcal{L}_{\text{Actor}}(\theta)$ and Critic model $\mathcal{L}_{\text{Critic}}(\phi)$
      Update the parameters of models through the overall loss function $\mathcal{L}(\theta, \phi)$ in Equation (10)
    Clear the replay buffer $\mathcal{M}$ to $\varnothing$
Output : Well-trained Actor model with parameters θ_{trained}

D The Detailed Introduction of the Implementation Details

D.1 Baselines

The methods detailed below serve as baseline models:

LLM w/o RAG: This approach answers questions solely based on the internal knowledge embedded within LLMs, without employing any retrieval mechanisms.

Vanilla RAG w/o train: This method leverages a retrieval model to obtain relevant documents, thereby augmenting the LLM's internal knowledge with external sources. Here, the LLM remains in a pre-trained state and has not undergone any fine-tuning.

Vanilla RAG w SFT: Building on the Vanilla RAG framework, this variant involves a fine-tuned LLM to improve the integration of retrieved external knowledge with the LLM's internal knowledge, potentially enhancing the quality of final answers.

https://doi.org/10.52202/085713-4047

SELF-RAG [1]: This innovative framework advances LLM performance by incorporating both adaptive retrieval mechanisms and self-reflection processes, aiming to produce precise and dependable answers.

RetRobust [54]: This approach fortifies the RAG architecture against irrelevant contexts, thereby boosting its effectiveness in open-domain question-answering scenarios.

Rewrite-Retrieve-Read [30]: A small-scale query rewriter model is trained using reinforcement learning, optimizing the interaction between retrieval and answer generation.

BGM [22]: Utilizing PPO, this method trains a bridge component to filter and identify documents that are more likely to be helpful, thus refining the quality of the retrieved context.

RAG-DDR [27]: This approach utilize DPO to align data preferences between different modules, enhancing performance and reducing hallucinations in RAG.

D.2 Implementation Details

To ensure fairness in comparison, all baselines were re-implemented using Llama-3-8B-Instruct as the backbone architecture. Within these methods, the untuned modules employed the pre-trained version of Llama-3-8B-Instruct, whereas the trainable modules were derived through specific SFT processes on Llama-3-8B-Instruct. Notably, in the Rewrite-Retrieve-Read framework, the query rewrite module, which is trainable, was optimized using the PPO algorithm applied to Llama-3-8B-Instruct; meanwhile, answer generation was implemented based on the SFT-refined backbone. Regarding the BGM method, we rebuilt the bridge to connect the retrieval model and the generation model, leveraging Llama-3-8B-Instruct, with this bridge being trained using the PPO algorithm. The generation model for BGM was similarly obtained from the SFT-refined backbone.

We utilize Contriever [16] as the Retriever. Regardless of how many sub-questions $subq$ the Query Rewriter generates from the initial question q, the Selector consistently receives a fixed set of $K = 10$ documents as input. For example, if the Query Rewriter yields 2 sub-questions, each sub-question is used for retrieval, with the top-5 documents from each retrieval being selected as part of the candidate documents D for the Selector. Furthermore, it is important to emphasize that we do not utilize any support facts or positive passages [4] that come with the official datasets to generate answers. Instead, we only use the K candidate documents from the retrieval model as external knowledge for answer generation.

Besides, we also employ Llama-3-8B-Instruct [7] as the foundational LLM for the baselines and MMOA-RAG. Building on the PPO code from LLama-Factory[5] [60], we have developed MMOA-RAG, which optimizes the RAG multi-agent system using Multi-Agent PPO. And the critical hyperparameters of MMOA-RAG are detailed in Table 6.

Table 6: Key hyperparameters in the training process of MMOA-RAG.

Name	Explanation	Values
β_{max}	Maximum β in Equation (15)	0.2
β_{min}	Minimum β in Equation (15)	0.06
γ	Key hyperparameter in GAE	1.0
λ	Key hyperparameter in GAE	0.95
ϵ	Clip range in MAPPO	0.2
α	Coefficients in Equation (10)	0.1
lr	Maximum learning rate	2e-5
`bueffer_size`	Buffer size in MAPPO	128
`lr_scheduler`	Learning rate scheduler	cosine
`top_p`	Sampling parameters in training	0.9

[4] Some QA datasets inherently include supportive texts that aid in answering questions. And some studies incorporate these supportive texts alongside retrieved candidate documents as input to LLMs for answer prediction, which significantly enhances answer quality. However, we adhere to the natural RAG process by using only the candidate documents provided by the retriever for answer prediction, as annotated supportive texts are not present in the natural RAG workflow.

[5] `https://github.com/hiyouga/LLaMA-Factory`

Table 7: The performance of different methods based on Retriever BGE [47] and E5 [49]. Llama-3-8B-Instruct and its the fine-tuned version is the backbone for all methods. The highest baseline value in each dataset is underscored. The Δ displays the improvement of MMOA-RAG over the best baseline.

Methods	HotpotQA			2WikiMultihopQA			AmbigQA		
	Acc	EM	F1	Acc	EM	F1	Acc	EM	F1
Retriever: BGE [47]									
Vanilla RAG w/o train	37.72	28.99	41.62	28.86	18.54	27.87	48.05	35.10	52.31
Vanilla RAG w SFT	45.47	41.16	54.32	42.79	41.50	46.70	47.10	42.45	57.41
Self-RAG [1]	33.41	30.59	42.12	37.39	36.41	40.91	30.00	27.00	40.77
RetRobust [54]	46.66	42.74	55.77	47.03	**45.57**	50.62	47.00	42.20	58.14
Rewrite-Retrieve-Read [30]	45.58	40.95	54.35	43.91	42.48	47.83	46.30	41.75	57.19
BGM [22]	45.50	41.49	54.73	41.88	40.67	45.53	48.15	42.70	58.61
RAG-DDR [27]	45.13	41.51	54.32	42.49	41.21	46.00	48.50	42.95	58.71
MMOA-RAG	**47.22**	**43.46**	**56.45**	**47.14**	45.47	**50.94**	**48.65**	**43.45**	**59.10**
Δ	**+0.56**	**+0.72**	**+0.68**	**+0.11**	-0.10	**+0.32**	**+0.15**	**+0.50**	**+0.39**
Retriever: E5 [49]									
Vanilla RAG w/o train	36.92	28.01	40.80	28.57	17.50	27.67	**50.50**	37.00	53.93
Vanilla RAG w SFT	46.48	42.08	55.51	44.74	43.32	48.66	48.20	43.40	58.53
Self-RAG [1]	33.64	30.84	42.36	38.39	37.38	42.03	30.10	27.20	40.83
RetRobust [54]	46.66	42.78	55.85	49.27	**47.59**	52.75	48.85	44.10	59.29
Rewrite-Retrieve-Read [30]	46.62	42.00	55.54	46.41	44.83	50.35	47.85	43.55	58.36
BGM [22]	45.81	41.72	55.32	43.24	41.99	46.83	48.35	44.05	59.04
RAG-DDR [27]	44.50	41.09	53.96	43.89	42.49	47.29	49.60	43.75	59.75
MMOA-RAG	**47.46**	**43.73**	**57.23**	**49.32**	47.53	**53.13**	50.45	**44.80**	**60.80**
Δ	**+0.80**	**+0.95**	**+1.38**	**+0.05**	-0.06	**+0.38**	-0.05	**+0.70**	**+1.05**

E The Performance of Different Methods Based on Retriever BGE and E5

We firstly perform the training of different methods based on retrieval model Contriever [16], and the results are shown in Table 1.

We further test the trained models on different retriever BGE [47] and E5 [49]. From Table 7, we can see that our MMOA-RAG also surpasses most of baselines on each datasets.

F Out-of-Domain Experiments

We also conducted out-of-domain (OOD) experiments to evaluate the generalization capabilities of MMOA-RAG compared to the baselines. We trained LLM on HotpotQA dataset and evaluated on the AmbigQA dataset. The experimental results are shown in Table 8.

Table 8: Out-of-domain experimental results: The model is trained on the HotpotQA dataset and tested on the AmbigQA dataset.

Methods	Acc	EM	F1
SELF-RAG [1]	26.70	24.25	36.38
RetRobust [54]	34.19	31.75	44.08
Rewrite-Retrieve-Read [30]	33.91	30.73	43.61
BGM [22]	32.58	29.77	42.07
MMOA-RAG (ours)	**35.45**	**32.43**	**45.62**

From Table 8, it is evident that MMOA-RAG demonstrates superior performance in the OOD experiments, underscoring its notable generalization capabilities and effectively. Additionally, it is noteworthy that RetRobust outperforms all other baselines. This can be attributed to its strategy of integrating both relevant and irrelevant data during the SFT process, which significantly enhances the robustness and generalization abilities of the RAG system.

 https://doi.org/10.52202/085713-4047

G Discussion about Training Time and Inference Time

Training Time In our MMOA-RAG framework, multiple agents (modules) are optimized simultaneously, which results in increased training time as the number of agents rises. When we have n agents, the time required for rollouts and updating model parameters becomes n times that of a single agent. However, since this increase in training time is linear relative to the number of agents, it does not lead to an excessively large overhead. Therefore, this is entirely manageable.

Inference Time The inference process of MMOA-RAG follows the workflow outlined in Figure 1. Training with multiple agents does not introduce any additional overhead during the inference phase.

H Reasons for Using Reward Sharing and Parameter Sharing Strategies

H.1 Reward sharing

It is through the way of reward sharing that the optimization objectives of multiple modules can be unified and aligned to optimize the quality of the final predicted answer. Multiple modules in MMOA-RAG naturally fit into a fully cooperative relationship, and reward sharing is a near-standard setting in fully cooperative MARL algorithms [34, 29, 55, 3], as fully cooperative modeling only allows all agents to have the same goal (here, F1 score). Therefore, reward sharing is reasonable and necessary here.

H.2 Parameter sharing

Parameter sharing is also a common and effective strategy in multi-agent reinforcement learning [34, 29, 55, 3]. Parameter sharing means that multiple agents share the parameters of the same policy network instead of maintaining a separate policy network for each agent. The biggest advantage of this approach is that it can reduce the computational and storage overhead: sharing parameters can significantly reduce the total number of parameters of the model, thus reducing the computational complexity and storage requirements. This is particularly important for RAG systems with multiple modules based on LLM, which is an important reason why we chose to use the parameter sharing strategy.

However, it is undeniable that parameter sharing does have its limitations: it can lead to instability when the number of agents increases. However, in our framework, there are three agents (modules) that are jointly optimized, and the problem of instability generally does not occur when the number of agents is small. The experimental results also prove that under the setting of parameter sharing, the functions of Query Rewriter, Selctor and Generator are not affected at all, and they can complete the task well and exceed the baselines without parameter sharing such as RRR and BGM.

To summarize, we adopt the Settings of parameter sharing and reward sharing commonly used in previous MARL frameworks, which are common and versatile multi-agent reinforcement learning frameworks. Agents are distinguished by different prompts, which correspond to different observations of different agents. Therefore, there doesn't have to be a distinction at the architectural or parameter level to be called different agents.

I Case Study

In this section, we analyze the question from the HotpotQA dataset: "The Vermont Catamounts men's soccer team currently competes in a conference that was formerly known as what from 1988 to 1996?" The golden answer to this question is "the North Atlantic Conference." We compare two inference cases, Inference Case 1 involves the response process of MMOA-RAG after MAPPO training, while Inference Case 2 pertains to the response process before MAPPO training, where only SFT was conducted:

- In Inference Case 1, we observe that the LLM reformulates the initial question q into two sub-questions $subq$. In Candidate Documents D, Document 0 to 4 are retrieved based on Sub-question 1, and Documents 5 to 9 are retrieved based on Sub-question 2. Notably, the answer highlighted in red appears in Document 5, which the Selector accurately identifies. The

final answer generated based on the Initial question q and Document 5 is "North Atlantic Conference," consistent with the Golden answer Ans_{golden}. This process effectively demonstrates how, after MAPPO training, multiple modules can collaborate accurately to provide the most precise answer possible.

- In contrast, Inference Case 2 shows that the LLM only reformulates the initial question q into a single Sub-question 1. All 10 candidate documents are retrieved based on this Sub-question 1. Upon inspection, none of these documents contain text that can answer the initial question. The Selector also chooses multiple documents, which clearly does not aid in generating the correct answer. Consequently, the predicted answer Ans_{predict} is the incorrect "Yankee Conference." Hence, prior to MAPPO training, despite each module being well-optimized through SFT, there is a lack of effective collaboration among the modules, potentially leading to inaccurate responses.

These examples provide an intuitive understanding of the advantages of joint multi-module optimization within the complex RAG framework.

Inference Case 1: Training with MAPPO

- **Initial question** q:
  - The Vermont Catamounts men's soccer team currently competes in a conference that was formerly known as what from 1988 to 1996?
- **Golden answer** Ans_{golden}:
  - the North Atlantic Conference
- **Sub-questions** $subq$:
  - Sub-question 1: name of the conference that the Vermont Catamounts men's soccer team currently competes in
  - Sub-question 2: name of the conference from 1988 to 1996
- **Candidate Documents** D:
  - **Document 0:**
    - Title: Vermont Catamounts football
    - Content: The Vermont Catamounts football program were the intercollegiate American football team for the University of Vermont located in Burlington, Vermont. The team competed in the NCAA Division I and were members of the Yankee Conference. The school's first football team was fielded in 1886. The football program was discontinued at the conclusion of the 1974 season. Vermont fields a team at the club football level, in a conference that also uses the Yankee Conference name. The first intercollegiate game in the state of Vermont happened on November 6, 1886 between Dartmouth and Vermont in Burlington. Dartmouth won
  - **Document 1:**
    - Title: Vermont Catamounts men's ice hockey
    - Content: The Vermont Catamounts men's ice hockey team is a National Collegiate Athletic Association (NCAA) Division I college ice hockey program that represents the University of Vermont. The Catamounts are a member of Hockey East, joining in 2005 after competing in ECAC Hockey from 1974-2005. They play home games at Gutterson Fieldhouse in Burlington, Vermont. Vermont has appeared in the NCAA Men's Hockey Championship five times since making the move to Division I in 1974-75 including trips to the Frozen Four in 1996 and 2009. Prior to moving to Division I, UVM competed in ECAC Division
  - **Document 2:**
    - Title: Vermont Catamounts football
    - Content: Members of the conference. Vermont began Yankee Conference play in 1947 with Connecticut, Maine, Massachusetts, New Hampshire, and Rhode Island. Although they played UMass and UNH in the first season, they didn't play Maine until 1950, Rhode Island until 1955, and UConn until 1966. Boston University began league play in 1973. Notable alumni include: Vermont Catamounts football The Vermont Catamounts football program were the intercollegiate American football team for the University of Vermont located in Burlington, Vermont. The team competed in the NCAA Division I and were members of the Yankee Conference. The school's first football team was fielded in
  - **Document 3:**
    - Title: Vermont Catamounts men's basketball
    - Content: The Vermont Catamounts Basketball team is the basketball team that represents the University of Vermont in Burlington, Vermont. The school's team currently competes in the America East Conference and plays its home games at Patrick Gym. The team has reached the NCAA Division I Men's Basketball Tournament six times, in 2003, 2004, 2005, 2010, 2012, and 2017. UVM famously upset Syracuse University in the first round of the 2005 tournament. The Catamounts are coached by John Becker. America East Coach of the Year America East Player of the Year America East Defensive Player of the Year
  - **Document 4:**
    - Title: Vermont Catamounts
    - Content: The Vermont Catamounts are the varsity intercollegiate athletic programs of the University of Vermont, based in Burlington, Vermont, United States. The school sponsors 18 athletic programs (8 men's, 10 women's), most of which compete in the NCAA Division I America East Conference (AEC), of which the school has been a member since 1979. The men's and women's ice hockey programs compete in Hockey East. The men's and women's alpine and nordic skiing teams compete in the Eastern Intercollegiate Ski Association (EISA). The school's athletic director is Robert Corran. The Catamounts have won six national championships, all in skiing.

https://doi.org/10.52202/085713-4047

- **Document 5:**
  - Title: America East Conference
  - Content: The America East Conference is a collegiate athletic conference affiliated with the NCAA Division I, whose members are located mainly in the Northeastern United States, specifically New England. Its nine members include the public flagship universities of three states, and one private university. The America East Conference was founded as the Eastern College Athletic Conference-North, a men's basketball-only athletic conference in 1979. The conference was known as the Eastern College Athletic Conference-North from 1979 to 1988 and **the North Atlantic Conference from 1988 to 1996**. The charter members were the University of Rhode Island, the College of
- **Document 6:**
  - Title: Northwest Conference
  - Content: The Northwest Conference (NWC) is an athletic conference which competes in the NCAA's Division III. Member teams are located in the states of Oregon and Washington. The NWC was formed in 1926, making it one of the oldest athletics conferences in the western United States. For 60 years, the Northwest Conference sponsored sports exclusively for men, but in 1984 it joined with the Women's Conference of Independent Colleges to become the Northwest Conference of Independent Colleges, shortening the name to its current moniker in 1996 when it joined the NCAA. The charter members included Willamette University, Pacific University,
- **Document 7:**
  - Title: Big West Conference
  - Content: The Big West Conference (BWC) is an American collegiate athletic conference whose member institutions participate in the National Collegiate Athletic Association's Division I. The conference was originally formed in 1969 as the Pacific Coast Athletic Association (PCAA) and in 1988 was renamed the Big West Conference. The conference stopped sponsoring college football after the 2000 season. The Big West Conference was formed in June 1968 as the Pacific Coast Athletic Association. The five original charter members were Fresno State, San Jose State, UC Santa Barbara, San Diego State, and Long Beach State. Two other schools, Cal State
- **Document 8:**
  - Title: Big Eight Conference Men's Basketball Tournament
  - Content: The Big Eight Conference was an NCAA athletic conference that existed from 1907 to 1996, when it and the Southwest Conference disbanded to create the Big 12 Conference. The post-season conference tournament was instituted in 1977 and from that time the winner won the conference's automatic NCAA Tournament bid. From 1977 until 1985, the quarterfinals were played on the campus sites of the higher seeded teams. The last Big Eight men's basketball tournament ran through the conference's final season in 1996. Missouri won the most Big Eight tournament titles with six. Colorado was the
- **Document 9:**
  - Title: Big South Conference Men's Basketball Tournament
  - Content: The Big South Conference Men's Basketball Tournament (popularly known as the Big South Tournament) is the conference championship tournament in basketball for the Big South Conference. The tournament has been held every year since 1986. It is a single-elimination tournament and seeding is based on regular season records. The winner, declared conference champion, receives the conference's automatic bid to the NCAA men's basketball tournament. However, the conference did not have an automatic bid to the NCAA tournament from 1986 to 1990. From 2003 through 2012, the tournament was held mostly at campus sites. In

- **IDs of Selected Documents** D_{selected}:
  - Document 5
- **Predicted answer** Ans_{predict}:
  - North Atlantic Conference ✓

Inference Case 2: Training with SFT

- **Initial question** q:
  - The Vermont Catamounts men's soccer team currently competes in a conference that was formerly known as what from 1988 to 1996?
- **Golden answer** Ans_{golden}:
  - the North Atlantic Conference
- **Sub-questions** $subq$:
  - Sub-question 1: name of the conference that the Vermont Catamounts men's soccer team currently competes in
- **Candidate Documents** D:
  - **Document 0:**
    - Title: Vermont Catamounts football
    - Content: Vermont Catamounts football The Vermont Catamounts football program were the intercollegiate American football team for the University of Vermont located in Burlington, Vermont. The team competed in the NCAA Division I and were members of the Yankee Conference. The school's first football team was fielded in 1886. The football program was discontinued at the conclusion of the 1974 season. Vermont fields a team at the club football level, in a conference that also uses the Yankee Conference name. The first intercollegiate game in the state of Vermont happened on November 6, 1886 between Dartmouth and Vermont in Burlington. Dartmouth won
  - **Document 1:**

    - Title: Vermont Catamounts men's ice hockey
    - Content: Vermont Catamounts men's ice hockey The Vermont Catamounts men's ice hockey team is a National Collegiate Athletic Association (NCAA) Division I college ice hockey program that represents the University of Vermont. The Catamounts are a member of Hockey East, joining in 2005 after competing in ECAC Hockey from 1974-2005. They play home games at Gutterson Fieldhouse in Burlington, Vermont. Vermont has appeared in the NCAA Men's Hockey Championship five times since making the move to Division I in 1974-75 including trips to the Frozen Four in 1996 and 2009. Prior to moving to Division I, UVM competed in ECAC Division
- **Document 2:**
    - Title: Vermont Catamounts men's basketball
    - Content: Vermont Catamounts men's basketball The Vermont Catamounts Basketball team is the basketball team that represents the University of Vermont in Burlington, Vermont. The school's team currently competes in the America East Conference and plays its home games at Patrick Gym. The team has reached the NCAA Division I Men's Basketball Tournament six times, in 2003, 2004, 2005, 2010, 2012, and 2017. UVM famously upset Syracuse University in the first round of the 2005 tournament. The Catamounts are coached by John Becker. America East Coach of the Year America East Player of the Year America East Defensive Player of the Year
- **Document 3:**
    - Title: Vermont Catamounts football
    - Content: members of the conference. Vermont began Yankee Conference play in 1947 with Connecticut, Maine, Massachusetts, New Hampshire, and Rhode Island. Although they played UMass and UNH in the first season, they didn't play Maine until 1950, Rhode Island until 1955, and UConn until 1966. Boston University began league play in 1973. Notable alumni include: Vermont Catamounts football The Vermont Catamounts football program were the intercollegiate American football team for the University of Vermont located in Burlington, Vermont. The team competed in the NCAA Division I and were members of the Yankee Conference. The school's first football team was fielded in
- **Document 4:**
    - Title: Vermont Catamounts
    - Content: Vermont Catamounts The Vermont Catamounts are the varsity intercollegiate athletic programs of the University of Vermont, based in Burlington, Vermont, United States. The school sponsors 18 athletic programs (8 men's, 10 women's), most of which compete in the NCAA Division I America East Conference (AEC), of which the school has been a member since 1979. The men's and women's ice hockey programs compete in Hockey East. The men's and women's alpine and nordic skiing teams compete in the Eastern Intercollegiate Ski Association (EISA). The school's athletic director is Robert Corran. The Catamounts have won six national championships, all in skiing.
- **Document 5:**
    - Title: 2018–19 Vermont Catamounts men's basketball team
    - Content: to win their conference tournament, the Catamounts received an automatic bid to the National Invitation Tournament, where they lost to Middle Tennessee in the First Round. !colspan=12 style=| Exhibition !colspan=12 style=| Non-conference regular season !colspan=9 style=| America East Conference regular season !colspan=12 style=| America East Tournament Source 2018–19 Vermont Catamounts men's basketball team The 2018–19 Vermont Catamounts men's basketball team will represent the University of Vermont in the 2018–19 NCAA Division I men's basketball season. They will play their home games at the Patrick Gym in Burlington, Vermont and will be led by 8th-year head coach John Becker. The Catamounts
- **Document 6:**
    - Title: Vermont
    - Content: based in Burlington. They were named the Vermont Expos before 2006. Up until the 2011 season, they were the affiliate of the Washington Nationals (formerly the Montreal Expos). Currently the highest teams in basketball, representing Vermont are the NCAA's Vermont Catamounts – male and female. The Vermont Frost Heaves, the 2007 and 2008 American Basketball Association national champions, were a franchise of the Premier Basketball League, and were based in Barre and Burlington from the fall of 2006 through the winter of 2011. The Vermont Bucks, an indoor football team, were based in Burlington and began play in 2017 as
- **Document 7:**
    - Title: 2017–18 Vermont Catamounts men's basketball team
    - Content: 2017–18 Vermont Catamounts men's basketball team The 2017–18 Vermont Catamounts men's basketball team represented the University of Vermont during the 2017–18 NCAA Division I men's basketball season. The Catamounts, led by seventh-year head coach John Becker, played their home games at Patrick Gym in Burlington, Vermont as members of the America East Conference. They finished the season 27–8, 15–1 in America East play to win the America East regular season championship. The Catamounts defeated Maine and Stony Brook to advance to the championship game of the America East Tournament where they lost to UMBC. As a regular season conference champion
- **Document 8:**
    - Title: Vermont Catamounts
    - Content: basketball team has performed well in the America East Tournament. In 2013, it made a conference-record 18th appearance in the semifinals. It has the most wins in tournament play, with 35 (2013 field). It has advanced at least one round in 19 of the 24 tournaments. The Catamounts were the first women's basketball program to go undefeated during the regular season in back-to-back seasons (1991-1992 and 1992-1993), a feat matched only by Connecticut (2008-2009 and 2009-2010). In the 2012-2013 season, the program had a total attendance mark of 10,579. After the 2009 season, the Vermont baseball program, which played at
- **Document 9:**
    - Title: 2018–19 Vermont Catamounts men's basketball team
    - Content: 2018–19 Vermont Catamounts men's basketball team The 2018–19 Vermont Catamounts men's basketball team will represent the University of Vermont in the 2018–19 NCAA Division I men's basketball season. They will play their home games at the Patrick Gym in Burlington, Vermont and will be led by

 https://doi.org/10.52202/085713-4047

8th-year head coach John Becker. The Catamounts finished the 2017–18 season 27–8, 15–1 in the America East Conference play to finish in first place. In the America East Tournament, they defeated Maine and Stony Brook to advance to the championship game, where they lost to UMBC. As a regular season conference champion who failed

- **IDs of Selected Documents** D_{selected}:
  - Document 0, Document 2, Document 3, Document 4, Document 5, Document 6, Document 8, Document 9
- **Predicted answer** Ans_{predict}:
  - Yankee Conference ✗

https://doi.org/10.52202/085713-4047