TRIO: Task Refinement with Individual Optimization for Retrieval-Augmented Generation

Yingtao Ren[1], Xiao Luo[2], Yiwei Fu[3], Jinzhao Zhou[1], Yu-Cheng Chang[1], Chin-Teng Lin[1],
[1]University of Technology Sydney, [2]University of Wisconsin–Madison, [3]Peking University
Correspondence: chin-teng.lin@uts.edu.au

Abstract

This paper studies retrieval-augmented generation (RAG), which leverages external knowledge to improve the performance of language models. Despite the remarkable progress, current RAG approaches are still far from satisfactory due to inadequate retrieval and potential hallucinations. Towards this end, we propose a novel approach named Task Refinement with Individual Optimization (TRIO) for RAG. The core idea of our TRIO is to equip three well-designed agents with an iterative refinement paradigm by aligning their individual optimization objectives. Specifically, we first expand ambiguous queries with a query extension agent and quantify the reward based on the improved answer quality, which can be utilized to refine our agent for higher expansion efficiency. To enhance the retrieval quality, we include a unanimity voting to annotate the expanded query as insufficient or sufficient, and their generative outcomes are utilized as ground truth to optimize the judge agent. To further improve evidence utilization and answer accuracy, an answer agent is optimized with dynamic matching-based rewards with curriculum learning for final outputs. Extensive experiments across multiple benchmark datasets validate the effectiveness of the proposed TRIO in comparison with state-of-the-art approaches. Our code is available at https://github.com/yingtaoren/TRIO.

1 Introduction

Large language models (LLMs) have achieved remarkable success in natural language understanding and reasoning. By pre-training on a series of high-quality instruction datasets, LLMs can acquire a wide range of factual knowledge (Achiam et al., 2024; Du et al., 2026). These pre-trained models have enabled a wide range of applications beyond conventional language tasks (Xiao et al., 2025; Fu et al., 2026). Nevertheless, pre-trained models may fail to capture the most up-to-date information in highly specialized domains, particularly those involving private or sensitive knowledge, such as healthcare and law (Huang et al., 2025). This issue limits LLMs' capabilities on some knowledge-intensive tasks, even leading to hallucinations.

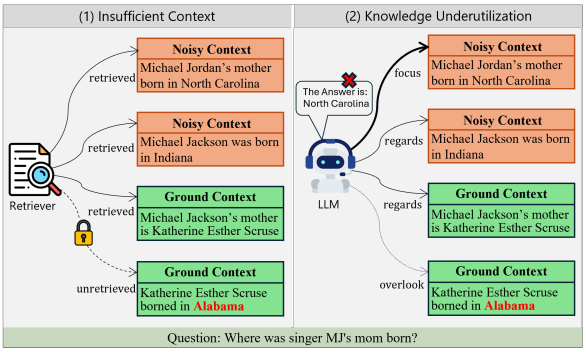


Figure 1: Problem illustration. RAG struggles with inefficient knowledge acquisition in the retrieval stage and knowledge underutilization in the answer stage.

To enhance the response quality, RAG has been employed as a framework that leverages external knowledge to reduce reliance on the pre-trained LLMs' parametric knowledge (Wang et al., 2026; Gu et al., 2026). Typical RAG systems include two main steps: retrieval and generation. In the retrieval step, the key point is to locate the lacking knowledge and define a retrieval query to retrieve the knowledge from an external corpus. And then, in the generation stage, the retrieved contents are integrated with the original question and fed into the LLM. The augmented input provides the LLM with relevant knowledge and the latest information, thereby enhancing the quality of responses and reducing hallucinations (Mishra et al., 2024). The main advantage of RAG is that it provides external contextual information to answer questions when combined with the model's parametric knowledge (Yu et al., 2025). However, this advantage is constrained by two challenges. First, the system may not be able to accurately identify the missing knowledge and generate an effective query for retrieval. An inefficient query formulation may lead

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 19765–19782
October 24-29, 2026 ©2026 Association for Computational Linguistics

to the retrieval of numerous irrelevant or even biased documents, thereby increasing the model's reasoning burden (Yu et al., 2024a; Cuconasu et al., 2024). Second, even when relevant documents are retrieved, the model may be unable to appropriately utilize the provided context, which may lead to further hallucinations (Hsieh et al., 2024; Liu et al., 2024; Wang et al., 2025a; Ren et al., 2026b).

Existing approaches to optimizing the pipeline of RAG include the branching method (Kim et al., 2024; Shi et al., 2024), context ranking (Yu et al., 2024b), and iterative retrieval (Asai et al., 2024; He et al., 2024). These approaches merely adjust the pipeline of RAG, remaining insufficient to counteract the noise or bias introduced by retrieved knowledge. Furthermore, many recent works fine-tune specific LLMs for the RAG pipeline to improve robustness and factual consistency in the generation stage. Such efforts include supervised fine-tuning with human-annotated datasets (Niu et al., 2024) and utilizing reinforcement learning to integrate reasoning ability with retrieval (Song et al., 2025). However, by placing the burden of generating retrieval queries and answering on the same backbone model, these approaches inherit the models' intrinsic biases, limiting their robustness and ability to generate effective queries.

Recently, a growing trend in LLMs is to break complex processes into multiple specialized agents or modules that work (Chen et al., 2024; Luo et al., 2025) collaboratively. Instead of uniform models handling everything, each agent is responsible for a specific subtask in multi-agent systems (Bo et al., 2024). This paradigm has also been introduced into the RAG framework, resulting in a remarkable improvement in removing hallucinations and effectively utilizing external knowledge (Nguyen et al., 2025; Hu et al., 2025). However, existing multi-agent RAG frameworks still share the same backbone without adaptive adjustment, which potentially undermines response reliability.

To address these challenges, we propose a multi-agent system with Task Refinement and Individual Optimization (TRIO), an innovative multi-agent collaborative RAG system that tightly couples the retrieval and generation stages to improve retrieval and answer quality. Unlike existing multi-agent RAG approaches that utilize uniform backbones (Hu et al., 2025), TRIO assigns task-adaptive agents to distinct subtasks, allowing each agent to specialize in its role. This targeted specialization enables the system to pinpoint missing knowledge and utilize retrieved knowledge more effectively, yielding higher answer quality. Because each agent operates on a narrow slice of the pipeline, TRIO also lowers training cost and improves reliability.

Our TRIO implements an adaptive multi-agent RAG framework that decomposes the workflow into three subtasks: retrieval query generation, sufficient information judgment, and answer generation, each subtask handled by an optimized expert agent. This adaptive design enables effective handling of diverse questions and yields robust performance across domains. In addition, we present a streamlined data-generation and training protocol that tightly couples the optimization of all three agents, significantly improving effectiveness while maintaining modest supervision overhead. Crucially, the training data construction and fine-tuning procedures embed substantial cross-agent collaboration, which strengthens coordination within the multi-agent system and tightly couples the subtasks of the pipeline. The protocol facilitates a straightforward transfer to a wide range of RAG scenarios and provides a practical path to enhance real-world RAG deployments. Our main contributions can be summarized as follows:

(1) Existing RAG systems struggle with inefficient knowledge acquisition and hallucination issues. We address these challenges through TRIO, which strategically assigns specialized agents to distinct RAG pipeline stages.

(2) Our TRIO introduces novel data generation methods that expand limited supervised data to align across different subtasks of RAG and further adopts adaptive fine-tuning paradigms for each agent. This design helps enhance retrieval efficiency and answer quality.

(3) Comprehensive experiments across the latest benchmarks demonstrate that our TRIO outperforms a range of competitive baselines. In addition, the framework can be readily extended to various RAG scenarios.

2 Related Work

Retrieval-Augmented Generation. Large language models have achieved strong performance across diverse tasks (Chen et al., 2026). However, they remain susceptible to hallucinations and insufficient context. Retrieval augmented generation addresses this limitation by retrieving external evidence and conditioning the model's output on

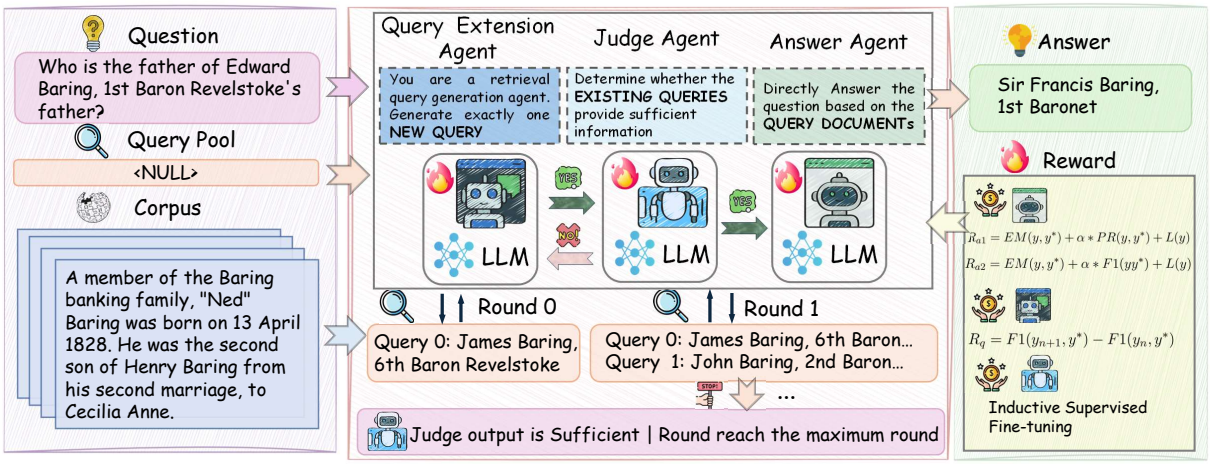


Figure 2: An overview of our TRIO framework. The tight cooperation among three specific optimized agents improves the retrieval efficiency and answer quality.

the retrieved context (Chen et al., 2025; Fan et al., 2024). Early RAG pipelines use the original input as a query to search on an external corpus, and then integrate the retrieval context into the model's input as supplementary knowledge (Gao et al., 2024). To meet more complex retrieval needs, many methods introduce iterative retrieval generation branching (Kim et al., 2024) or loops (Yoran et al., 2024; Chan et al., 2024; Jiang et al., 2023; Shao et al., 2023), letting the model adapt its retrieval strategy when evidence is insufficient. With the rise of chain-of-thought prompting (Zhang et al., 2025b), reasoning traces have also been used to improve retrieval efficiency and recall (Li et al., 2025; Trivedi et al., 2023). Some multi-agent methods have also been explored in RAG to enhance retrieval quality (Hu et al., 2025; Nguyen et al., 2025). Despite these advances, most existing methods still share a single backbone model across all roles in the RAG pipeline and lack targeted fine-tuning, which limits further performance gains.

Multi-Agent System. Recent work shows that several agents' interactions can substantially enhance LLM capabilities across three main regimes (Agashe et al., 2025). First, debate-based frameworks run adversarial dialogues or critiques in which agents challenge each other to refine reasoning and factuality (Du et al., 2024; Zhang et al., 2025a). Second, ensemble coordination runs multiple agents in parallel with minimal communication and aggregates their outputs via voting, validation, or game-theoretic consensus (Wang et al., 2025b; Yue et al., 2024). Third, role-specialized collaboration assigns complementary roles to decompose problems and iteratively refine solutions (Wei et al., 2025; Liu et al., 2025). However, the multi-agent setting for RAG remains underexplored. Contemporary RAG pipelines typically rely on a single LLM to both retrieve and generate, with limited support for collaboration among specialized agents.

3 The Proposed TRIO

Problem Definition. Retrieval augmented generation enhances large language models by integrating external knowledge as context that is not stored in the model's parameters. Given a collection of retrieval text chunks C and a corresponding retrieval query Q, which was identified by the LLM to obtain knowledge from an external corpus. If necessary, it can perform multiple retrieval rounds to acquire additional knowledge. However, excessive context may introduce noise and increase the risk of hallucination. Our objective is to build an efficient multi-agent RAG system, where each agent is specialized and fine-tuned to a specific subtask within the RAG pipeline. This multi-agent RAG system must operate as a tightly coupled loop, collaboratively formulating retrieval queries targeting missing evidence in an external corpus to answer question x while minimizing irrelevant content, and then seamlessly utilizing the acquired context to generate accurate responses. The central problem is training multiple agents to cooperate across retrieval, judgment, and answer generation so they function as an effective whole.

3.1 Framework Overview

We address a limitation of existing RAG systems that rely on uniform LLMs across all stages, leading to weak coordination among components. In TRIO, specialized agents are trained to adapt to their respective subtasks while engaging in substantial cross-agent cooperation. This training strat-

egy strengthens coordination within the multi-agent RAG system, tightly couples the subtasks, and improves robustness and factual consistency. In particular, during the construction of training data and the training process, the query agent is trained to retrieve evidence aligned with the preferences of the answer agent, making it easier to produce accurate and unbiased answers. The judge agent is trained to understand the other agents' behaviors and capabilities, enabling more reliable decisions about whether the current retrieval evidence is sufficient. The overall optimization loop is summarized in Algorithm 1.

3.2 Answer Agent Optimization with Matching-based Reinforcement Learning

The answer agent is responsible for using the retrieved content to answer questions with high accuracy. The primary goal for training the answer agent is to enhance the utilization of retrieval content and avoid hallucinations. To train the agent, we build a training corpus that spans different numbers of retrieval steps and various chunk set sizes, enabling the agent to learn to ignore biases in the context and focus on the key evidence.

We sample training questions for the answer agent from multi-hop QA datasets, assigning difficulty labels (easy/medium/hard) and retaining medium and hard questions following (Song et al., 2025). We then construct a retrieval-only debate pipeline (Hu et al., 2025) that iteratively refines retrieval queries and retrieved contents, capping the debate at three rounds. We store the query pool and retrieved chunks at the end of each debate round, treating each snapshot as a training sample. For each question, we generate multiple training samples per question by varying the number of collected queries and chunks. This setup trains the agent to filter noise and focus on key evidence across different context sizes. Further details on the debate pipeline are provided in Appendix C.

To improve the generative capabilities of the answer agent, we propose a two-stage outcome-based reinforcement learning (RL) training method based on the GRPO algorithm (Guo et al., 2025). The model is first trained to utilize the provided information and minimize internal hallucinations, using questions with the top 3 retrieval chunks for each retrieval query. Specifically, an outcome-based reward consists of an exact match (EM) score and a precision score, which are used in this stage. The EM is defined as follows:

$$EM(y, y^*) = \begin{cases} 1, & \text{if the } y = y^* \\ 0, & \text{else,} \end{cases} \quad (1)$$

where y is the predicted answer generated by the answer agent and $y*$ is the gold answer of the input question x. The precision score is:

$$PR(y, y^*) = \frac{IN(y, y^*)}{PN(y)}, \quad (2)$$

where the $PN(y)$ represents the word count of the answer, IN indicates the word count of the intersection between the two answers. To further discourage explanations and other extraneous content in the answer, we add a format term to the reward:

$$L(y) = \begin{cases} -1, & \text{if the length of } y \text{ larger than } l \\ 0, & \text{else,} \end{cases} \quad (3)$$

where l is a tuned hyper-parameter. Therefore, the final reward R_{a1} of the stage is the sum of the outcome metrics and format metrics:

$$R_{a1} = EM(y, y^*) + \alpha * PR(y, y^*) + L(y), \quad (4)$$

where α is the tuned hyper-parameter.

Subsequently, the answer agent is trained to effectively locate key evidence in richer contexts and to explore broader reasoning paths, using questions with the top 5 retrieval chunks for each retrieval query. The outcome-based reward comprises an EM score and an F1 score:

$$R_{a2} = EM(y, y^*) + \alpha * F1(y, y^*) + L(y), \quad (5)$$

where the token-level $F1$ score is:

$$F1(y, y^*) = \frac{2 * IN(y, y^*)}{PN(y) + PN(y^*)}. \quad (6)$$

3.3 Query Extension Agent Optimization with Outcome-based Reward

The goal of retrieval-query generation in RAG is to identify gaps in the current context and generate new retrieval queries, reducing whole-context bias and thereby providing the key evidence needed by the answer agent. Existing methods struggle with a lack of a clear signal to judge the quality of a single query. In TRIO, we train an adaptive query extension agent to improve retrieval quality based on the answer agent's generation quality.

We first use each original question as a retrieval query to retrieve the corresponding training sample, resulting in a query-refinement dataset. Utilizing the strong answer agent in Section 3.2, we

define the quality of an individual query by the change in answer quality after adding the chunk and before. As with the answer agent, we train the query-extension agent using GRPO, with the query quality reward:

$$R_q = F1(y_{n+1}, y^*) - F1(y_n, y^*), \quad (7)$$

where y_n is the answer generated by the answer agent with chunks retrieved in the first n rounds. The training encourages the query extension agent to pinpoint missing evidence under varied contexts and to generate effective queries accordingly.

3.4 Judge Agent Optimization with Inductive Supervised Fine-tuning

To decide whether the query extension agent has retrieved sufficient evidence and when to stop retrieval and begin answering, thereby limiting hallucinations from long contexts. The judge agent estimates whether the retrieved context is sufficient for the current answer agent to answer correctly.

As with the query extension agent, we use the answer agent's response quality as the sole supervision signal for judging context quality. Unlike the query extension agent, the judge is trained via supervised fine-tuning (Joren et al., 2025). For training data generation, we also sample questions from multi-hop QA datasets and use the same Debate-RAG method with Section 3.2 to generate judge training samples. In order to get the supervised signal for the judge agent, we follow the test-time labeling method in (Zuo et al., 2025), run the answer agent on each sample, and draw 16 stochastic predicted answers. If all 16 are correct, we label the sample as 'Sufficient Information'; if all 16 are incorrect, we label it 'Insufficient Information'. This procedure yields reliable supervised signals, helping the judge learn to assess context sufficiency across various domains and context lengths. Finally, we combine these labeled samples to form the judge enhancement dataset, used to train the judge agent via supervised fine-tuning.

3.5 Inference Pipeline

With well-designed training, each agent within TRIO coordinates closely to generate concise outputs. At inference time, the three agents collaborate to acquire external knowledge through two main steps: the retrieval step and the answer step. Standard RAG methods often suffer from insufficient or irrelevant retrieval chunks in the retrieval process.

Algorithm 1: Training Algorithm of TRIO

Require: Question dataset Q, Retriever $\mathcal{R}$, Corpus C, Backbone model L

Ensure: Query-Extension $\mathcal{A}_q$, Judge $\mathcal{A}_j$, Answer $\mathcal{A}_g$

1. Extract Q_1 (hard questions) and Q_2 (normal questions) from Q
2. Conduct the retrieval debate method with $\mathcal{R}$ and C on Q_1, build training dataset T_1
3. Train L on T_1 with two-stage RL to obtain $\mathcal{A}_g$, using reward functions in Eq. 4 and Eq. 5
4. Conduct a single retrieval using question on Q_2 with $\mathcal{R}$ and C, build training dataset T_2
5. Train L on T_2 with GRPO to obtain $\mathcal{A}_q$, using the reward in Eq. 7
6. Use $\mathcal{A}_g$ to answer all questions in T_2 with 16 samples, constructing training dataset T_3
7. Supervised fine-tuning of L on T_3, obtaining the judge agent $\mathcal{A}_j$.

To address this, TRIO tightly couples the agents to ensure precise cooperation in an iterative retrieval process. Starting from an initial retrieval, the query agent uses the current context to identify knowledge gaps and generate targeted new queries, while the judge agent dynamically judges whether the accumulated information is sufficient to halt the iteration. This precise cooperation ensures the finalized query pool and retrieved chunks are highly concise. Once sufficient information is gathered, we switch to the answer step in which the answer agent generates the final answer based exclusively on the current context.

As the query and judge agents collaborate effectively to supply only the strictly necessary evidence, the answer agent is relieved from processing massively irrelevant contexts. Consequently, TRIO can generate highly concise output tokens to reach an accurate conclusion, mitigating hallucination risks and significantly improving overall inference efficiency. The detailed inference loop and agent prompts are provided in Appendix B.

4 Experiment

4.1 Experimental Settings

Baselines. We implement base RAG with different generation models (Jin et al., 2025b; Qiu

Method	In-Domain		Out-of-Domain			AVG
	HotpotQA	2Wiki	NQ	Musique	Bamboogle	
Single-Hop Retrieval						
Qwen3-4B	0.2770	0.1584	0.3357	0.0571	0.2240	0.2074
Llama3.1-8B	0.3095	0.1639	0.3720	0.0533	0.1600	0.2237
Qwen3-30B	0.2979	0.2479	0.3175	0.1278	0.2400	0.2606
Multi-Hop Retrieval						
SuRe	0.2257	0.1166	0.3532	0.0653	0.1680	0.1757
IRCOT	0.1841	0.1266	0.2171	0.0690	0.2080	0.1505
Iter-RetGen	0.3350	0.1651	0.3343	0.0724	0.1920	0.2282
DRAG	0.3060	0.2094	0.3574	0.1171	0.2880	0.2490
Self-RAG	0.1596	0.1159	0.3457	0.0422	0.0640	0.1530
Search-R1	0.4116	0.4301	0.4382	0.1836	0.4640	0.4034
R1-Searcher	0.4012	0.4046	0.3681	0.1779	0.4640	0.3779
TRIO-4B	0.4174	0.4373	0.3878	0.1547	0.4640	0.3988
TRIO-8B	0.4350	0.4710	0.4339	0.1857	0.4960	0.4294

Table 1: The overall evaluation results of TRIO and other baselines on five benchmarks. Red marks the best-performing method, and blue represents the second-best-performing method. All methods are evaluated under the same settings. Our TRIO achieves the strongest performance.

et al., 2025), which represents the basic RAG method without optimization. Moreover, we compare TRIO with several recent RAG methods, including branching method SuRE (Kim et al., 2024), loop method Self-RAG (Asai et al., 2024), IR-CoT (Trivedi et al., 2023), and Iter-RetGen (Shao et al., 2023), R1 style reasoning RAG method Search-R1 (Jin et al., 2025a), R1-Searcher (Song et al., 2025), and multi-agent debate method DRAG (Hu et al., 2025).

Datasets & Evaluation metrics. Evaluation experiments are conducted on the dev or test datasets of five benchmarks: Hotpotqa (Yang et al., 2018), 2Wiki (Ho et al., 2020), NQ (Kwiatkowski et al., 2019), Musique (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). Hotpotqa and 2Wiki are in-domain datasets because parts of their training sets are used to train our agents and other training-based baselines. NQ, Musique, and Bamboogle are out-of-domain datasets used solely for evaluation. For evaluation metrics, we report the exact match score in the results.

Implementation Details. All compared models are reproduced and evaluated using FlashRAG (Jin et al., 2025b). We employed E5-base-v2 (Wang et al., 2024) as the retriever, and the retrieval corpus using the Wikipedia dump 2018 (Karpukhin et al., 2020). For each question, the external evidence is capped at 20 retrieved chunks. In the training processes of our TRIO, the backbone models are Llama3.1-8B-Instruct for TRIO-8B and Qwen3-4B-Instruct for TRIO-4B. We select questions from the training sets of Hotpotqa and 2Wiki to generate training samples. The trained models for the three agents are available in our code. More details are provided in Appendix B.

4.2 Empirical Results and Analysis

We conduct comprehensive experiments to evaluate the performance of TRIO against various baselines across five benchmark datasets. As reported in Table 1, TRIO-8B achieves strong and consistent improvements on four benchmarks, demonstrating its robustness in settings that demand complex retrieval and multi-step inference. TRIO-4B remains competitive with R1-style reasoning-based RAG methods while using a substantially smaller backbone model. On average, TRIO-8B improves 6.45% over the second-best method. This superior performance can be attributed to TRIO's decomposition of the RAG pipeline into smaller subtasks and to performing each with a well-designed agent, which makes both retrieval and answering more effective. In particular, compared to the other multi-agent baseline DRAG, our method TRIO-8B improves 72.45%, indicating that targeted fine-tuning for each agent design substantially boosts multi-agent RAG. We also observe that TRIO-8B achieves strong results on the Musique and Bamboogle datasets without any training on these

Variants	Hotpotqa		2Wiki		Musique		Bamboogle	
	EM	F1	EM	F1	EM	F1	EM	F1
TRIO w/o Query Opt.	0.4235 (↓2.64%)	0.5555 (↓3.73%)	0.4547 (↓3.46%)	0.5191 (↓8.29%)	0.1357 (↓26.93%)	0.2456 (↓12.85%)	0.4560 (↓8.06%)	0.5590 (↓9.84%)
TRIO w/o Judge Opt.	0.4284 (↓1.52%)	0.5587 (↓3.17%)	0.4765 (↑1.17%)	0.5416 (↓4.31%)	0.1593 (↓14.22%)	0.2635 (↓6.49%)	0.4480 (↓9.68%)	0.5643 (↓8.98%)
TRIO w/o Answer Opt.	0.3735 (↓14.14%)	0.4926 (↓14.63%)	0.2778 (↓41.02%)	0.3618 (↓36.08%)	0.1051 (↓43.40%)	0.1855 (↓34.17%)	0.2800 (↓43.55%)	0.3967 (↓36.02%)
TRIO (Full)	0.4350	0.5770	0.4710	0.5660	0.1857	0.2818	0.4960	0.6200

Table 2: Ablation study results on the four multi-hop benchmarks with three variants.

benchmarks. This suggests that the agents learn effective retrieval and coordination under our diverse training scheme, yielding robust performance on new test sets. Furthermore, while TRIO-4B outperforms baseline models on in-domain benchmarks, it degrades on out-of-domain datasets. This observation seemingly highlights the impact of scaling laws in RAG tasks, indicating that limited model capacity inherently limits the system's generalization capabilities. We further evaluate TRIO's robustness to noisy irrelevant contexts under different noise levels, with detailed results and analysis provided in Appendix F.

4.3 Ablation Study

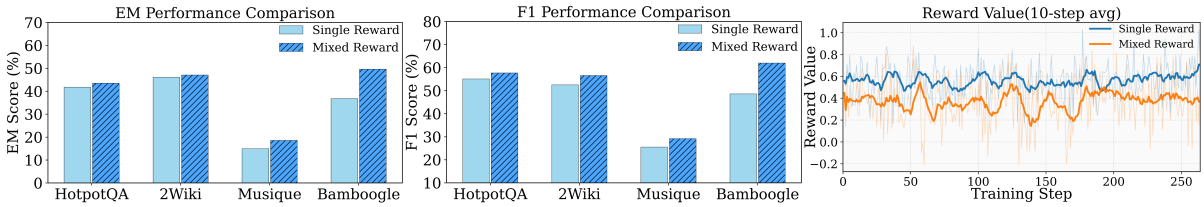
In order to evaluate the impact of each specific agent optimization in TRIO, we conduct ablation studies on the multi-hop benchmarks. Specifically, we evaluate three variants: (1) TRIO w/o Query Opt.: replace the trained query extension agent with the base backbone. (2) TRIO w/o Judge Opt.: replace the trained judge with the base backbone. (3) TRIO w/o Answer Opt.: replace the trained answer agent with the base backbone. The experimental results are reported in Table 2. On average, all variants perform well below TRIO, with a larger drop on out-of-domain datasets than on in-domain datasets. This indicates that each specialized agent is effective and that collaboration among the role-specific agents further improves performance, especially on unseen data. Specifically, TRIO w/o Answer Opt. performs worst on all benchmarks, indicating that a well-trained answer agent is critical for multi-agent RAG systems. TRIO w/o Judge Opt. is the strongest among the variants and even shows a slight EM gain over TRIO on 2Wiki. But TRIO w/o Judge Opt. still presents worse performance on out-of-domain datasets than in-domain datasets, suggesting that the supervised training method for the judge agent potentially leads to overfitting. Moreover, although all variants underperform TRIO, they still exceed the basic RAG method and even some recent RAG methods, reinforcing that specialized multi-agent designs strengthen RAG. These results suggest that targeted training enables effective coordination, thereby improving factual accuracy.

Method	Musique		Bamboogle	
	EM	F1	EM	F1
Fixed Round = 1	0.1556	0.2597	0.4480	0.5614
Fixed Round = 2	0.1585	0.2609	0.4480	0.5587
Fixed Round = 3	0.1585	0.2599	0.4560	0.5811
Fixed Round = 4	0.1547	0.2561	0.4560	0.5816
Fixed Round = 5	0.1531	0.2516	0.4560	0.5818
Flexible Round	0.1857	0.2818	0.4960	0.6200

Table 3: TRIO performance with flexible and different fixed iteration rounds on the retrieval step.

4.4 Analysis of Retrieval Rounds

To systematically assess the impact of different retrieval rounds on the performance of TRIO. We fix the number of rounds in the retrieval step from one to five and evaluate them on the two out-of-domain multi-hop datasets. The results are shown in Table 3, where the flexible round is TRIO with a maximum round equal to three, and the end decision is made by the judge agent. Fixing the round count reduces performance across all variants. On the Musique dataset, as the number of retrieval rounds increases, the F1 score initially increases slightly and then declines. On the Bamboogle dataset, the F1 score shows a large jump when the round is equal to three and only minor gains thereafter. These trends suggest that longer contexts can help, but also introduce irrelevant noise, increasing the risk of hallucination. In addition, the flexible setting achieves the best results, indicating that supplying an appropriate number of retrieval chunks is beneficial and that the judge agent effectively decides when to stop and answer.



(a) EM (b) F1 (c) Reward

Figure 3: (a) and (b): Visualization of EM and F1 metrics for TRIO with single or mixed reward training. (c): The reward values trajectory is shown. Mixed reward design surpasses single reward design in overall performance.

4.5 Reward Design Analysis

The existing reasoning RAG methods (Song et al., 2025; Jin et al., 2025a) often rely on a single thought throughout the training process. But the single reward struggles with weak generalization across RAG tasks. To investigate the impact of two-stage reward design on TRIO, we train the answer agent only with a single reward R_{a2} (see Equation 5) and the proposed two-stage reward design (described in Section 3.2, both using the same Llama3.1-8B-Instruct as backbone model). The results and reward values in the last 264 steps are shown in Figure 3. We find that single reward design yields a small drop (-5.9% on average) in two in-domain datasets and a much larger drop in two out-of-domain datasets (-17.03% on average), indicating that the two-stage reward improves overall performance, especially for generalization. Furthermore, comparing reward values over the last 264 training steps, where the reward functions are the same for both designs, further supports this finding. The single reward method achieves higher training rewards than the mixed reward design. This pattern also suggests that the training model with a single reward is prone to overfitting the training distribution. We conclude that using the two-stage reward design to train the answer agent reduces overfitting and improves generalization to unseen data.

4.6 Efficiency Analysis

To investigate the real-world inference efficiency of our method, we deployed TRIO and all baselines on two L40S GPUs with identical VRAM allocation. We sampled 100 questions from each multi-hop benchmark and measured the inference time across all methods. Table 4 presents the overall inference time of the eight retrieval-augmented baselines. While early loop-based methods, Self-RAG and Iter-RetGen, require less inference time, their performance is significantly weaker than recent reasoning and multi-agent methods, illustrating the typical trade-off between accuracy and speed.

	Inference Time (s / 100 queries)				
Method	**Hotpotqa**	**2Wiki**	**Musique**	**Bamboogle**	**Avg**
SuRe	603.2	711.8	521.7	687.4	631.0
IRCOT	64.8	90.1	84.5	82.7	80.5
Iter-RetGen	20.4	23.2	20.5	20.4	21.1
Self-RAG	10.9	10.2	9.6	10.1	10.2
Search-R1	91.5	86.4	97.4	85.1	90.1
R1-Searcher	107.4	104.9	65.6	74.2	88.0
DRAG	571.3	699.4	852.4	660.1	695.8
TRIO-8B	80.4	82.1	74.9	82.3	79.9
TRIO-4B	43.6	61.1	40.8	36.9	45.6

Table 4: Inference time of eight methods (seconds per 100 queries) across different datasets.

In contrast, TRIO-4B achieves 98% of the performance of reasoning-based methods while requiring only half the inference time. Furthermore, TRIO-8B outperforms these reasoning baselines while using slightly less inference time. As further analyzed in Appendix D, TRIO's efficiency stems from generating a highly compact number of output tokens during inference. This minimal token generation makes it highly suitable for large-scale real-world deployment while maintaining interpretability.

5 Conclusion

We present TRIO, a framework that enhances RAG by coordinating three individually optimized agents for query extension, context sufficiency judgment, and answer generation. Starting from question-answer supervision only, we design distinct data generation methods for each subtask and optimize agents using reinforcement learning or supervised fine-tuning with their specific corpora. Experiments demonstrate that TRIO-8B achieves the best average score and the best result on four of the five main benchmarks. Our analysis reveals that subtask-specific training corpora and mixed reward functions enhance each agent's specialization, while effective coordination improves answer quality. TRIO can be readily extended to various scenarios and provides a new direction for incorporating multi-agent cooperation into RAG systems.

Limitations

TRIO demonstrates strong effectiveness in improving retrieval efficiency and answer quality in different benchmarks. However, TRIO is evaluated primarily on short-answer QA and has not yet been validated for live web search or long-form generation. Its adaptive loop may fail when on tasks requiring very long-form responses or extended retrieval chains, as errors and irrelevant evidence may accumulate across retrieval rounds and hinder effective answer generation (Ren et al., 2026a). Live web search may introduce additional challenges, including changing document availability and heterogeneous webpage structures. Future work should calibrate retrieval-stopping decisions and evaluate these failure modes in dynamic, real-world corpora.

Acknowledgement

This research was supported in part by the Australian Research Council (ARC) under discovery grant DP250103612 and DP260101395, ARC Research Hub for Human-Robot Teaming for Sustainable and Resilient Construction (ITRH) grant IH24010016, and Australian National Health and Medical Research Council (NHMRC) Ideas Grant APP2021183.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2024. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, arXiv:2303.08774.

Saaket Agashe, Yue Fan, Anthony Reyna, and Xin Eric Wang. 2025. LLM-coordination: Evaluating and analyzing multi-agent coordination abilities in large language models. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 8038–8057, Albuquerque, New Mexico. Association for Computational Linguistics.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-RAG: Learning to retrieve, generate, and critique through self-reflection. In *The Twelfth International Conference on Learning Representations*.

Xiaohe Bo, Zeyu Zhang, Quanyu Dai, Xueyang Feng, Lei Wang, Rui Li, Xu Chen, and Ji-Rong Wen. 2024. Reflective multi-agent collaboration based on large language models. In *Advances in Neural Information Processing Systems*, volume 37, pages 138595–138631. Curran Associates, Inc.

Chi-Min Chan, Chunpu Xu, Ruibin Yuan, Hongyin Luo, Wei Xue, Yike Guo, and Jie Fu. 2024. RQ-RAG: Learning to refine queries for retrieval augmented generation. In *First Conference on Language Modeling*.

Haotian Chen, Qingqing Long, Meng Xiao, Xiao Luo, Wei Ju, Chengrui Wang, Xuezhi Wang, Yuanchun Zhou, and Hengshu Zhu. 2025. Scirerankbench: Benchmarking rerankers towards scientific retrieval-augmented generated llms. *arXiv preprint arXiv:2508.08742*.

Sully F Chen, Anton Alyakin, Andreas Seas, Eunice Yang, Joanne J Choi, Jin Vivian Lee, Amelia L Chen, Pranav I Warman, Rochelle T Bitolas, Robert J Steele, and 1 others. 2026. Llm-assisted systematic review of large language models in clinical medicine. *Nature medicine*, page 1152–1159.

Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, Yujia Qin, Xin Cong, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. 2024. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *The Twelfth International Conference on Learning Representations*.

Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. 2024. 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.

Shangheng Du, Jiabao Zhao, Jinxin Shi, Zhentao Xie, Xin Jiang, Yanhong Bai, and Liang He. 2026. A survey on the optimization of large language model-based agents. *ACM Comput. Surv.*, 58(9).

Yilun Du, Shuang Li, Antonio Torralba, Joshua B. Tenenbaum, and Igor Mordatch. 2024. Improving factuality and reasoning in language models through multiagent debate. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

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

Yisong Fu, Zezhi Shao, Chengqing Yu, Yujie Li, Yongjun Xu, Xueqi Cheng, and Fei Wang. 2026. Zeus: Towards tuning-free foundation model for time series analysis. In *Forty-third International Conference on Machine Learning*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang,

and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

Yiyang Gu, Jiayue Fan, Kaili Liu, Bohan Wu, Binqi Chen, Zequn Liu, Zhiping Xiao, Rong-Cheng Tu, Xiao Luo, and Ming Zhang. 2026. Evirag: Evidence-guided retrieval-augmented generation for medical vision-language models. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3753–3758.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, and 175 others. 2025. DeepSeek-R1 incentivizes reasoning in LLMs through reinforcement learning. *Nature*, 645(8081):633–638.

Bolei He, Nuo Chen, Xinran He, Lingyong Yan, Zhenkai Wei, Jinchang Luo, and Zhen-Hua Ling. 2024. Retrieving, rethinking and revising: The chain-of-verification can improve retrieval augmented generation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, Miami, Florida, USA. Association for Computational Linguistics.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Cheng-Yu Hsieh, Yung-Sung Chuang, Chun-Liang Li, Zifeng Wang, Long Le, Abhishek Kumar, James Glass, Alexander Ratner, Chen-Yu Lee, Ranjay Krishna, and Tomas Pfister. 2024. Found in the middle: Calibrating positional attention bias improves long context utilization. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14982–14995, Bangkok, Thailand. Association for Computational Linguistics.

Wentao Hu, Wengyu Zhang, Yiyang Jiang, Chen Jason Zhang, Xiaoyong Wei, and Li Qing. 2025. Removal of hallucination on hallucination: Debate-augmented RAG. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vienna, Austria. Association for Computational Linguistics.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

Zhengbao Jiang, Frank F Xu, Luyu Gao, Zhiqing Sun, Qian Liu, Jane Dwivedi-Yu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 7969–7992.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025a. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*.

Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. 2025b. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In *Companion Proceedings of the ACM on Web Conference 2025*, page 737–740, New York, NY, USA. Association for Computing Machinery.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William Cohen, and Xinghua Lu. 2019. PubMedQA: A dataset for biomedical research question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2567–2577, Hong Kong, China. Association for Computational Linguistics.

Hailey Joren, Jianyi Zhang, Chun-Sung Ferng, Da-Cheng Juan, Ankur Taly, and Cyrus Rashtchian. 2025. Sufficient context: A new lens on retrieval augmented generation systems. In *The Thirteenth International Conference on Learning Representations*.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Online. Association for Computational Linguistics.

Jaehyung Kim, Jaehyun Nam, Sangwoo Mo, Jongjin Park, Sang-Woo Lee, Minjoon Seo, Jung-Woo Ha, and Jinwoo Shin. 2024. Sure: Summarizing retrievals using answer candidates for open-domain qa of llms. *CoRR*, abs/2404.13081.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Yangning Li, Weizhi Zhang, Yuyao Yang, Wei-Chieh Huang, Yaozu Wu, Junyu Luo, Yuanchen Bei, Henry Peng Zou, Xiao Luo, Yusheng Zhao, Chunkit Chan, Yankai Chen, Zhongfen Deng, Yinghui Li, Hai-Tao Zheng, Dongyuan Li, Renhe Jiang, Ming

Zhang, Yangqiu Song, and Philip S. Yu. 2025. Towards agentic rag with deep reasoning: A survey of rag-reasoning systems in llms. *Preprint*, arXiv:2507.09477.

Haowei Liu, Xi Zhang, Haiyang Xu, Yuyang Wanyan, Junyang Wang, Ming Yan, Ji Zhang, Chunfeng Yuan, Changsheng Xu, Weiming Hu, and Fei Huang. 2025. PC-agent: A hierarchical agentic framework for complex task automation on PC. In *Workshop on Reasoning and Planning for Large Language Models*.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Junyu Luo, Weizhi Zhang, Ye Yuan, Yusheng Zhao, Junwei Yang, Yiyang Gu, Bohan Wu, Binqi Chen, Ziyue Qiao, Qingqing Long, Rongcheng Tu, Xiao Luo, Wei Ju, Zhiping Xiao, Yifan Wang, Meng Xiao, Chenwu Liu, Jingyang Yuan, Shichang Zhang, and 7 others. 2025. Large language model agent: A survey on methodology, applications and challenges. *Preprint*, arXiv:2503.21460.

Abhika Mishra, Akari Asai, Vidhisha Balachandran, Yizhong Wang, Graham Neubig, Yulia Tsvetkov, and Hannaneh Hajishirzi. 2024. Fine-grained hallucination detection and editing for language models. In *First Conference on Language Modeling*.

Thang Nguyen, Peter Chin, and Yu-Wing Tai. 2025. Ma-rag: Multi-agent retrieval-augmented generation via collaborative chain-of-thought reasoning. *Preprint*, arXiv:2505.20096.

Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, KaShun Shum, Randy Zhong, Juntong Song, and Tong Zhang. 2024. RAGTruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, Thailand. Association for Computational Linguistics.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics.

Zihan Qiu, Zekun Wang, Bo Zheng, Zeyu Huang, Kaiyue Wen, Songlin Yang, Rui Men, Le Yu, Fei Huang, Suozhi Huang, Dayiheng Liu, Jingren Zhou, and Junyang Lin. 2025. Gated attention for large language models: Non-linearity, sparsity, and attention-sink-free. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 100092–100118. Curran Associates, Inc.

Yingtao Ren, Xiao Luo, Yu-Cheng Chang, and Chin-teng Lin. 2026a. Context-attended adversarial reinforcement learning for robust multi-step retrieval augmented generation. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 17305–17319, San Diego, California, United States. Association for Computational Linguistics.

Yingtao Ren, Ziyi Zhao, Yiwei Fu, Xiao Luo, Yu-Cheng Chang, and Chin-Teng Lin. 2026b. When context bites: Detecting rag poisoning via document-level attention collapse. In *Proceedings of the 49th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '26, page 4080–4086, New York, NY, USA. Association for Computing Machinery.

Zhihong Shao, Yeyun Gong, yelong shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Richard James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2024. REPLUG: Retrieval-augmented black-box language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, Mexico City, Mexico. Association for Computational Linguistics.

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

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

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.

Fei Wang, Xingchen Wan, Ruoxi Sun, Jiefeng Chen, and Sercan O Arik. 2025a. Astute RAG: Overcoming Imperfect Retrieval Augmentation and Knowledge Conflicts for Large Language Models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30553–30571, Vienna, Austria. Association for Computational Linguistics.

Junlin Wang, Jue WANG, Ben Athiwaratkun, Ce Zhang, and James Zou. 2025b. Mixture-of-agents enhances large language model capabilities. In *The Thirteenth International Conference on Learning Representations*.

Juyuan Wang, Rongchen Zhao, Wei Wei, Yufeng Wang, Mo Yu, Jie Zhou, Jin Xu, and Liyan Xu. 2026. Comorag: A cognitive-inspired memory-organized rag for stateful long narrative reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 33557–33565.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2024. Text embeddings by weakly-supervised contrastive pre-training. *Preprint*, arXiv:2212.03533.

Hui Wei, Zihao Zhang, Shenghua He, Tian Xia, Shijia Pan, and Fei Liu. 2025. PlanGenLLMs: A modern survey of LLM planning capabilities. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 19497–19521, Vienna, Austria. Association for Computational Linguistics.

Yijia Xiao, Wanjia Zhao, Junkai Zhang, Yiqiao Jin, Han Zhang, Zhicheng Ren, Renliang Sun, Haixin Wang, Guancheng Wan, Pan Lu, Xiao Luo, Yu Zhang, James Zou, Yizhou Sun, and Wei Wang. 2025. Protein large language models: A comprehensive survey. *Preprint*, arXiv:2502.17504.

Zhichao Xu, Minheng Wang, Yawei Wang, Wenqian Ye, Yuntao Du, Yunpu Ma, and Yijun Tian. 2026. Recon: Reasoning with condensation for efficient retrieval-augmented generation. *Preprint*, arXiv:2510.10448.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context. In *The Twelfth International Conference on Learning Representations*.

Tian Yu, Shaolei Zhang, and Yang Feng. 2024a. Autorag: Autonomous retrieval-augmented generation for large language models. *Preprint*, arXiv:2411.19443.

Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, and Bryan Catanzaro. 2024b. RankRAG: Unifying context ranking with retrieval-augmented generation in LLMs. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Zhiyin Yu, Chao Zheng, Chong Chen, Xian-Sheng Hua, and Xiao Luo. 2025. scRAG: Hybrid retrieval-augmented generation for LLM-based cross-tissue single-cell annotation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 954–970, Vienna, Austria. Association for Computational Linguistics.

Shengbin Yue, Siyuan Wang, Wei Chen, Xuanjing Huang, and Zhongyu Wei. 2024. Synergistic multi-agent framework with trajectory learning for knowledge-intensive tasks. *CoRR*, abs/2407.09893.

Hangfan Zhang, Zhiyao Cui, Qiaosheng Zhang, and Shuyue Hu. 2025a. Multi-LLM-agents debate - performance, efficiency, and scaling challenges. In *The Fourth Blogpost Track at ICLR 2025*.

Yufeng Zhang, Xuepeng Wang, Lingxiang Wu, and Jinqiao Wang. 2025b. Enhancing chain of thought prompting in large language models via reasoning patterns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25985–25993.

Yuxin Zuo, Kaiyan Zhang, Li Sheng, Shang Qu, Ganqu Cui, Xuekai Zhu, Haozhan Li, Yuchen Zhang, Xinwei Long, Ermo Hua, Biqing Qi, Youbang Sun, Zhiyuan Ma, Lifan Yuan, Ning Ding, and Bowen Zhou. 2025. TTRL: Test-time reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

A AI Assistants Usage

We use large language models (ChatGPT) to refine grammar and enhance sentence clarity in our manuscript, and to aid in debugging code by explaining errors. This disclosure follows the ACL 2026 policy on AI usage.

B Prompts and Details of TRIO

B.1 Inference loop for TRIO

The multi-agent workflow begins by putting the raw question to the retriever to obtain the initial chunks from the external corpus:

$$Q_0, C_0 = \mathcal{R}(x, k), \tag{8}$$

where Q_0 and C_0 represent the initial query pool and its corresponding set of retrieved chunks for question x. The retriever $\mathcal{R}$ maps a query pool Q to the union of the top-k chunks returned from the corpus for each query.

In the iterative retrieval steps, each round begins with the *query extension agent* $\mathcal{A}_q$ using the previous query pool and retrieval chunks (Q_n, C_n) to identify the knowledge gaps in answering the question x and generate a new query. And then send the query to the retriever, to a new query pool Q_{n+1} and corresponding retrieved chunks:

$$Q_{n+1}, C_{n+1} = Q_n + \mathcal{R}(\mathcal{A}_q(x, Q_n, C_n)). \tag{9}$$

At the end of each retrieval round, the *judge agent* $\mathcal{A}_j$ judges whether the current query pool is sufficient to answer the question x:

$$D = \mathcal{A}_j(x, Q_{n+1}, C_{n+1}). \tag{10}$$

If the decision D is that the current information is insufficient to answer the question, we iterate to the next round(up to a fixed round budget). Otherwise, we switch to the answer step in which the *answer agent* generates the final answer based on the optimized query pool and their corresponding retrieved chunks:

$$y = \mathcal{A}_g(x, Q^*, C^*), \tag{11}$$

where $\mathcal{A}_g$ denotes the answer agent, and y is the final answer to the question x.

B.2 Prompts of Inference

The prompt below is for the query extension agent to generate exactly one new retrieval query in a multi-step retrieval process. Given the question, the existing query pool, and retrieved documents, the agent identifies missing information and proposes a keyword-based query. The output is strictly limited to at most eight keywords, with no sentences or stopwords, and must differ from all existing queries to encourage diversity and reduce redundant retrieval.

Prompt for Query Extension Agent

System Prompt:

You are a retrieval query generation agent.

Input:

QUESTION: ...

EXISTING QUERIES and RETRIEVED DOCUMENTS: ...

Task:

Generate exactly one NEW QUERY that:

1) Directly helps answer the QUESTION or fill gaps not covered by EXISTING QUERIES and RETRIEVED DOCUMENTS;

2) Is different from all EXISTING QUERIES.

Output rules:

Only keywords; no sentences, stopwords, or connectors (e.g., and, or, of, the, about).

Separate keywords with a single space;

Maximum 8 keywords.

Output: Only the NEW QUERY text. No explanations or extra text.

User Prompt:

Question: |*Input Question*|

Query Pool: |*Current Query Pool*|

The prompt below instructs the answer agent to produce a final answer using the retrieval context associated with the current query pool. The agent is required to output only the final answer, without explanations or additional text, ensuring a clean, comparable output format. This design supports consistent evaluation of answer correctness across different inference settings.

Prompt for Answer Agent

System Prompt:

Answer the question based on the given document. Output only the final answer with no explanations or additional text.

Query Pool:

|*Current Query Pool*|

User Prompt:

Question: |*Input Question*|

The prompt below guides the judge agent in deciding whether the existing queries and retrieved documents provide sufficient information to answer the question correctly. The agent outputs only a binary decision: "Sufficient Information" or "Insufficient Information", with no further explanation. This minimal output enables the system to control whether additional query expansion and retrieval rounds are needed.

Prompt for Judge Agent

System Prompt:

Determine whether the EXISTING QUERIES and RETRIEVED DOCUMENTS provide sufficient information to answer the QUESTION correctly. Output only one of the following: 'Sufficient Information' or 'Insufficient Information'. Do not include explanations or additional text.

User Prompt:

Question: | *Input Question* |

Query Pool: | *Current Query Pool* |

B.3 Setting Details for TRIO Training

All the 8148 training questions come from two multi-hop QA benchmarks: HotpotQA (Yang et al., 2018) and 2WikiMultiHopQA (Ho et al., 2020). The answer agent and query extension agent are trained using GRPO reinforcement learning. For the judge agent, we constructed 49785 candidate samples for SFT. Among them, 5312 samples produced mixed answers and were discarded, representing approximately 10.7%. The remaining 44473 samples were used for training, including 9087 sufficient samples and 35386 insufficient samples. Notably, we do not need any additional human annotations, and all the supervised signals were generated by the TRIO training loop (see Section 3). Each sample is rolled out 16 times under GRPO. We included more details and the trained models in the code we provided. Train batch size is 512, rollout batch size is 32, with the learning rate set to 10^{-6} and the KL divergence set to 10^{-4}. And the judge agent is trained using supervised fine-tuning. Training tasks are conducted on a single node with two H100 GPUs. For the query-extension agent, reward computation is offloaded to two additional L40 GPUs. All the evaluation experiments are conducted on two L40S GPUs.

B.4 Prompts in Training

The prompt below is used to train the Judge Agent to output a binary sufficiency label indicating whether the current queries and retrieved documents contain sufficient evidence to answer the question correctly. It enforces a strict output format ("Sufficient Information"/"Insufficient Information") to enable consistent supervision.

Prompt for Judge Agent in Training

System Prompt:

Determine whether the EXISTING QUERIES and RETRIEVED DOCUMENTS provide sufficient information to answer the QUESTION correctly. Output only one of the following: 'Sufficient Information' or 'Insufficient Information'. Do not include explanations or additional text.

User Prompt:

Question: Are the directors of both films I Can Do Bad All By Myself (Film) and Shit Year from the same country?

EXISTING QUERIES and RETRIEVED DOCUMENTS

Query 1: Are the directors of both films I Can Do Bad All By Myself (Film) and Shit Year from the same country?

Retrieved Content:

Doc 1: ...

Doc 2: ...

Doc 3: ...

Doc 4: ...

Doc 5: ...

The prompt below is used to train the Query Extension Agent to produce one additional keyword query based on the question and the current retrieval results. The output is constrained to a short, non-overlapping query to encourage informative and diverse retrieval during training.

Prompt for Query Extension Agent in Training

System Prompt:

You are a retrieval query generation agent.

Input:

QUESTION: ...

EXISTING QUERIES and RETRIEVED DOCUMENTS: ...

Task:

Generate exactly one NEW QUERY that:

1) Directly helps answer the QUESTION or fill gaps not covered by EXISTING QUERIES and RETRIEVED DOCUMENTS;

2) Is different from all EXISTING QUERIES.

Output rules:

Only keywords; no sentences, stopwords, or connectors (e.g., and, or, of, the, about).

Separate keywords with a single space;

Maximum 8 keywords.

Output: Only the NEW QUERY text. No explanations or extra text.

User Prompt:

Question: Are the directors of both films I Can Do Bad All By Myself (Film) and Shit Year from the same country?

EXISTING QUERIES and RETRIEVED DOCUMENTS

Query 1: Are the directors of both films I Can Do Bad All By Myself (Film) and Shit Year from the same country?

Retrieved Content:

Doc 1: ...

Doc 2: ...

Doc 3: ...

Doc 4: ...

Doc 5: ...

The prompt below is used to train the Answer Agent to produce the final answer from the provided multi-query retrieval context. It presents the question along with the existing queries and their retrieved documents, and enforces an answer-only output without explanations to maintain consistency in supervision and evaluation.

Prompt for Answer Agent in Training

System Prompt:

Answer the question based on the given document. Output only the final answer with no explanations or additional text.

EXISTING QUERIES and RETRIEVED DOCUMENTS

Query 1: Are the directors of both films I Can Do Bad All By Myself (Film) and Shit Year from the same country?

Retrieved Content:

Doc 1(Title: "I Can Do Bad All by Myself (film) "): ...

Doc 2(Title: "I Can Do Bad All by Myself (film) ") : ...

Doc 3(Title: "Shit Year") : ...

Doc 4(Title: "Shit Year"): ...

Doc 5(Title: Themselves): ...

Query 2: Tyler Perry

Retrieved Content:

Doc 1(Title: "Tyler Perry"): ...

Doc 2(Title: "Tyler Perry Studios"): ...

Doc 3(Title: "Tyler Perry"): ...

Doc 4(Title: "Tyler Perry Studios"): ...

Doc 5(Title: "Tyler Perry"): ...

User Prompt:

Question: Are the directors of both films I Can Do Bad All By Myself (Film) and Shit Year from the same country?

B.5 Comparison with Existing Adaptive RAG Methods

Existing adaptive RAG methods differ in where adaptation is introduced. RQ-RAG (Chan et al., 2024) trains a query-refinement model. IR-CoT (Trivedi et al., 2023) interleaves prompting-based reasoning and retrieval. Self-RAG (Asai et al., 2024) trains a single generator to emit reflection tokens. Search-R1 (Jin et al., 2025a) and R1-Searcher (Song et al., 2025) optimize a single retrieval-reasoning policy. DRAG (Hu et al., 2025) and MA-RAG (Nguyen et al., 2025) coordinate role-specialized agents mainly through debate or collaborative prompting without task-specific training. In contrast, TRIO sequentially optimizes separate query-extension, stopping, and answer agents, with answer agent outcomes providing supervision for query extension and sufficiency judgment.

C Prompts of Supervised Data Generation

The prompts below implement a lightweight retrieval debate mechanism to assess the sufficiency of evidence. The Proponent argues that retrieved documents support an answer and provide a conclusion without requesting further retrieval, while the Opponent challenges sufficiency and either proposes a new query or concedes if evidence is complete. The Moderator then selects the more convincing argument and outputs only the winning agent name to drive the pipeline.

Prompt for Moderator in Retrieval Debate

System Prompt:

You are the judge in a debate. Your task is to evaluate the arguments from agents. There are two types of agents:

1. Proponent Agent: Argues that the current retrieved content is sufficient.

2. Opponent Agent: Argues that the current retrieved content is insufficient and proposes query refinement.

User Prompt:

Question: |*Input Question*|

Query Pool: |*Current Query Pool*|

Proponent:

|*Output of Proponent*|

Opponent:

|*Output of Opponent*|

Output only the agent's name. Do NOT output more than one agent or any explanation.

Prompt for Proponent in Retrieval Debate

System Prompt:

You are a debater. Argue that the current retrieved content is sufficient to answer the question and try to give the answer based on the given documents. Deliver a brief, strong argument with clear reasoning. Do not suggest further retrieval. No extra explanations.

User Prompt:

Question: |*Input Question*|

Query Pool: |*Current Query Pool*|

Prompt for Opponent in Retrieval Debate

System Prompt:

You are a critical thinker and debater. Your task is to challenge the sufficiency of the current documents. However, your ultimate goal is to find the correct answer efficiently. If you believe the provided information is TRULY and COMPLETELY sufficient, your duty is to concede.

The action you can choose:

1. Query Expansion: If a completely new line of inquiry is needed. Output exactly in this format in the end of your response: Query Expansion: [New Query]

2. Concede: If the current information is truly sufficient. Only Output: Concede

Deliver a brief, strong argument, then you must choose one action in the exact format required.

User Prompt:

Question: |*Input Question*|

Query Pool: |*Current Query Pool*|

D Case Study

We provide two typical cases to show how our agents collaborate to effectively answer multi-hop questions. Table 5 illustrates how TRIO dynamically adapts to context quality. Case 1 shows an efficient "early stop" in which the Judge Agent correctly recognizes a perfect initial retrieval, avoiding additional retrieval costs. In contrast, Case 2 highlights the "Correction Pattern" for complex queries. When the initial retrieval returns related entities (e.g., "FC Barcelona") but lacks the specific league rules, the Judge Agent issues an "Insufficient" verdict. This triggers the Query Extension Agent to pivot the search towards "La Liga". This multi-round collaboration ensures that the Answer Agent generates a response once the semantic gap is bridged, effectively reducing hallucinations caused by irrelevant context. Furthermore, the entire inference process generates only 5 to 7 words. Even for complex questions that require up to three retrieval rounds, the output length remains significantly shorter than that of state-of-the-art reasoning methods, which typically generate hundreds of words per question.

Case 1: Immediate Success (Early Stop)
Question: Who played the character in the Santa Clause 3 that has a series named after it that includes Frost at Christmas?
Ground Truth: Martin Short
Round 0:
Retrieval Query: Who played the character in the Santa Clause 3 that has a series named after it that includes Frost at Christmas?
Retrieved Content: 1. *The Santa Clause 3*; 2. *The Santa Clause (film series)*; 3. *The Santa Clause 3: The Escape Clause*...
Judge: Sufficient *(Decision: Stop & Answer)*
Answer: Jack Frost played by Martin Short.
Case 2: Iterative Refinement (Correction)
Question: How many games do team in the league where Barcelona won titles in 1948 and 1949 play?
Ground Truth: 38
Round 0:
Retrieval Query: How many games do team in the league where Barcelona won titles in 1948 and 1949 play?
Retrieved Content: 1. *UER Pineda de Mar*; 2. *1949 Campeonato Profesional*; 3. *History of FC Barcelona*...
Judge: Insufficient *(Reason: Lacking league rules.)*
Round 1:
Query Extension: "La Liga"
Retrieval Query: La Liga
Retrieved Content: 1. *La Liga*; 2. *Liga de Futbol Profesional*...
Judge: Sufficient *(Decision: Gap filled.)*
Answer: 38

Table 5: Case studies demonstrating the token efficiency of TRIO. Driven by the coordination among the three agents, the system completes tasks with highly concise token outputs.

E Resource Analysis

To further investigate the training resource requirements of TRIO, we report the computational cost and memory usage of TRIO-8B and TRIO-4B in Table 6. As a reference point, a previous work (Xu et al., 2026) reports that its reproduction with Qwen2.5-3B requires 14.7 hours on 4 H200 GPUs, corresponding to 58.8 H200 GPU-hours. In comparison, our TRIO-4B version requires 58.1 H100 GPU-hours, which is slightly lower in absolute GPU-hour cost despite training multiple specialized agents and using larger models. In addition, the peak training memory usage is approximately 94 GB per GPU, or 188 GB in aggregate across two GPUs. At deployment time, current TRIO uses three independently optimized model checkpoints. The total deployment memory footprint is approximately 40 GB for TRIO-4B and 60 GB for TRIO-8B. For training, the answer agent is optimized with GRPO on 18380 samples constructed from 8148 questions, the query extension agent is trained with GRPO on 10000 samples, and the judge agent is trained with supervised fine-tuning on 44473 samples.

Agent	**Stage**	**GPU-hours (H100)**
TRIO-8B (Llama-8B)		
Answer agent	Stage 1	33.38h
Answer agent	Stage 2	13.06h
Query extension agent	–	33.96h
Judge agent	–	12.60h
Total		93.00h
TRIO-4B (Qwen-4B)		
Answer agent	Stage 1	21.80h
Answer agent	Stage 2	8.87h
Query extension agent	–	20.60h
Judge agent	–	6.83h
Total		58.10h

Table 6: Training resource consumption of TRIO-8B and TRIO-4B.

F Additional Results and Robustness Analysis

To further investigate TRIO's robustness against distracting retrieval contexts, we conduct an additional noisy retrieval experiment. Specifically, we introduce two noise levels by replacing chunks of the retrieved context with rank 998-1000 noisy chunks from the corpus. In the low-noise setting, one chunk among the top-5 retrieved chunks is replaced, while in the high-noise setting, three are replaced. The results are shown in Table 7.

TRIO demonstrates greater robustness to retrieval noise than existing methods. For example, under the high-noise setting, TRIO-8B maintains EM scores of 0.1348 and 0.4000 on MuSiQue

Method	MuSiQue		Bamboogle	
	Low	High	Low	High
SuRe	0.0645	0.0508	0.1280	0.1200
IRCoT	0.0662	0.0559	0.1520	0.1920
Iter-RetGen	0.0699	0.0488	0.1760	0.1600
DRAG	0.1129	0.1067	0.2720	0.2640
Self-RAG	0.0384	0.0372	0.0640	0.0560
Search-R1	0.1531	0.0144	0.1920	0.1600
TRIO-4B	0.1407	0.1283	0.3440	0.3120
TRIO-8B	**0.1708**	**0.1348**	**0.4560**	**0.4000**

Table 7: EM scores under low and high noise settings. Low and High denote replacing one and three of the top-5 retrieved chunks with noisy chunks, respectively.

Method	HotpotQA	2Wiki	MuSiQue	Bamboogle
SuRe	0.3538	0.1917	0.1296	0.2727
IRCoT	0.2825	0.2363	0.1244	0.3112
Iter-RetGen	0.4419	0.2636	0.1297	0.2701
DRAG	0.4146	0.2791	0.2010	0.4030
Self-RAG	0.2842	0.2351	0.1195	0.1576
Search-R1	0.5306	0.4879	0.2608	0.5780
R1-Searcher	0.5200	0.4481	<u>0.2617</u>	<u>0.5782</u>
TRIO-4B	<u>0.5455</u>	<u>0.5023</u>	0.2555	0.5555
TRIO-8B	**0.5770**	**0.5660**	**0.2818**	**0.6200**

Table 8: F1 results on four multi-hop benchmarks. The best result in each column is shown in bold, and the second-best result is underlined.

and Bamboogle, respectively, while Search-R1 decreases substantially to 0.0144 and 0.1600. Interestingly, we observe that reasoning-based Search-R1 is more sensitive to noisy retrieval contexts, indicating that stronger reasoning ability alone does not necessarily guarantee robustness when the retrieved context contains highly irrelevant information. In contrast, TRIO benefits from the collaboration among the query extension, sufficiency judgment, and answer agents, enabling the system to better identify and utilize useful evidence while reducing the influence of distracting contexts. We also report the F1 scores of compared methods and our methods across the four benchmarks in Table 8.

G Evaluation on Biomedical Domain Dataset

To further investigate TRIO's generalization to out-of-domain questions that differ substantially from the training data, we conduct an additional evaluation on the PubMedQA benchmark (Jin et al., 2019). PubMedQA represents a completely unseen biomedical domain relative to the training datasets. We construct the evaluation set using the corresponding 270K biomedical retrieval corpus and evaluate all methods under the same retrieval and generation settings as the main experiments. The results are shown in Table 9.

Method	PubMedQA
SuRe	0.256
IRCoT	0.458
Iter-RetGen	0.716
DRAG	0.644
Self-RAG	0.628
Search-R1	0.568
TRIO-4B	0.666
TRIO-8B	**0.766**

Table 9: Results of TRIO and other baselines on PubMedQA benchmark.

TRIO-8B achieves the best performance on the unseen biomedical benchmark, and TRIO-4B achieves 0.666, suggesting that TRIO's task-specific optimization can generalize beyond Wikipedia-based multi-hop QA scenarios. In addition, we observe that the reasoning-based method Search-R1 does not transfer effectively to the unseen biomedical domains. In contrast, methods using untrained general models, such as Iter-RetGen and DRAG, exhibit stronger transferability on the biomedical dataset, achieving 0.716 and 0.644 in accuracy, respectively. We also observe that the earlier method SuRe exhibits relatively limited generalization on the biomedical benchmark.