OPERA: A Reinforcement Learning–Enhanced Orchestrated Planner-Executor Architecture for Reasoning-Oriented Multi-Hop Retrieval

Yu Liu[1, 2], Yanbing Liu[1, 2], Fangfang Yuan[1], Cong Cao[1], Youbang Sun[4], Kun Peng[1, 2], WeiZhuo Chen[1, 2], Jianjun Li[3], Zhiyuan Ma[3*]

[1]Institute of Information Engineering, Chinese Academy of Sciences
[2]School of Cyber Security, University of Chinese Academy of Sciences
[3]School of Computer Science and Technology, Huazhong University of Science and Technology
[4]Department of Electronic Engineering, Tsinghua University
liuyu@iie.ac.cn, mzyth@hust.edu.cn

Abstract

Recent advances in large language models (LLMs) and dense retrievers have driven significant progress in retrieval-augmented generation (RAG). However, existing approaches face significant challenges in complex reasoning-oriented multi-hop retrieval tasks: ***1) Ineffective reasoning-oriented planning:*** Prior methods struggle to generate robust multi-step plans for complex queries, as rule-based decomposers perform poorly on out-of-template questions. ***2) Suboptimal reasoning-driven retrieval:*** Related methods employ limited query reformulation, leading to iterative retrieval loops that often fail to locate golden documents. ***3) Insufficient reasoning-guided filtering:*** Prevailing methods lack the fine-grained reasoning to effectively filter salient information from noisy results, hindering utilization of retrieved knowledge. Fundamentally, these limitations all stem from the weak coupling between retrieval and reasoning in current RAG architectures. We introduce the **O**rchestrated **P**lanner-**E**xecutor **R**easoning **A**rchitecture (**OPERA**), a novel reasoning-driven retrieval framework. OPERA's Goal Planning Module (**GPM**) decomposes questions into sub-goals, which are executed by a Reason-Execute Module (**REM**) with specialized components for precise reasoning and effective retrieval. To train OPERA, we propose Multi-Agents Progressive Group Relative Policy Optimization (**MAPGRPO**), a novel variant of GRPO. Experiments on complex multi-hop benchmarks show OPERA's superior performance, validating both the MAPGRPO method and OPERA's design.

Code — https://github.com/Ameame1/OPERA
Extended version — https://arxiv.org/abs/2508.16438

1 Introduction

The ability to solve complex problems is a core aspect of intelligence, and within Retrieval-Augmented Generation (RAG), reasoning-centric retrieval provides an effective means to address these tasks in the post-training era. The concurrent improvement of Large Language Models (LLMs) (Brown et al. 2020; Devlin et al. 2019) and dense retrieval systems (Karpukhin et al. 2020; Khattab and Zaharia 2020) has propelled the evolution of RAG. The traditional RAG follows a retrieve-then-reason paradigm (Lee, Chang, and Toutanova 2019; Lewis et al. 2020), which has now been widely optimized into a multi-stage pipeline including query rewriting, document retrieval, document filtering, and answer generation (Izacard and Grave 2021). However, despite significant progress, effectively orchestrating these capabilities remains challenging, as existing approaches struggle with the demands of multi-hop reasoning. Consider a query such as, "What was the previous occupation of the person who succeeded the founder of the company that acquired WhatsApp?" Such questions demand not merely retrieving documents, but orchestrating a precise sequence of retrieval and reasoning steps where each operation depends critically on its predecessors.

Current approaches face several key challenges. **First**, existing solutions have ***limited reasoning-oriented planning***. While planner-first models like PlanRAG (Lee, An, and Kim 2024) and REAPER (Joshi et al. 2024) introduce upfront planning mechanisms, their static plans cannot dynamically adapt to unforeseen challenges during retrieval. **Second**, most methods ***lack effective reasoning-aware retrieval***. Even adaptive methods like Adaptive-RAG (Jeong et al. 2024) and AT-RAG (Rezaei et al. 2024), which adjust retrieval strategy based on query complexity, lack the fine-grained, reasoning-driven query reformulation needed for complex multi-hop scenarios. **Third**, current systems provide ***inadequate reasoning-guided filtering***. Even when retrieval fetches golden documents, they are often buried within noisy top-K results. While iterative approaches like ReAct (Yao et al. 2023b) attempt to address this through reasoning-action loops, and recent methods such as BGE (Ke et al. 2024) show promise in bridging retriever-LLM preferences, their effectiveness remains limited by indirect reward signals that fail to capture the nuanced reasoning required for effective filtering. These limitations persist because advanced enhancement strategies—spanning SFT, preference optimization, and RL—are insufficient or misaligned, often leading to goal misalignment between modules (Qi et al. 2024; Wang et al. 2023). These issues stem from fundamental weakness in the coupling between retrieval and reasoning, preventing full utilization of modern LLMs and dense retrievers' capabilities.

To address these limitations, we introduce OPERA, a

*Corresponding author.

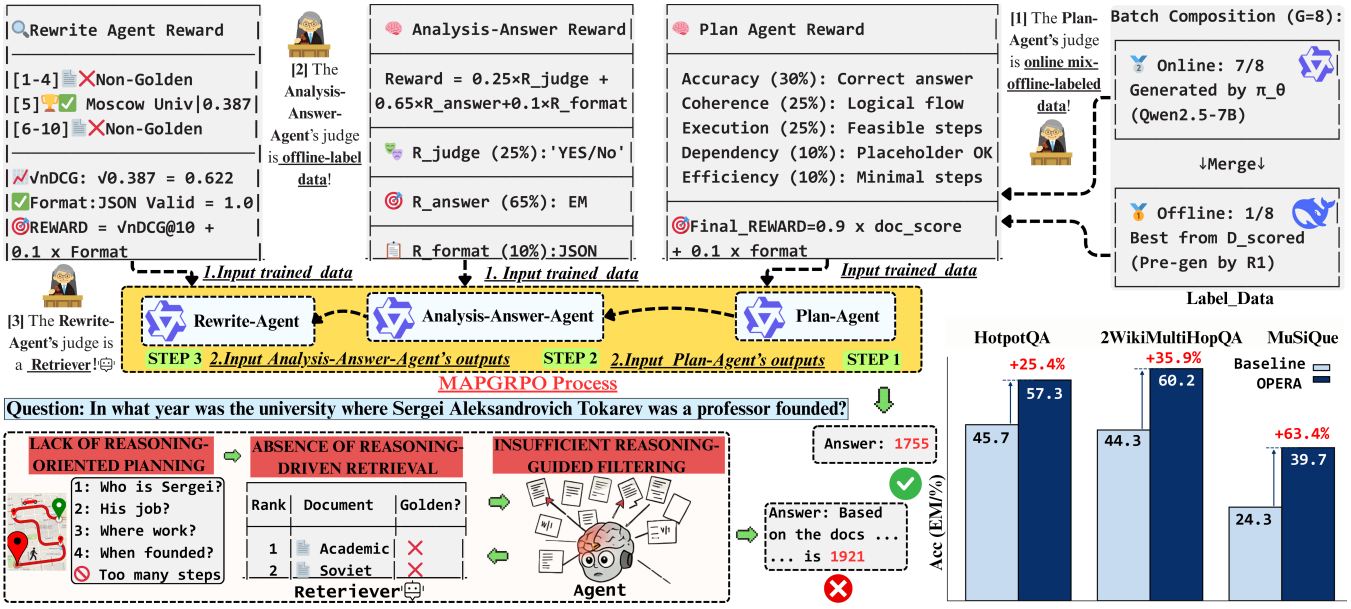

Figure 1: Overview of OPERA's MAPGRPO training framework and performance comparison with traditional RAG.

novel reasoning-driven framework. OPERA systematically decouples high-level strategic planning from low-level tactical execution through two core modules: a Goal Planning Module (GPM) and a Reason-Execute Module (REM). The GPM uses a dedicated **Plan Agent** to decompose complex queries into coherent, executable sub-goals. The REM implements a dual-agent system supported by a neural dense retriever: the **Analysis-Answer Agent** extracts precise answers from retrieved context, while a specialized **Rewrite Agent** reformulates queries to improve subsequent retrieval attempts. Furthermore, OPERA features a Trajectory Memory Component (TMC) to enhance interpretability, providing a clear rationale for each action taken by the agents. Our training protocol sequentially optimizes each of the three agents with a GRPO reward function tailored to its role—plan quality, reasoning accuracy, and retrieval effectiveness. Our key contributions are three-fold:

- **A reasoning-driven retrieval framework.** OPERA integrates reasoning into each component, improving planning, retrieval, and filtering effectiveness in RAG systems. OPERA features a TMC that enhances interpretability through action rationales.
- **A specialized training algorithm.** MAPGRPO enhances reasoning capabilities through fine-grained, role-specific credit assignment, improving individual agent skills while ensuring coordination across the planning, retrieval, and reasoning workflow.
- **Strong empirical results on multi-hop benchmarks.** Extensive experiments validate OPERA's reasoning-centric architecture and training approach are effective.

2 Related Work

RAG. To mitigate hallucination, Retrieval-Augmented Generation (RAG) was introduced to ground outputs in external knowledge (Lewis et al. 2020). The initial "retrieve-then-read" paradigm (Lee, Chang, and Toutanova 2019), however, proved insufficient for multi-hop reasoning due to its static pipeline (Trivedi et al. 2022; Arabzadeh, Yan, and Clarke 2021; Luan et al. 2021). The field has since evolved toward more dynamic retrieval, for instance by routing queries based on complexity (Jeong et al. 2024) or introducing explicit planning (Lee, An, and Kim 2024). While these methods improve upon static RAG, they primarily optimize the retrieval act itself, rather than the overarching reasoning strategy. OPERA is distinct in its hierarchical approach: a specialized Goal Planning Module (GPM) governs high-level strategy, while an agentic Reason-Execute Module (REM) handles tactical execution, including fine-grained analysis and adaptive query reformulation.

Chain-of-Thought. Chain-of-Thought (CoT) methods (Wei et al. 2022; Yao et al. 2023a) and agentic frameworks like ReAct (Yao et al. 2023b) and IRCoT (Trivedi et al. 2023) established the importance of decomposing problems and interleaving reasoning with actions. However, these frameworks rely on a single, general LLM for high-level planning and low-level execution, which can compromise reliability. While multi-agent systems like MetaGPT (Hong et al. 2024) assign distinct roles, they typically use generalist models. OPERA advances by employing an asymmetric architecture, pairing the strategic Goal Planning Module (GPM) with the agentic Reason-Execute Module (REM) to separate strategic and tactical concerns.

Reinforcement Learning. A key aspect of our work is the training of our specialized planner and executor. Many frameworks utilize on-policy algorithms like Proximal Policy Optimization (PPO) (Schulman et al. 2017); however, PPO often struggles with the large action spaces and sparse rewards common in RAG (Stiennon et al. 2020; Uc-Cetina et al. 2022; Ramamurthy et al. 2023), making training un-

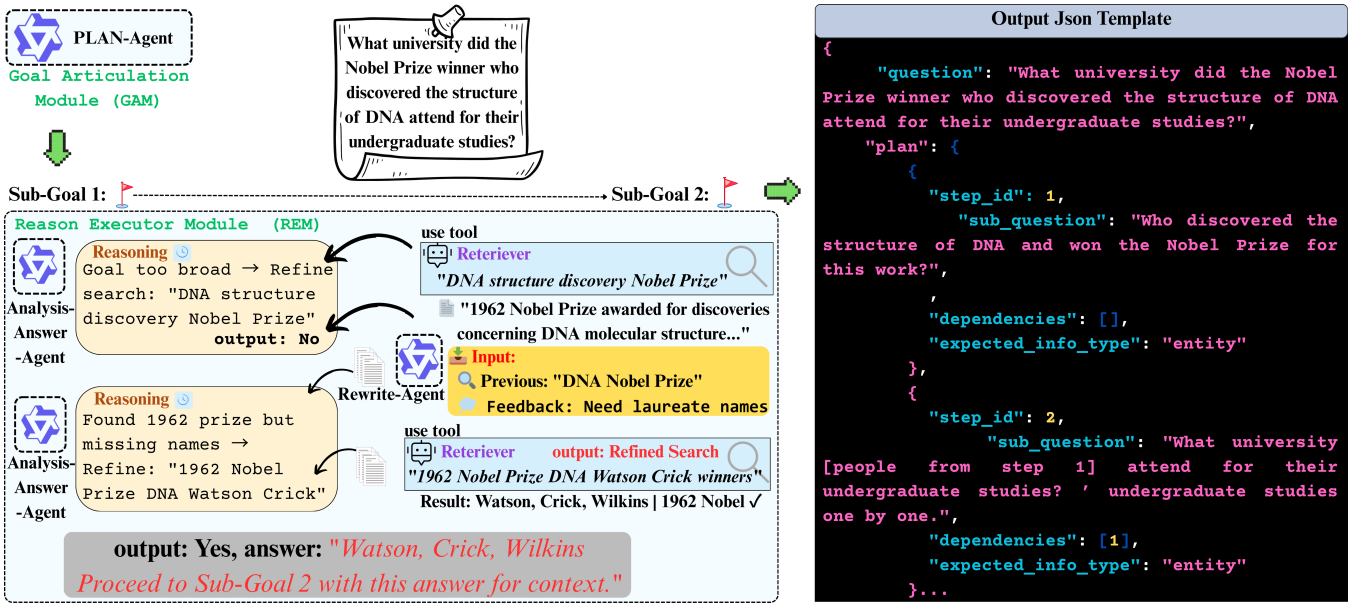

Figure 2: Overview of OPERA architecture showing the Goal Planning Module (GPM) with Plan Agent for strategic decomposition, and the Reason-Execute Module (REM) with Analysis-Answer and Rewrite Agents for adaptive execution. The Trajectory Memory Component (TMC) will record all things.

stable and sample-inefficient. To address this, preference-based optimization has gained traction. While Direct Preference Optimization (DPO) (Rafailov et al. 2023) has become a standard for learning from binary preferences (chosen $>$ rejected), it is ill-suited for more nuanced reward signals (Ivison et al. 2024). Our training methodology generates fine-grained scalar scores reflecting the quality of a plan or an execution step. Using DPO would discard this rich information by compressing it into a binary signal. To fully leverage these scalar rewards within our multi-agent setting, we build upon Group Relative Policy Optimization (GRPO) (Shao et al. 2024) and introduce our own variant: Multi-Agents Progressive Group Relative Policy Optimization (MAPGRPO). Unlike standard GRPO, MAPGRPO is specifically designed for our staged training protocol, enabling fine-grained credit assignment (Papangelis et al. 2019) and ensuring coordinated optimization across the distinct roles of the GPM and REM agents.

3 Method

Problem Formulation

We formalize reasoning-driven multi-hop retrieval as follows. Given a complex question q, the goal is to generate an accurate answer a^* through orchestrated reasoning and retrieval operations. Let $\mathcal{D}$ denote the document corpus and $\mathcal{R} : \mathcal{Q} \to \mathcal{D}^k$ be a retrieval function that maps queries from query space $\mathcal{Q}$ to top-k documents.

The task decomposes into three reasoning-driven subproblems: (1) **Reasoning-Driven Planning**: $f_{\text{plan}} : q \to \{p_1, ..., p_m\}$ where m represents the number of sub-goals, (2) **Reasoning-Driven Retrieval**: $f_{\text{rewrite}} : (p_i, \mathcal{D}_i^{\text{insuf}}) \to q_i'$ where $\mathcal{D}_i^{\text{insuf}}$ denotes insufficient documents and q_i' is the reformulated query, and (3) **Reasoning-Driven Answering**: $f_{\text{exec}} : (p_i, \mathcal{D}_i) \to (a_i, \phi_i)$ where a_i is the answer and $\phi_i \in \{0, 1\}$ guides conditional execution.

Overview and Architecture

We introduce OPERA (Orchestrated Planner-Executor Reasoning Architecture), a framework that systematically decouples strategic planning from tactical execution. As illustrated in Figures 1 and 2, OPERA operates through two core modules: the **Goal Planning Module (GPM)** containing a Plan Agent for strategic decomposition, and the **Reason-Execute Module (REM)** containing Analysis-Answer and Rewrite Agents for conditional execution and adaptive retrieval. The Plan Agent decomposes complex questions into sub-goals $\mathcal{P}$ (plan consisting of sub-goals) with placeholder dependencies. The Analysis-Answer Agent performs information sufficiency assessment ϕ (information sufficiency indicator) and answer extraction from retrieved documents $\mathcal{D}_i$ (documents retrieved for sub-goal i). The Rewrite Agent reformulates queries when information is insufficient. To optimize this multi-agent system, we introduce MAPGRPO for sequential training with role-specific rewards.

Multi-Agents Progressive Group Relative Policy Optimization (MAPGRPO)

We propose MAPGRPO, a novel variant of Group Relative Policy Optimization (GRPO) (Shao et al. 2024).

Theoretical Foundation. Given a policy π_θ parameterized by θ, GRPO optimizes objective:

$$\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{x \sim \mathcal{D}} \left[\mathbb{E}_{y_i \sim \pi_\theta(\cdot|x)} \left[A_i(x, y_i) \right] \right], \quad (1)$$

Algorithm 1: MAPGRPO Training for OPERA

Require: Dataset $\mathcal{D}$, group size G, KL coefficient β, pre-scored dataset $\mathcal{D}_{\text{scored}}$
Ensure: Optimized parameters $\{\theta^*_{\text{plan}}, \theta^*_{\text{ana}}, \theta^*_{\text{rew}}\}$

Stage 1: Plan Agent Training
for epoch $e = 1$ to E_1 **do**
  for batch $(q, \mathcal{G}) \in \mathcal{D}$ **do**
    $\mathcal{C}_{\text{plan}} \leftarrow \{c_1, ..., c_{G-1}\} \sim \pi^{(\theta)}_{\text{plan}}(\cdot|q)$ {Generate $G-1$ candidates}
    $c_{\text{best}} \leftarrow \arg\max_{c \in \mathcal{D}_{\text{scored}}(q)} r_{\text{pre}}(q, c)$ {Select best pre-scored sample}
    $\mathcal{C}_{\text{plan}} \leftarrow \mathcal{C}_{\text{plan}} \cup \{c_{\text{best}}\}$ {Add to candidate set}
    Compute rewards $\{r_{\text{plan}}(q, c)\}_{c \in \mathcal{C}_{\text{plan}}}$
    Update θ_{plan} via GRPO loss with advantages from Eq. (2)
  end for
end for
Stage 2: Analysis-Answer Agent Training
for epoch $e = 1$ to E_2 **do**
  for batch $(p, \mathcal{D}, a^*) \in \mathcal{D}_{\text{exec}}(\theta^*_{\text{plan}})$ **do**
    $\mathcal{C}_{\text{ana}} \leftarrow \{c_1, ..., c_{G-1}\} \sim \pi^{(\theta)}_{\text{ana}}(\cdot|p, \mathcal{D})$
    $c_{\text{best}} \leftarrow \arg\max_{c \in \mathcal{D}_{\text{scored}}(p)} r_{\text{pre}}(p, \mathcal{D}, c)$
    $\mathcal{C}_{\text{ana}} \leftarrow \mathcal{C}_{\text{ana}} \cup \{c_{\text{best}}\}$
    Compute rewards $\{r_{\text{ana}}(p, \mathcal{D}, c)\}_{c \in \mathcal{C}_{\text{ana}}}$
    Update θ_{ana} via GRPO loss
  end for
end for
Stage 3: Rewrite Agent Training
for epoch $e = 1$ to E_3 **do**
  for batch $(p, \mathcal{D}_{\text{insuf}}, \mathcal{G}) \in \mathcal{D}_{\text{neg}}$ **do**
    $\mathcal{C}_{\text{rew}} \leftarrow \{c_1, ..., c_{G-1}\} \sim \pi^{(\theta)}_{\text{rew}}(\cdot|p, \mathcal{D}_{\text{insuf}})$
    $c_{\text{best}} \leftarrow \arg\max_{c \in \mathcal{D}_{\text{scored}}(p)} r_{\text{pre}}(p, c)$
    $\mathcal{C}_{\text{rew}} \leftarrow \mathcal{C}_{\text{rew}} \cup \{c_{\text{best}}\}$
    Compute rewards $\{r_{\text{rew}}(p, c)\}_{c \in \mathcal{C}_{\text{rew}}}$
    Update θ_{rew} via GRPO loss
  end for
end for
return $\{\theta^*_{\text{plan}}, \theta^*_{\text{ana}}, \theta^*_{\text{rew}}\}$

where x is the input, y_i denotes the i-th generated output, and the advantage function A_i is computed relative to the group mean:

$$A_i(x, y_i) = r(x, y_i) - \frac{1}{G}\sum_{j=1}^{G} r(x, y_j). \tag{2}$$

Here, G (group size) is the number of candidates in each group, $r(x, y_i)$ is the reward for the i-th sample, and $\bar{r}(x)$ serves as a baseline computed from the current batch. The policy gradient is then:

$$\nabla_\theta \mathcal{J}_{\text{GRPO}} = \mathbb{E}\left[\sum_{i=1}^{G} A_i(x, y_i)\nabla_\theta \log \pi_\theta(y_i|x)\right]. \tag{3}$$

To prevent policy collapse, GRPO incorporates a KL divergence constraint with coefficient β (KL divergence coefficient controlling the strength of regularization):

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathcal{J}_{\text{GRPO}}(\theta) + \beta\mathbb{D}_{\text{KL}}[\pi_\theta||\pi_{\text{ref}}], \tag{4}$$

where π_{ref} denotes the reference policy (typically the initial model) and $\mathbb{D}_{\text{KL}}$ represents the Kullback-Leibler divergence.

Definition 1: MAPGRPO. Given N specialized agents $\{\pi^{(k)}\}_{k=1}^N$ with heterogeneous reward functions $\{r^{(k)}\}_{k=1}^N$, MAPGRPO optimizes each agent sequentially:

$$\theta^*_k = \arg\max_{\theta_k} \mathcal{J}_k(\theta_k|\theta^*_{<k}), \tag{5}$$

where $\theta^*_{<k} = \{\theta^*_1, ..., \theta^*_{k-1}\}$ represents the parameters of previously optimized agents, and:

$$\mathcal{J}_k(\theta_k|\theta^*_{<k}) = \mathbb{E}_{x\sim\mathcal{D}_k(\theta^*_{<k})}\left[\mathbb{E}_{y_i\sim\pi_k^{(\theta_k)}}\left[A_i^{(k)}(x, y_i)\right]\right]. \tag{6}$$

Here, $\mathcal{D}_k(\theta^*_{<k})$ represents the distribution induced by previously trained agents, ensuring each agent adapts to its actual execution environment.

Principle. MAPGRPO differs from standard GRPO in several key ways. First, it uses heterogeneous reward functions for specialized agents instead of homogeneous objectives. Second, it employs sequential optimization to address credit assignment problems in multi-agent training. Third, each agent trains on distributions induced by its predecessors, providing realistic execution conditions.

Plan Agent. The Plan agent π_{plan} decomposes queries into sub-goals with placeholder dependencies. Given query q, it generates a plan $\mathcal{P} = \{p_1, ..., p_m\}$ where each p_i may contain placeholders $[t$ from step $j]$ for $j < i$, with t indicating the expected information type (entity, location, etc.) and j (step index in placeholder) the dependency step.

The **reward function** is:

$$\begin{aligned} r_{\text{plan}}(q, \mathcal{P}) = \lambda_1 \cdot f_{\text{logic}}(q, \mathcal{P}) + \lambda_2 \cdot f_{\text{struct}}(\mathcal{P}) \\ + \lambda_3 \cdot f_{\text{exec}}(\mathcal{P}, \mathcal{E}). \end{aligned} \tag{7}$$

where f_{logic} measures decomposition validity, f_{struct} evaluates placeholder syntax correctness, f_{exec} represents end-to-end execution success, and $\lambda_1, \lambda_2, \lambda_3$ are weighting coefficients with $\lambda_1 + \lambda_2 + \lambda_3 = 1$.

Analysis-Answer Agent. The Analysis-Answer agent π_{ana} performs information sufficiency assessment and answer extraction. For sub-goal p_i and documents $\mathcal{D}_i$:

$$\pi_{\text{ana}}(p_i, \mathcal{D}_i) = \begin{cases} (y_i, a_i, c_i) & \text{if } \phi(p_i, \mathcal{D}_i) = 1 \\ (n_i, \perp, \rho_i) & \text{if } \phi(p_i, \mathcal{D}_i) = 0 \end{cases}, \tag{8}$$

where ϕ is the sufficiency indicator function, y_i denotes YES decision output, a_i is the extracted answer, c_i is confidence score, n_i denotes NO decision output, $\perp$ represents null answer, and ρ_i represents missing information type.

The **reward function** is:

$$\begin{aligned} r_{\text{ana}}(p_i, \mathcal{D}_i, o_i) = \alpha \cdot \mathbb{I}[\phi = \phi^*] + \beta \cdot \text{EM}(a_i, a_i^*) \\ + \gamma \cdot f_{\text{format}}(o_i), \end{aligned} \tag{9}$$

where o_i is the output tuple, $\mathbb{I}[\cdot]$ is the indicator function, ϕ^* is the ground-truth sufficiency, EM denotes exact match score, a_i^* is the ground-truth answer, and weights α, β, γ satisfy $\alpha + \beta + \gamma = 1$.

Rewrite Agent. The Rewrite agent π_{rew} reformulates queries when Analysis-Answer agent determines insufficient information. The **reward function** combines retrieval effectiveness and format compliance:

Method	HotpotQA		2WikiMultiHopQA		Musique		Type
	EM (%)	F1 (%)	EM (%)	F1 (%)	EM (%)	F1 (%)	
Qwen2.5-7B (No Retrieval)	18.5	26.8	16.2	23.7	4.1	9.1	Single LLM
Single-Step RAG	31.5	44.2	25.9	37.6	14.1	18.4	Naive RAG
IRCoT (Trivedi et al. 2023)	42.7	54.8	43.3	56.2	18.8	23.9	CoT
OPERA (CoT)	44.9	58.5	42.3	50.7	21.2	26.4	CoT
Adaptive-RAG (Jeong et al. 2024)	45.7	56.9	30.1	39.3	24.3	35.7	SFT
BGM (Ke et al. 2024)	41.5	53.8	44.3	55.8	19.6	26.8	RL
OPERA (MAPGRPO)	**57.3**(+11.6)	**69.5**(+11.0)	**60.2**(+15.9)	**72.7**(+16.5)	**39.7**(+15.4)	**51.9**(+16.2)	RL

[a]All SFT and RL methods are trained on mixed datasets (Musique+HotpotQA+2WikiMultiHopQA). Numbers in parentheses show improvement over best baseline.

Table 1: Main experimental results on three multi-hop QA benchmarks (underlined: best baseline).[a]

$$r_{\text{rew}}(q, q', \mathcal{R}) = \omega_1 \cdot \sqrt{\text{NDCG@}k(\mathcal{R}(q'), \mathcal{G})} + \omega_2 \cdot f_{\text{format}}(q'). \tag{10}$$

where q' is the rewritten query, $\mathcal{R}(q')$ represents documents retrieved using q', $\mathcal{G}$ denotes golden documents, NDCG@k is the normalized discounted cumulative gain at rank k, $\text{Score}_{\text{format}}$ evaluates query format quality, and weights ω_1, ω_2 satisfy $\omega_1 + \omega_2 = 1$ with $\omega_1 \gg \omega_2$ to prioritize retrieval effectiveness.

High-Score Sample Selection Strategy. To address reward sparsity in early training, we select high-scoring samples from pre-scored offline data into each candidate group. For a training instance with query q, we generate candidates $\mathcal{C}$ (set of candidates) = $\{c_1, ..., c_{G-1}, c_{\text{best}}\}$ where:

$$c_{\text{best}} = \arg \max_{c \in \mathcal{D}_{\text{scored}}} r_{\text{pre}}(q, c). \tag{11}$$

The best candidate is selected from pre-scored dataset $\mathcal{D}_{\text{scored}}$, which contains samples generated by **large-scale** LLMs and scored through end-to-end execution, ensuring at least one high-reward sample per group. Selection ratio is maintained at $1/G$ throughout training.

Theoretical Analysis

We provide rigorous theoretical foundations for OPERA's design choices. Our analysis establishes that MAPGRPO converges to local optima with rate $\mathcal{O}(1/\sqrt{T})$ under standard regularity conditions, while our reward functions are information-theoretically optimal by maximizing respective components of the mutual information decomposition $I(Q; A|D)$. Furthermore, our three-agent architecture achieves computational complexity $\mathcal{O}(h \cdot s \cdot r)$ compared to exponential $\mathcal{O}(s^h \cdot r^h)$ scaling for single-agent approaches, and our high-score sample selection strategy reduces policy gradient variance by factor $(1 - 1/G)$, accelerating convergence while maintaining exploration diversity.

4 Experimental

Experimental Setup

Datasets. We evaluate OPERA on three multi-hop reasoning benchmarks: **HotpotQA** (Yang et al. 2018) (90K questions), **2WikiMultiHopQA** (Ho et al. 2020) (150K questions), and **Musique** (Trivedi et al. 2022) (25K questions). For out-of-domain evaluation, we use **NQ** (Kwiatkowski et al. 2019) and **MultiHopRAG** (Tang and Yang 2024).

Implementation Details. We use Qwen2.5-7B-Instruct (Yang et al. 2024) for Plan and Analysis-Answer agents and other baseline's backbone, Qwen2.5-3B-Instruct (Yang et al. 2024) for the Rewrite agent, and BGE-M3 (Chen et al. 2024) as our dense retriever with top-5 document retrieval. For pre-scored dataset construction, we utilize the DeepSeek R1 (DeepSeek-AI et al. 2025) API as the data generation model.

Main Result

Baselines. We compare OPERA against methods in four categories: **Naive**: (1) Qwen2.5-7B (No Retrieval); (2) Single-Step RAG. **CoT**: (3) IRCoT (Trivedi et al. 2023); (4) OPERA (CoT Only). **SFT**: (5) Adaptive-RAG (Jeong et al. 2024). **RL**: (6) BGM (Ke et al. 2024); (7) OPERA (MAPGRPO). All SFT and RL methods are trained on mixed datasets, with Plan Agent utilizing pre-scored dataset for high-score sample selection. Baselines use defaults with Faiss (Douze et al. 2024).

Evaluation Metrics. We report **EM (%)** (exact match), **F1 (%)** (token-level overlap), **Steps** (average reasoning steps), **Latency** (processing time), and **Success Rate** (execution completion rate). Table 1 shows OPERA's performance across all benchmarks.

Performance Scales with Difficulty. OPERA shows larger improvements on more challenging datasets—63.4% relative improvement on Musique (from 24.3% to 39.7% EM) versus 25.4% relative improvement on HotpotQA (from 45.7% to 57.3% EM). This suggests our approach works better for complex multi-hop reasoning tasks.

Comparison with RL Methods. BGM applies RL to bridge retriever-LLM gaps but achieves only 19.6% EM on Musique. OPERA reaches 39.7% EM on the same dataset, indicating that specialized agent architecture provides benefits beyond RL optimization alone.

Consistent Improvements. OPERA achieves 57.3% EM on HotpotQA (versus 45.7% best baseline), 60.2% EM on 2WikiMultiHopQA (versus 44.3%), and 39.7% EM on Musique (versus 24.3%). These results span different reasoning patterns—comparison, entity traversal, and compositional reasoning—showing the approach works across var-

ied multi-hop tasks.

Configuration	EM (%)	F1 (%)
Module Ablation		
w/o Plan Agent	17.1(-22.6)	28.5(-29.5)
w/o Rewrite Agent	34.5(-5.2)	51.8(-6.2)
w/o Plan & Rewrite	16.7(-23.0)	27.2(-30.8)
Training Ablation		
CoT	21.2(-18.5)	32.1(-25.9)
SFT	24.3(-15.4)	38.2(-19.8)
GRPO	34.8(-4.9)	51.5(-6.5)
OPERA (MAPGRPO)	**39.7**	**58.0**

Table 2: Module and Training Ablation

Ablation Studies

We select MuSiQue, the most challenging dataset from our main results, for ablation experiments. All training method variants (CoT, SFT, GRPO) use the OPERA architecture but differ in their optimization approach, and are trained on decomposed sub-problems—(sub-question, documents, sub-answer) tuples—to isolate training methodology impact from architectural contributions.

Architecture Has Larger Impact Than Training. Table 2 shows that removing architectural components causes catastrophic performance drops, while training method improvements are more gradual. Removing the Plan Agent reduces performance from 39.7% to 17.1% EM—below even the untrained CoT baseline (21.2% EM)—as retrieval and reasoning modules receive poorly formed queries, leading to cascading errors. The Rewrite Agent has smaller but crucial impact (reducing EM to 34.5%): while many questions succeed through direct retrieval, it converts otherwise failed cases into successful retrievals. Most strikingly, removing both components simultaneously drops performance to 16.7% EM—worse than removing either alone—indicating that OPERA's components form an integrated system where each module depends on others functioning properly.

Training Methods Show Clear Progression. The progression across training methods follows distinct patterns. SFT improves over CoT from 21.2% to 24.3% EM through pattern learning. The jump to GRPO (34.8% EM) comes from trajectory-level optimization, where the model learns effective reasoning paths rather than just correct answers. MAPGRPO's improvement to 39.7% EM shows that specialized reward functions and sequential training better match the distinct requirements of planning, reasoning, and retrieval. However, even optimal training cannot compensate for missing architectural components—the gap between full OPERA and ablated versions persists across all training methods.

These results indicate that OPERA's performance gains stem from the synergistic blend of architectural design and training: while specialized agents with defined responsibilities provide the foundational framework, coordinated optimization through MAPGRPO ensures effective collaboration in planning, retrieval, and reasoning workflow.

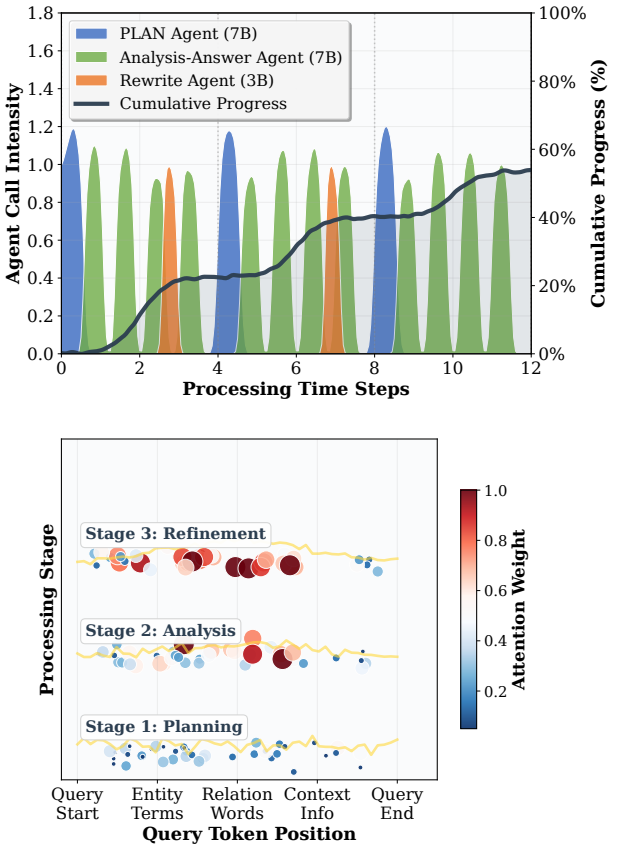

Figure 3: OPERA's runtime dynamics. (**Top**) Agent call intensity and question completion rate over processing steps. (**Bottom**) Attention visualization across query token types and processing stages.

Trajectory and Training Analysis

Runtime Dynamics and Attention Flow. Figure 3 shows OPERA's decision-making and execution patterns. The top panel shows agent activation over time: the Plan Agent (blue) initiates question cycles, the Analysis-Answer Agent (green) performs reasoning, and the Rewrite Agent (orange) activates upon retrieval failures. The black trajectory tracks the cumulative question completion rate, demonstrating performance across three questions of varying complexity. The bottom panel illustrates attention evolution across processing stages—from entity-focused planning to relation-aware analysis and context-integrated refinement.

Stable Convergence and Reward Evolution. In reward-driven GRPO, each training step samples eight diverse gold/silver candidates per input, enabling contrastive learning that steers the policy toward better outputs. These signals help agents reach or even surpass C_{best} (Eq. 11). MAPGRPO enables stable and efficient training with consistent performance improvements. Figure 4 shows typical RL characteristics: initial instability in the early steps, followed by pro-

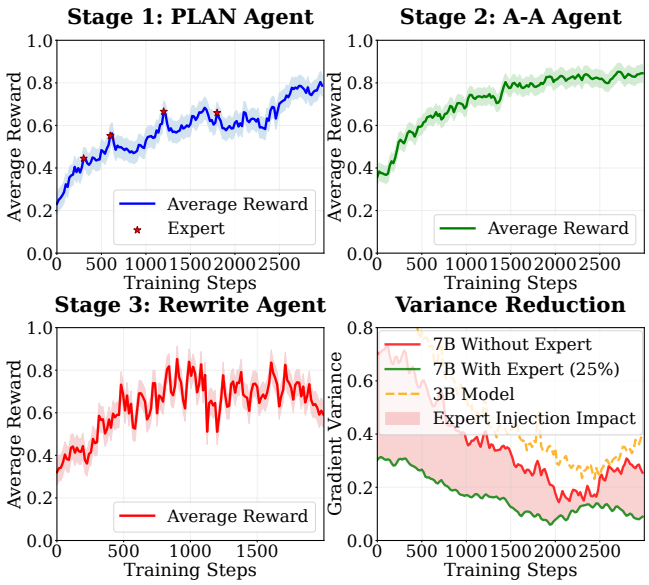

Figure 4: Training dynamics for MAPGRPO across three training stages. Top row shows average reward curves for (**top-left**) **Plan Agent** with expert samples and (**top-right**) **Analysis-Answer Agent**. Bottom row presents (**bottom-left**) **Rewrite Agent** training dynamics and (**bottom-right**) policy gradient variance reduction, with the shaded region highlighting the expert injection impact.

gressive improvement with occasional dips, particularly visible in the Rewrite Agent due to its conditional activation. Our expert demonstration strategy (Expert Injection Impact) is highly effective at reducing policy gradient variance, a key factor in training stability. The shaded region in the bottom-right panel highlights this effect, showing a significant variance reduction for the 7B model with expert injection compared to the no-expert variant, confirming our theoretical analysis. The Rewrite Agent is more unstable because it activates only on retrieval failures, yielding sparse rewards; retriever limitations further make Golden-Documents inherently difficult to obtain. Theory (Eq. 6) guarantees only local (not global) convergence and thus no performance guarantee (e.g., Musique EM remains <40% in Table 1).

Performance Analysis

Latency Analysis Across Questions. To analyze latency variations across questions of varying complexity, we evaluate 100 multi-hop test questions, focusing on agent call patterns and component contributions. Figure 5 shows that the Plan Agent maintains relatively consistent latency, while the Analysis-Answer Agent shows higher variance depending on reasoning complexity. The Rewrite Agent activates only when retrieval failures occur, confirming OPERA's adaptive behavior and efficiency for deployment.

Out-of-Domain Evaluation. Table 3 shows how training methods affect generalization. On NQ—single-hop QA over Wikipedia pages where planning is unnecessary—MAPGRPO achieves 36.6% EM while SFT drops to

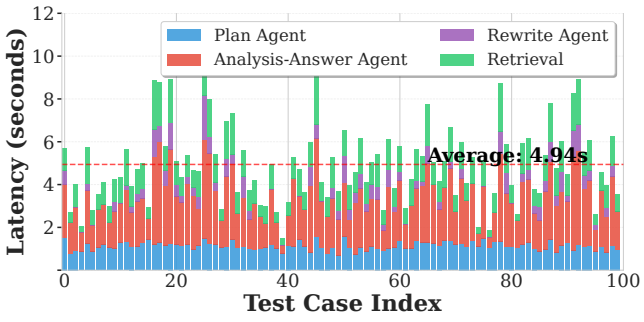

Figure 5: Component-wise latency analysis (100 **random** questions test)

Method[a]	NQ		MHRAG	
	EM (%)	F1 (%)	EM (%)	F1 (%)
CoT	23.1	29.5	42.3	51.7
SFT	19.5(-3.6)	23.8(-5.7)	44.6(+2.3)	54.2(+2.5)
GRPO	35.6(+12.5)	43.9(+14.4)	50.9(+8.6)	59.5(+7.8)
MAPGRPO	**36.6**(+13.5)	**45.1**(+15.6)	**55.7**(+13.4)	**63.8**(+12.1)

[a]All methods use the same OPERA architecture.

Table 3: Out-of-domain evaluation on single-hop (NaturalQuestions) and multi-hop patterns (MultiHopRAG).

19.5% (from 23.1% CoT baseline). MAPGRPO preserves OPERA's flexibility, allowing it to bypass planning for single-hop queries and use the Analysis-Answer Agent's training on (sub-question, document, answer) tuples to handle long documents. SFT overfits to multi-hop patterns, attempting decomposition even when not needed. On MHRAG, which has multi-hop structure similar to training data, all methods improve, with MAPGRPO reaching 55.7% EM. RL methods perform well on both single- and multi-hop tasks, whereas SFT performs worse on single-hop tasks. This suggests that trajectory-based optimization enables adaptive reasoning, while SFT induces rigid behavior.

5 Conclusion

We propose OPERA, a multi-agent framework that addresses limitations in RAG systems through specialized planning and execution roles. OPERA shows improvements—reaching 39.7% EM on Musique (63.4% relative improvement) and exceeding 60% EM on 2WikiMultiHopQA—by combining architectural design and MAPGRPO training, where specialized agents provide separation of concerns while role-specific rewards enable coordination. Ablations reveal that architecture contributes more than training, as removing the Plan Agent drops performance below untrained baselines. This suggests reasoning-driven retrieval benefits from architecture beyond optimization. Although OPERA still struggles with ambiguous decomposition and long reasoning chains, its out-of-domain generalization demonstrates robust reasoning adaptability.

Acknowledgments

This research is supported by the National Key R&D Program of China through grant 2023YFC3303800 and Procurement Project through grant E5V01511D3, the National Natural Science Foundation of China (No. 62406161), the China Postdoctoral Science Foundation (No. 2023M741950) and the Postdoctoral Fellowship Program of CPSF (No. GZB20230347).

References

Arabzadeh, N.; Yan, X.; and Clarke, C. L. A. 2021. Predicting Efficiency/Effectiveness Trade-offs for Dense vs. Sparse Retrieval Strategy Selection. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21)*, 2862–2866.

Brown, T. B.; Mann, B.; Ryder, N.; and Subbiah, M. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 1877–1901.

Chen, J.; Xiao, S.; Zhang, P.; Luo, K.; Lian, D.; and Liu, Z. 2024. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv:2402.03216.

DeepSeek-AI; Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; et al. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint arXiv:2501.12948*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 4171–4186.

Douze, M.; Guzhva, A.; Deng, C.; Johnson, J.; Szilvasy, G.; Mazaré, P.-E.; Lomeli, M.; Hosseini, L.; and Jégou, H. 2024. The Faiss Library. *arXiv preprint arXiv:2401.08281*.

Ho, X.; Duong Nguyen, A.-K.; Sugawara, S.; and Aizawa, A. 2020. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In Scott, D.; Bel, N.; and Zong, C., eds., *Proceedings of the 28th International Conference on Computational Linguistics*, 6609–6625. Barcelona, Spain (Online): International Committee on Computational Linguistics.

Hong, S.; Zhuge, M.; Chen, J.; Zheng, X.; Cheng, Y.; Zhang, C.; Wang, J.; Wang, Z.; Yau, S. K. S.; Lin, Z.; Zhou, L.; Ran, C.; Xiao, L.; Wu, C.; and Schmidhuber, J. 2024. MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework. In *Proceedings of the 12th International Conference on Learning Representations (ICLR 2024)*.

Ivison, H.; Wang, Y.; Liu, J.; Wu, Z.; Pyatkin, V.; Lambert, N.; Smith, N. A.; Choi, Y.; and Hajishirzi, H. 2024. Unpacking DPO and PPO: Disentangling Best Practices for Learning from Preference Feedback. In *Proceedings of the 38th Conference on Neural Information Processing Systems (NeurIPS 2024) Poster*.

Izacard, G.; and Grave, E. 2021. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume (EACL 2021)*, 874–880.

Jeong, S.; Baek, J.; Cho, S.; Hwang, S. J.; and Park, J. 2024. Adaptive-RAG: Learning to Adapt Retrieval-Augmented Large Language Models through Question Complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 7036–7050.

Joshi, A.; Sarwar, S. M.; Varshney, S.; Nag, S.; Agrawal, S.; and Naik, J. 2024. REAPER: Reasoning based Retrieval Planning for Complex RAG Systems. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24)*, 4621–4628.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6769–6781. Association for Computational Linguistics.

Ke, Z.; Kong, W.; Li, C.; Zhang, M.; Mei, Q.; and Bendersky, M. 2024. Bridging the Preference Gap between Retrievers and LLMs. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 10438–10451. Bangkok, Thailand: Association for Computational Linguistics.

Khattab, O.; and Zaharia, M. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 39–48.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; Toutanova, K.; Jones, L.; Kelcey, M.; Chang, M.-W.; Dai, A. M.; Uszkoreit, J.; Le, Q.; and Petrov, S. 2019. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7: 452–466.

Lee, K.; Chang, M.-W.; and Toutanova, K. 2019. Latent Retrieval for Weakly Supervised Open Domain Question Answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, volume 1, 6086–6096. Florence, Italy: Association for Computational Linguistics.

Lee, M.; An, S.; and Kim, M.-S. 2024. PlanRAG: A Plan-then-Retrieval Augmented Generation for Generative Large Language Models as Decision Makers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, 6537–6555. Seattle, WA, USA: Association for Computational Linguistics.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; Riedel, S.; and Kiela, D. 2020. Retrieval-Augmented

Generation for Knowledge-Intensive NLP Tasks. In *Advances in Neural Information Processing Systems*, volume 33, 9459–9474.

Luan, Y.; Eisenstein, J.; Toutanova, K.; and Collins, M. 2021. Sparse, Dense, and Attentional Representations for Text Retrieval. *Transactions of the Association for Computational Linguistics*, 9: 329–345.

Papangelis, A.; Wang, Y.-C.; Molino, P.; and Tur, G. 2019. Collaborative Multi-Agent Dialogue Model Training via Reinforcement Learning. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, 92–102.

Qi, X.; Zeng, Y.; Xie, T.; Chen, P.-Y.; Jia, R.; Mittal, P.; and Henderson, P. 2024. Fine-Tuning Aligned Language Models Compromises Safety, Even When Users Do Not Intend To! In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Rafailov, R.; Sharma, A.; Mitchell, E.; Ermon, S.; Manning, C. D.; and Finn, C. 2023. Direct Preference Optimization: Your Language Model is Secretly a Reward Model. In *Advances in Neural Information Processing Systems*, volume 36.

Ramamurthy, R.; Ammanabrolu, P.; Brantley, K.; Hessel, J.; Sifa, R.; Bauckhage, C.; Hajishirzi, H.; and Choi, Y. 2023. Is Reinforcement Learning (Not) for Natural Language Processing: Benchmarks, Baselines, and Building Blocks for Natural Language Policy Optimization. In *Proceedings of the International Conference on Learning Representations (ICLR 2023)*.

Rezaei, M. R.; Hafezi, M.; Satpathy, A.; Hodge, L.; and Pourjafari, E. 2024. AT-RAG: An Adaptive RAG Model Enhancing Query Efficiency with Topic Filtering and Iterative Reasoning.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal Policy Optimization Algorithms. *arXiv preprint arXiv:1707.06347*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, K., Y. Wu, Y.; and Guo, D. 2024. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300*.

Stiennon, N.; Ouyang, L.; Wu, J.; Ziegler, D. M.; Lowe, R.; Voss, C.; Radford, A.; Amodei, D.; and Christiano, P. F. 2020. Learning to Summarize with Human Feedback. In *Advances in Neural Information Processing Systems*, volume 33, 3008–3021.

Tang, Y.; and Yang, Y. 2024. MultiHop-RAG: Benchmarking Retrieval-Augmented Generation for Multi-Hop Queries. In *First Conference on Language Modeling (COLM)*.

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

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 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: Long Papers*, 10014–10037. Toronto, Canada: Association for Computational Linguistics.

Uc-Cetina, V.; Navarro-Guerrero, N.; Martin-González, A.; Weber, C.; and Wermter, S. 2022. Survey on reinforcement learning for language processing. *Artificial Intelligence Review*, 56(2): 1543–1575.

Wang, Y.; Kordi, Y.; Mishra, S.; Liu, A.; Smith, N. A.; Khashabi, D.; and Hajishirzi, H. 2023. Self-Instruct: Aligning Language Models with Self-Generated Instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 13484–13508.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q. V.; and Zhou, D. 2022. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. In *Advances in Neural Information Processing Systems*, volume 35, 24824–24837. Curran Associates, Inc.

Yang, A.; Li, A.; Yang, B.; and *et al.* 2024. Qwen2.5 Technical Report. *arXiv preprint arXiv:2412.15115*.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. HotpotQA: A Dataset for Diverse, Explainable Multi-hop Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2369–2380.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023a. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. In *Advances in Neural Information Processing Systems*, volume 36.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023b. ReAct: Synergizing Reasoning and Acting in Language Models. In *Proceedings of the International Conference on Learning Representations (ICLR)*.