REAP: Enhancing RAG with Recursive Evaluation and Adaptive Planning for Multi-Hop Question Answering

Yijie Zhu[*1], **Haojie Zhou**[*1], **Wanting Hong**[1], **Tailin Liu**[1], **Ning Wang**[†1]

[1]The School of Artificial Intelligence and Computer Science, Jiangnan University
ningwang@jiangnan.edu.cn

Abstract

Retrieval-augmented generation (RAG) has been extensively employed to mitigate hallucinations in large language models (LLMs). However, existing methods for multi-hop reasoning tasks often lack global planning, increasing the risk of falling into local reasoning impasses. Insufficient exploitation of retrieved content and the neglect of latent clues fail to ensure the accuracy of reasoning outcomes. To overcome these limitations, we propose **R**ecursive **E**valuation and **A**daptive **P**lanning (REAP), whose core idea is to explicitly maintain structured sub-tasks and facts related to the current task through the Sub-task Planner (SP) and Fact Extractor (FE) modules. SP maintains a global perspective, guiding the overall reasoning direction and evaluating the task state based on the outcomes of FE, enabling dynamic optimization of the task-solving trajectory. FE performs fine-grained analysis over retrieved content to extract reliable answers and clues. These two modules incrementally enrich a logically coherent representation of global knowledge, enhancing the reliability and the traceability of the reasoning process. Furthermore, we propose a unified task paradigm design that enables effective multi-task fine-tuning, significantly enhancing SP's performance on complex, data-scarce tasks. We conduct extensive experiments on multiple public multi-hop datasets, and the results demonstrate that our method significantly outperforms existing RAG methods in both in-domain and out-of-domain settings, validating its effectiveness in complex multi-hop reasoning tasks.

Code — https://github.com/Deus-Glen/REAP

Introduction

Large Language Models (LLMs) have demonstrated advanced capabilities across various natural language processing (NLP) tasks (Touvron et al. 2023; Fan et al. 2024; Gao et al. 2025). However, their reliance on parameterized knowledge renders them prone to factually incorrect answers due to outdated information or hallucinations (Huang et al. 2025; Cheng et al. 2024). To mitigate these limitations, Retrieval-augmented generation (RAG) has emerged as an effective approach for enhancing LLM performance in knowledge-intensive tasks by dynamically incorporating external non-parameterized knowledge sources (Lewis et al. 2020; Gao et al. 2023b). While traditional RAG systems can efficiently handle simple single-hop question-answering (QA) tasks through a single retrieval-generation paradigm (Ye et al. 2024; Roy et al. 2024), they struggle with multi-hop question-answering (MHQA) that require integrating information scattered across multiple documents (He et al. 2024; Trivedi et al. 2022a; Jiang et al. 2023b).

To overcome the challenges of MHQA, recent studies have adopted multi-round retrieval and iterative refinement strategies to progressively gather and filter relevant information. These approaches effectively integrate knowledge from multiple sources, significantly strengthening the reasoning capabilities of LLMs in MHQA tasks (Tang and Yang 2024; Yang et al. 2024b; Teng et al. 2025). Nevertheless, such methods often suffer from inefficiencies in reasoning trajectory planning and limited depth of information exploitation. To identify optimal reasoning trajectories, some studies have introduced complex search algorithms such as Monte Carlo Tree Search (MCTS) (Dong et al. 2024; Li et al. 2024), but these typically come with significant computational overhead. Moreover, when solving sub-queries, models tend to extract direct answers, ignoring latent clues that are crucial to the final solution (Ye et al. 2025; Wang et al. 2024a). Furthermore, to reinforce end-to-end reliability, recent efforts have introduced external components such as rerankers and decision-makers to improve retrieval accuracy (Glass et al. 2022; Jeong et al. 2024), but such approaches often come at the expense of increased system complexity and diminished interpretability (Su et al. 2025).

To address the above limitations, we propose **R**ecursive **E**valuation and **A**daptive **P**lanning (REAP), a novel iterative framework to enhance the performance of RAG. Our core insight is that the incremental decomposition of complex queries into sub-queries may lead to reasoning impasses, while missing or inaccurate facts can cause deviations from the correct reasoning trajectory (Zhang et al. 2025). To maintain coherent and reliable reasoning throughout the process, we explicitly maintain two critical knowledge sets: structured sub-tasks and facts, which serve as the foundation for guiding reasoning. Specifically, we introduce the Sub-Task Planner (SP) module that offers a global perspective, plans intermediate reasoning steps and dynam-

[*]These authors contributed equally.
[†]Corresponding author.

ically guides the overall reasoning trajectory. Then, the Fact Extractor (FE) module is proposed to perform retrieval, integrate retrieved content with the structured facts for reasoning, thereby providing precise and complete facts and continuously expanding global context. These two modules are functionally decoupled yet tightly coordinated, continuously advancing the reasoning process. This modular and synergistic design allows the framework to rapidly identify optimal reasoning trajectories in complex scenarios while ensuring the reliability of facts, thereby enhancing the overall framework's performance.

Our contributions are summarized as follows:

1. We propose REAP, a dual-module framework where the globally-aware SP module dynamically guides the reasoning trajectory, while the FE module continuously enriches the knowledge base with reliable facts, creating a mutually reinforcing cycle that enhances reasoning capabilities.
2. We introduce a unified task paradigm that facilitates knowledge transfer from data-rich routine planning to data-scarce critical replanning via multi-task fine-tuning, significantly enhancing the framework's robustness and strategic intelligence.
3. We separate simple and complex reasoning into distinct sub-modules for SP, enabling the simple sub-module to be replaced with a more lightweight model to improve efficiency while preserving performance.
4. We conduct extensive experiments on multiple MHQA datasets, demonstrating the effectiveness and robustness of REAP.

Related Work

RAG

Traditional RAG frameworks have been introduced to mitigate issues such as knowledge obsolescence and hallucinations in LLMs by incorporating external knowledge sources. Early representative methods like Standard RAG (Lewis et al. 2020) follow a static retrieve-then-generate paradigm, typically leveraging either sparse retrieval (Robertson, Zaragoza et al. 2009) or dense retrieval (Karpukhin et al. 2020; Wang et al. 2022) to refine the factual correctness of generated responses. Although these methods have promised accuracy of QA, their effectiveness in complex reasoning tasks remains limited, particularly in multi-hop reasoning tasks due to the lack of dynamic interaction and reasoning mechanisms. To address the above challenges, researchers have explored refinements to single-round RAG from multiple perspectives, aiming to better support complex reasoning tasks. Prior to retrieval, query enhancement and rewriting are used to optimize information alignment (Gao et al. 2023a; Ma et al. 2023; Chan et al. 2024). During the retrieval process, dynamic strategies guided by confidence intervals have been introduced to reduce redundant computations and improve retrieval efficiency (Jiang et al. 2023b; Su et al. 2024; Deng et al. 2025). Following retrieval, the retrieved content is further refined through end-to-end training (Shi et al. 2023) or post-processing modules (Glass et al. 2022; Jiang et al. 2023a; Kim et al. 2024) to enhance generation quality. Despite these advancements, the inherent inadequacy of traditional RAG with single-round retrieval in handling complex, multi-hop reasoning tasks has prompted a focus on iterative retrieval-generation architectures.

Iterative RAG

Iterative RAG frameworks operate by interleaving retrieval and generation across multiple rounds to incrementally synthesize evidence and construct reasoning chains in a context-sensitive and adaptive manner (Ram et al. 2023; Yang et al. 2024a). To achieve this, A fundamental strategy is problem decomposition, which partitions complex questions into simpler sub-queries, retrieves supporting evidence for each sub-query, and subsequently integrates the sub-answers to facilitate multi-hop reasoning (Xu et al. 2024; Shi et al. 2024). To further advance the reasoning planning process, a number of studies introduce search-based algorithms, such as MCTS, to simulate and evaluate alternative reasoning branches and identify optimal reasoning strategies (Dong et al. 2024; Li et al. 2024). Another line of research integrates chain-of-thought prompting with multi-round retrieval to guide the model in progressively constructing the final answer (Yao et al. 2023; Press et al. 2022; Wang et al. 2024b). Representative methods such as IRCoT (Trivedi et al. 2022a) and Iter-RetGen (Shao et al. 2023) dynamically generate sub-queries, retrieve relevant documents, and iteratively refine the reasoning trajectory throughout the generation process. Additionally, to strengthen strategic control capabilities during multi-round retrieval, some methods incorporate self-reflection mechanisms, enabling the model to autonomously determine whether to continue retrieving or proceed with answer generation (Asai et al. 2023), while others rely on reinforcement learning frameworks to learn retrieval and generation policies based on interactive feedback (Jin et al. 2025a; Song et al. 2025). However, these methods often fall short in global planning and facts extraction, leading to brittle and incoherent reasoning trajectories. In contrast to the aforementioned methods, our proposed REAP substantially enhances reasoning ability and robustness in multi-hop reasoning tasks through a recursive feedback loop between SP and FE.

Preliminary

For multi-hop scenarios, we formally define the iterative RAG task as follows: Given a query Q and an external corpus $\mathcal{C}$, the objective is to generate a factual answer A. This task is typically operationalized as an iterative reasoning process, where at each step t, a sub-query q_t is formulated to gather a specific piece of information. A retriever function Retriever$(\cdot)$, then fetches a set of relevant documents $D_t \subset \mathcal{C}$ based on the sub-query:

$$D_t = \text{Retriever}(q_t; \mathcal{C}) \tag{1}$$

An LLM, M_θ parameterized by θ, processes these documents, often conditioned on the history of previous interac-

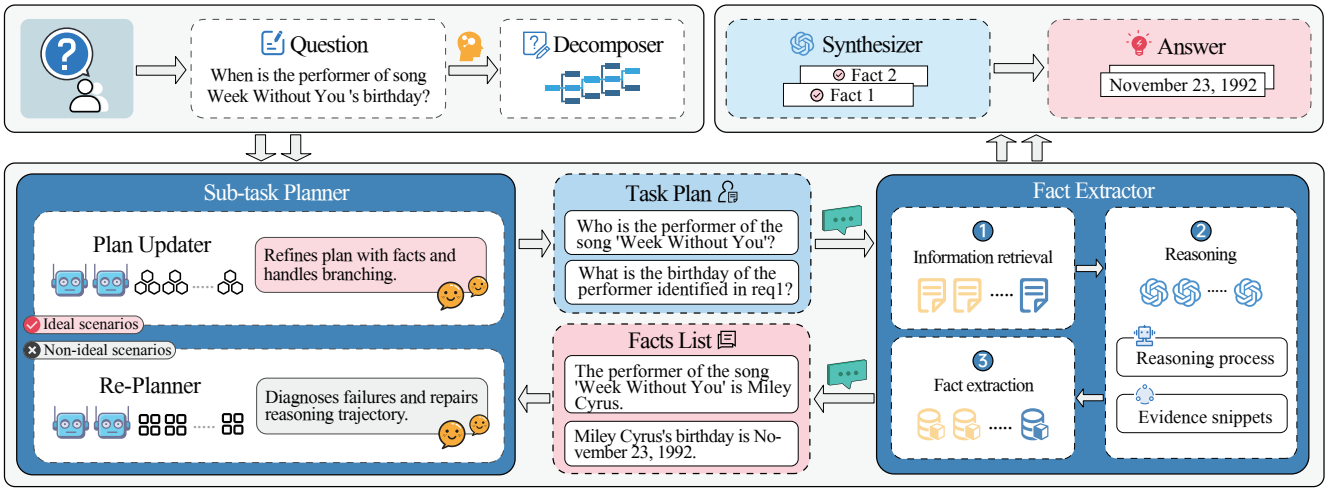

Figure 1: Overall framework of REAP. After the Decomposer breaks down the question, the Sub-task Planner and Fact Extractor iterate to guide the reasoning trajectory and gather the facts, and the Synthesizer produces the final answer.

tions $H_{t-1} = \{Q, (q_1, A_1), \ldots, (q_{t-1}, A_{t-1})\}$, to generate a sub-answer A_t:

$$A_t = M_\theta(\text{GenerateSubAnswer} \mid Q, D_t, H_{t-1}) \quad (2)$$

The history is then updated with the new pair (q_t, A_t), and the process repeats until a termination condition is met. Finally, a synthesis function generates the final answer A based on the complete interaction history H_{final}:

$$A = M_\theta(\text{Synthesize} \mid Q, H_{\text{final}}) \quad (3)$$

This iterative paradigm allows the model to progressively build a chain of reasoning by decomposing the original complex query into a sequence of sub-queries.

Method

Framework Overview

Our proposed framework, REAP, reframes the multi-hop reasoning process from a linear pipeline into a dynamic, state-driven loop. The architecture, illustrated in Figure 1, is designed around the core principle of explicitly decoupling planning from fact-gathering to enhance robustness and interpretability. It primarily consists of two synergistic modules operating in a recursive loop: the Sub-task Planner (SP) and the Fact Extractor (FE).

The overall workflow proceeds as follows: Given a complex query Q, an initial Decomposer first generates a structured task plan, $\mathcal{P}_0 = \{p_1, p_2, \ldots, p_N\}$. Each sub-task $p_i \in \mathcal{P}_0$ is a tuple $(id_i, q_i, deps_i)$, representing its unique identifier, query, and dependencies. This plan, along with an initially empty facts list $\mathcal{F}_0 = \emptyset$, defines the initial state of the framework.

The core of REAP is an iterative process that continues until a termination condition is met. At each step t:

1. The SP, acting as the strategic core, analyzes the current state $(\mathcal{P}_{t-1}, \mathcal{F}_{t-1})$ to determine a set of executable next actions, Actions_t.
2. For each sub-task $p_i \in \text{Actions}_t$, the FE performs retrieval and grounded reasoning to produce a new fact object, f_i.
3. The facts list is updated: $\mathcal{F}_t = \mathcal{F}_{t-1} \cup \{f_1, f_2, \ldots\}$. The SP then receives this feedback to generate the subsequent plan, $\mathcal{P}_t$.

This recursive loop, where the SP guides the FE and the FE's findings inform the SP's next planning cycle, continues until the plan is fully resolved. Finally, a Synthesizer generates the conclusive answer A by reasoning over the final, comprehensive facts list $\mathcal{F}_{final}$ and the original query Q:

$$A = M_\theta(\text{Synthesize} \mid Q, \mathcal{F}_{final}) \quad (4)$$

This modular and iterative design allows REAP to navigate complex reasoning trajectories, recover from errors, and construct a fully traceable line of evidence.

Sub-task Planner (SP)

The SP serves as the strategic core of the REAP framework, designed to overcome the limitations of myopic, step-by-step reasoning that often leads to local impasses or deadlocks. By receiving and dynamically maintaining the complete initial task plan $\mathcal{P}_0$, the SP retains a global perspective throughout the process. A key innovation of the SP is its state-aware, modular design. Based on the fulfillment level l_t of the new fact f_t returned by the FE, the SP dispatches the task to one of two specialized sub-modules:

1. **Plan Updater**: This sub-module handles ideal scenarios where reasoning progresses as expected (e.g., l_t is `DirectAnswer`). It performs deterministic and rule-based updates to maintain the task plan. Its primary functions are:
   - *Fact Substitution*: It systematically rewrites pending sub-tasks by substituting abstract placeholders with concrete entities derived from newly acquired facts.

This ensures that subsequent queries are self-contained and fully grounded.

- *Plan Forking*: In cases where a sub-task yields a set of multiple valid sub-answers, this function programmatically duplicates the subsequent dependent sub-tasks into parallel branches, ensuring all valid reasoning trajectories are exhaustively explored.

2. **Re-Planner**: This sub-module is invoked to handle non-ideal scenarios (e.g., l_t is `PartialClue` or `Failed`), acting as the critical reasoning and recovery mechanism. Its hierarchical responsibilities are:
   - *Pragmatic Sufficiency Assessment*: Its first and most crucial task is to evaluate whether a partially fulfilled sub-task, despite its incompleteness, is functionally sufficient to satisfy the informational requirements of subsequent reasoning steps. This assessment is made by analyzing the dependencies of the downstream plan relative to the ultimate query Q. If the partial information is deemed sufficient, the sub-task is considered resolved, thus preventing inefficient, perfectionist search loops.
   - *Scoped Plan Repair*: Only if the acquired information is assessed as insufficient does the Re-Planner proceed with plan repair. It first diagnoses the failure's scope—differentiating between a localized issue (e.g., a poorly formulated query) and a systemic flaw (e.g., an irrelevant reasoning trajectory). For localized issues, it performs a minor adjustment by refining the sub-task's query. For systemic flaws, it executes a major overhaul by pruning the invalid branch and injecting a new, more logical sequence of sub-tasks to create an alternative solution path.

At an iterative step t, given the task plan $\mathcal{P}_{t-1}$ and facts list $\mathcal{F}_{t-1}$ from the previous state, along with the new fact f_t, the decision process of the SP can be formalized as:

$$(\mathcal{P}_t, \text{Actions}_t) = \text{SP}(\mathcal{P}_{t-1}, \mathcal{F}_{t-1} \cup \{f_t\}, Q) \tag{5}$$

where Actions_t is the list of concrete sub-tasks to be executed next. The internal dispatch logic of the SP is determined by the fulfillment level l_t of the new fact f_t:

$$\text{SP} \leftarrow \begin{cases} \text{Plan Updater} & \text{if } l_t = \texttt{DirectAnswer} \\ \text{Re-Planner} & \text{otherwise} \end{cases} \tag{6}$$

This dual-module design allows REAP to handle routine progress with high efficiency while possessing robust, intelligent capabilities to recover from complex failures and adapt its strategy in real-time.

Fact Extractor (FE)

The facts list $\mathcal{F}$ serves as the evolving knowledge foundation for the entire reasoning process; thus, its reliability and comprehensiveness are paramount. The Fact Extractor module is designed to extract high-fidelity, structured facts from retrieved documents. To mitigate hallucinations and enhance traceability, we require the LLM to not only provide a conclusive statement but also articulate its reasoning process and cite direct textual evidence.

A key function of the FE is its ability to discern not just direct answers but also latent clues that may be crucial for subsequent reasoning steps. For a given sub-query q_t from a sub-task $p_t \in \text{Actions}_t$, the FE process is defined as follows:

First, a retriever function fetches a set of relevant documents D_t from an external corpus $\mathcal{C}$. Next, an LLM M_θ processes these documents to generate a new structured fact object f_t. Crucially, the model is conditioned not only on the current query and retrieved documents but also on the historical facts $\mathcal{F}_{t-1}$. This contextual conditioning allows the model to perform more sophisticated reasoning, such as coreference resolution and identifying relationships between new information and previously established facts. The generation process is formulated as:

$$f_t = M_\theta(\text{ExtractFact} | q_t, D_t, \mathcal{F}_{t-1}) \tag{7}$$

The fact object f_t is a structured tuple designed to capture the richness of the extracted information:

$$f_t = (s_t, e_t, r_t, l_t) \tag{8}$$

where:

- s_t is the core statement, a concise, self-contained factual assertion.
- $e_t \subseteq D_t$ is the set of direct textual evidence snippets that ground the statement s_t.
- r_t is the model's reasoning process, a chain-of-thought explanation of how s_t is derived from e_t.
- l_t is the fulfillment level, a categorical label that classifies the quality of the extracted fact. This level is crucial for the SP's subsequent decision-making.

The statement s_t and fulfillment level l_t are then used to directly guide the SP module's next planning cycle.

Multi-task Fine-tuning

To enhance the performance of the planning-related modules within the REAP framework, especially for the Re-Planner, which suffers from data scarcity due to its lower invocation frequency, we devise a multi-task fine-tuning strategy.

The core insight is that despite their varying difficulty, the Decomposer, Plan Updater, and Re-Planner modules share a significant functional commonality: they all require the model to generate or modify a structured task plan based on existing information. We leverage this commonality by consolidating their respective datasets, $\mathcal{D}_{\text{decomp}}, \mathcal{D}_{\text{update}}, \mathcal{D}_{\text{replan}}$, for joint training of a single planning model M_ϕ.

The training objective is to minimize a weighted joint loss function $\mathcal{L}_{\text{multi}}$:

$$\min_\phi \mathcal{L}_{\text{multi}}(\phi) = \sum_{task \in \mathcal{T}} \lambda_{task} \mathbb{E}_{(x,y)\sim\mathcal{D}_{task}} [\mathcal{L}_{\text{task}}(M_\phi(x), y)] \tag{9}$$

where $\mathcal{T} = \{\text{decomp}, \text{update}, \text{replan}\}$, λ_{task} is the task-specific weight, set to 1, and $\mathcal{L}_{\text{task}}$ is the corresponding loss function. This paradigm enables effective knowledge transfer from data-rich tasks (Decomposer, Plan Updater) to the data-scarce Re-Planner task. This significantly enhances the robustness and accuracy of the Re-Planner in complex, anomalous scenarios, thereby elevating the overall intelligence of the REAP framework.

Models	HotpotQA			2Wiki			MuSiQue‡			Bamboogle‡		
	CEM	F1	ACC†	CEM	F1	ACC†	CEM	F1	ACC†	CEM	F1	ACC†
Naive	24.6	31.8	34.6	31.6	35.1	32.8	6.2	10.6	10.0	13.6	17.7	21.6
Standard RAG	41.6	48.6	51.2	36.2	38.5	38.2	9.8	13.5	12.6	20.8	30.9	32.0
FT-Standard RAG	53.2	61.2	65.4	53.6	57.1	56.2	18.2	26.4	22.6	42.4	54.4	51.2
SuRe	29.4	33.8	40.4	24.4	24.6	28.2	5.8	11.6	11.4	10.4	19.1	20.0
IRCoT	42.2	51.4	54.8	41.2	36.5	41.8	18.2	18.6	20.4	35.2	30.1	44.8
Iter-Retgen	45.0	43.0	54.6	42.0	32.2	41.0	14.2	18.6	20.6	19.2	27.2	28.0
SearChain	39.8	39.6	50.4	50.6	46.7	50.0	20.6	23.6	26.0	47.2	49.4	53.6
Search-R1	44.8	52.7	57.0	47.4	49.7	50.8	20.0	25.7	25.8	40.8	50.8	49.6
R1-Searcher	56.8	63.4	68.4	68.4	69.4	70.0	29.2	33.8	34.4	48.8	58.0	56.0
REAP	**59.2**	**68.0**	**72.4**	**79.2**	**79.6**	**81.6**	**33.6**	**38.3**	**40.8**	**49.6**	**65.2**	**65.6**

Table 1: Performance comparisons between REAP and the baselines on four MHQA benchmarks. Bold numbers indicate the best result, while underlined number indicates the second-best result. ‡ represents out-of-domain datasets.

Experiments

Datasets and Metrics

We conduct evaluations of our method on four multi-hop datasets: HotpotQA (Yang et al. 2018), 2WikiMultihopQA (Ho et al. 2020), MuSiQue (Trivedi et al. 2022b), and Bamboogle (Press et al. 2022). HotpotQA and 2WikiMultihopQA are in-domain benchmarks, as parts of their training sets are used for model training, whereas MuSiQue and Bamboogle serve as out-of-domain benchmarks to assess the generalization capabilities of our method. For the first three datasets, we randomly sample 500 examples from each validation split as test sets. For Bamboogle, since it only contains 125 examples, we use the entire test set for evaluation. Regarding evaluation metrics, we adopt Cover Exact Match (CEM), F1 score, and ACC† (with an LLM serving as the judge). Detailed datasets and metrics descriptions are provided in the Appendix.

Baselines

We compare REAP against the following baselines. The Naive method directly generates answer without any retrieval. The Standard RAG (Lewis et al. 2020) performs one-step retrieval and concatenates the documents with the question for answer generation. Additionally, we fine-tune the Standard RAG with the same training data adopted in our method, and name it FT-Standard RAG. SuRe (Kim et al. 2024) ranks answer candidates by generating conditioned summaries of the retrieved content. IRCoT (Trivedi et al. 2022a), Iter-RetGen (Shao et al. 2023) and SearChain (Xu et al. 2024) combine reasoning chains with multi-round retrieval to form the reasoning trajectory. Search-R1 (Jin et al. 2025a) and R1-Searcher (Song et al. 2025) employ template-guided prompting, reinforcement learning, and summarization mechanisms to achieve more refined integration of evidence and answer generation.

Implementation Details

In our experiments, REAP and other non-fine-tuned baseline methods use Llama-3.1-8B-Instruct (Grattafiori et al. 2024) as the generator and are evaluated with FlashRAG (Jin et al. 2025b). For methods involving fine-tuning, we use the model checkpoints provided by the authors. We adopt the corpus provided by CoRAG (Wang et al. 2024b), which is based on the English Wikipedia provided by KILT, containing approximately 360,000 passages. We use e5-large-v2 (Wang et al. 2022) as the main retriever, with the top-5 results returned for each query. For all multi-round methods, we set the maximum number of iterations to 5. We randomly select 7,000 samples from HotpotQA and WikiMultihopQA and run REAP using GPT-4 (Achiam et al. 2023) to collect training data. After filtering, we finally select 5,556 samples as the training set, of which 2,988 are from HotpotQA and 2,568 are from 2WikiMultihopQA. Detailed training setting is provided in the Appendix.

Main Results

Table 1 presents the experimental results of REAP and other baseline methods on four representative MHQA datasets. Specifically, we observe the following:

1. Iterative interaction leads to substantial performance gains. Compared with the Standard RAG method, REAP improves the F1 score on HotpotQA from 48.6% to 68.0%, demonstrating the superiority of the iterative approach for solving MHQA. More importantly, REAP maintains a significant advantage over the FT-Standard RAG, achieving a 6.8% improvement in F1 score, indicating that the performance gain is not merely attributable to the training data but primarily stems from our proposed iterative reasoning and evidence extraction framework.
2. REAP significantly outperforms existing multi-round methods. It impressively surpasses the top-performing R1-Searcher method in FlashRAG on all datasets, achieving F1 score improvements of 4.6% on HotpotQA and 10.2% on 2WikiMultihopQA. This demonstrates that our method effectively enhances the model's ability to reason and extract factual evidence.
3. The model exhibits strong generalization capability. REAP is trained on only 5,556 samples from HotpotQA

Models	HQA	WQA	MQA‡	BQA‡
	F1	F1	F1	F1
w/o replan	64.9	78.6	34.2	61.6
w/o verify	65.1	78.0	34.8	60.8
w/o clue	64.6	76.5	35.2	62.7
REAP	68.0	79.6	38.3	65.2

Table 2: The ablation study on HotpotQA (HQA), 2WikimultihopQA (WQA), MuSiQue (MQA) and Bamboogle (BQA).

and 2WikiMultihopQA. It not only excels on these in-domain datasets but also achieves the best scores on out-of-domain datasets, showcasing its powerful generalization. This suggests that our training process enables the model to learn the fundamental skills of reasoning and facts extraction, rather than simply memorizing a generation pattern, thus leading to robust performance on unseen data.

Analysis

Ablation Studies

We conduct ablation studies on the same four multi-hop datasets to analyze the effectiveness of key module designs within our proposed REAP framework. Meanwhile, we also investigate the contribution of our multi-task fine-tuning strategy to the framework.

Module Ablation To evaluate the effectiveness of our individual mechanisms, we disable key functionalities from the modules. For the SP module, we ablate the Re-Planner sub-module (w/o replan), which is designed to handle challenging scenarios. For the FE module, we ablate the logical reasoning and verification mechanism (w/o verify) and the clue feedback mechanism (w/o clue), respectively. The results are presented in Table 2.

We observe that the full REAP framework consistently outperforms all ablated variants, confirming the positive contribution of each mechanism. The most significant performance degradation is observed in w/o replan, with F1 score dropping by 3.1% on HQA, 4.1% on MQA, and 3.6% on BQA. This substantial decline, particularly on the more complex MQA and BQA datasets, underscores the criticality of the Re-Planner's ability to dynamically correct and reroute the reasoning trajectory when encountering failures or suboptimal steps.

Removing the verification mechanism (w/o verify) consistently lowers performance, with a particularly sharp drop of 4.4% F1 score on BQA. This highlights the necessity of ensuring factual accuracy at each iterative step. Without robust verification, incorrect or hallucinatory information can be propagated, leading to a cascade of errors that derail the entire reasoning trajectory.

Disabling the clue feedback mechanism (w/o clue) also results in a notable performance drop across all benchmarks, for instance, a decrease of 3.4% F1 score on HQA and 3.1% on WQA. This demonstrates that the model's capacity to identify and leverage partial or latent clues, beyond just direct answers, is vital for constructing a comprehensive factual basis required for the final answer synthesis.

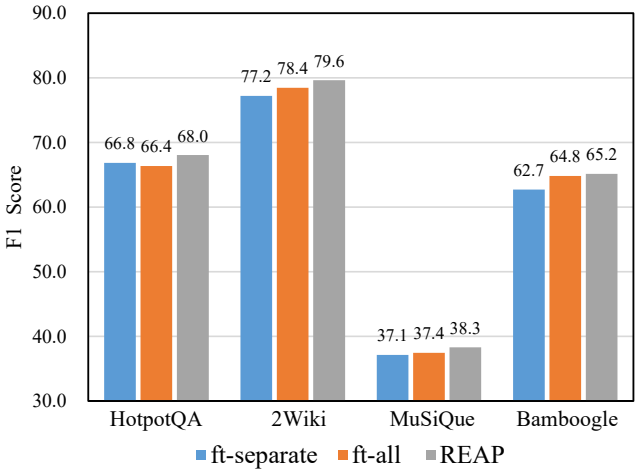

Figure 2: Ablation study on the effectiveness of Multi-task Fine-tuning.

Multi-task Fine-tuning Ablation To validate the effectiveness of our proposed multi-task fine-tuning strategy, we conduct an ablation study. We compare REAP with two settings: ft-separate, where we fine-tune a dedicated model for each module (Decomposer, Plan Updater, and Re-Planner) using only its corresponding training data; and ft-all, where a single model is jointly fine-tuned on the combined data from all modules.

The results, presented in Figure 2, reveal the clear advantages of our method. The ft-separate setting consistently yields the lowest performance, lagging behind REAP by a significant margin of 2.5% F1 score on Bamboogle and 2.4% on 2Wiki. This demonstrates that training on isolated tasks prevents the model from learning the underlying correlations and generalizable reasoning patterns shared across the planning modules, thereby limiting its capabilities.

Crucially, our REAP framework, which employs the carefully designed multi-task fine-tuning strategy, consistently achieves the highest scores across all benchmarks. It surpasses the ft-all setting by 1.2% F1 score on 2Wiki and 0.9% on MuSiQue. This final improvement highlights that merely combining training data is insufficient. Our multi-task fine-tuning strategy more effectively facilitates the transfer of robust planning capabilities to the data-scarce but critical Re-Planner module, confirming the effectiveness and rationale of our methodology.

Further Analysis

Efficiency Analysis In practical applications, in addition to pursuing the highest performance, inference efficiency (i.e., latency and computational cost) is also a key consideration. Our REAP framework dynamically assigns simple and complex scenarios to different modules. The Plan Updater

Models	HotpotQA			2Wiki			MuSiQue			Bamboogle		
	CEM	F1	ACC†	CEM	F1	ACC†	CEM	F1	ACC†	CEM	F1	ACC†
Naive_{70B}	32.4	39.7	40.4	36.8	39.3	38.8	11.6	15.1	17.4	37.6	37.0	44.0
StandardRAG_{70B}	51.0	56.3	61.2	46.8	47.2	46.6	16.8	21.1	24.8	35.2	46.1	46.4
SuRe_{70B}	35.8	43.1	46.0	27.6	34.6	33.2	12.2	20.4	17.4	10.4	19.1	20.0
IRCoT_{70B}	55.8	49.5	60.4	57.0	33.1	35.8	21.0	19.4	24.3	32.0	38.8	40.8
Iter-RetGen_{70B}	53.8	47.6	65.0	51.4	41.5	53.6	17.2	21.6	24.6	41.6	47.5	52.8
REAP_{70B}	**63.2**	**65.5**	**73.6**	**73.8**	**70.0**	**72.2**	**37.6**	**37.2**	**42.6**	**54.4**	**61.6**	**63.2**

Table 3: Performance comparison of REAP and non-fine-tuned baselines on four MHQA benchmarks under the Llama-3.1-70B-Instruct setting. Bold numbers indicate the best result, while underlined number indicates the second-best result.

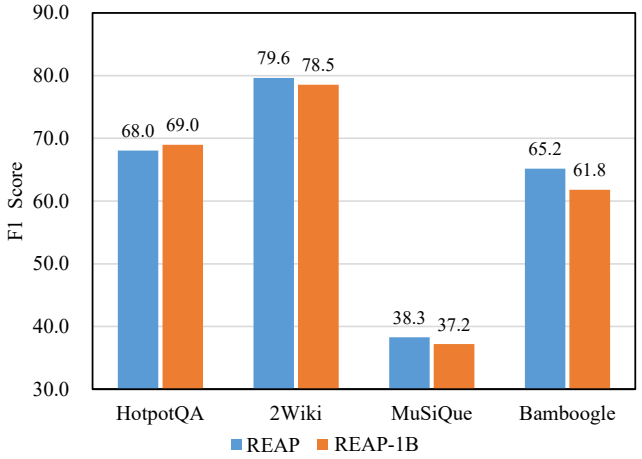

Figure 3: Module substitution experiment validating the efficiency gains of asymmetric configuration.

sub-module for simple cases is relatively simple, and according to our statistics, about 85% of steps are classified as simple scenarios. To improve overall efficiency, we replace the base model in the Plan Updater sub-module with Llama-3.2-1B-Instruct, fine-tuned on the same data. As shown in Figure 3, even with an 1B-parameter model, the task can still be accomplished with relatively high accuracy. At the same time, the framework benefits from the faster inference speed of the smaller model, resulting in improved overall efficiency. This discovery fully demonstrates the unique advantages of the REAP framework's asymmetric modularity design: it allows us to flexibly configure different scales of computing resources for different modules, achieving a delicate balance between performance and efficiency.

Performance on Larger Models To investigate the scalability of the REAP framework and its interaction with the capabilities of the base model, we conduct further experiments, replacing the base model from Llama-3.1-8B-Instruct with the more powerful Llama-3.1-70B-Instruct, and comparing its performance with five other non-fine-tuning methods. All methods are not fine-tuned to ensure fairness. As shown in Table 3. We observe two key phenomena. First, when the base model is replaced with Llama-3.1-70B-Instruct, most methods achieve improved performance. For example, the F1 score of Standard RAG on HotpotQA increases from 39.7% to 56.3% compared to Naive, confirming the general benefit of employing more powerful LLMs for complex MHQA tasks. More importantly, our REAP framework retains its leading performance when built upon a stronger base model. This result provides compelling evidence for the superiority of the REAP framework, suggesting that its efficient iterative mechanism can serve as a capability amplifier: when the base model possesses stronger reasoning and generation ability, REAP can effectively guide and leverage these strengths without fine-tuning, thereby achieving further performance gains.

Case Study

To clearly demonstrate the workflow of the REAP framework, we present a case study that showcases the precise task decomposition by the Decomposer, the adaptive plan updates and action selection by the SP module, and the meticulous reasoning and verification performed by the FE module, all synergistically interacting to arrive at the final answer. A detailed example is provided in the Appendix.

Conclusion

To overcome the limitations of existing RAG methods in MHQA, where they often fall into local impasses or suffer from factual inaccuracies, we propose REAP, a novel framework that enhances reasoning reliability through iterative interaction. At the core of REAP are two synergistic modules: the Sub-task Planner (SP) and the Fact Extractor (FE). The SP maintains a global planning perspective to dynamically evaluate and optimize the reasoning trajectory, while the FE performs fine-grained analysis on retrieved content to extract high-fidelity facts. This decoupled yet tightly coordinated design ensures the coherence and accuracy of the reasoning process. Furthermore, by leveraging a multi-task fine-tuning paradigm and replacing sub-module with a more lightweight model, we improve performance while maintaining inference efficiency. Extensive experiments on multiple MHQA datasets demonstrate that REAP significantly outperforms existing state-of-the-art methods in both in-domain and out-of-domain settings, thereby validating its effectiveness and robustness in addressing complex multi-hop reasoning tasks.

References

Achiam, J.; Adler, S.; Agarwal, S.; Ahmad, L.; Akkaya, I.; Aleman, F. L.; Almeida, D.; Altenschmidt, J.; Altman, S.; Anadkat, S.; et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Asai, A.; Wu, Z.; Wang, Y.; Sil, A.; and Hajishirzi, H. 2023. Self-rag: Self-reflective retrieval augmented generation. In *NeurIPS 2023 workshop on instruction tuning and instruction following*.

Chan, C.-M.; Xu, C.; Yuan, R.; Luo, H.; Xue, W.; Guo, Y.; and Fu, J. 2024. Rq-rag: Learning to refine queries for retrieval augmented generation. *arXiv preprint arXiv:2404.00610*.

Cheng, X.; Li, J.; Zhao, W. X.; Zhang, H.; Zhang, F.; Zhang, D.; Gai, K.; and Wen, J.-R. 2024. Small agent can also rock! empowering small language models as hallucination detector. *arXiv preprint arXiv:2406.11277*.

Deng, B.; Wang, W.; Zhu, F.; Wang, Q.; and Feng, F. 2025. CrAM: Credibility-Aware Attention Modification in LLMs for Combating Misinformation in RAG. In Walsh, T.; Shah, J.; and Kolter, Z., eds., *AAAI-25, Sponsored by the Association for the Advancement of Artificial Intelligence, February 25 - March 4, 2025, Philadelphia, PA, USA*, 23760–23768. AAAI Press.

Dong, G.; Zhang, C.; Deng, M.; Zhu, Y.; Dou, Z.; and Wen, J.-R. 2024. Progressive multimodal reasoning via active retrieval. *arXiv preprint arXiv:2412.14835*.

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

Gao, L.; Ma, X.; Lin, J.; and Callan, J. 2023a. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1762–1777.

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

Gao, Y.; Xiong, Y.; Zhong, Y.; Bi, Y.; Xue, M.; and Wang, H. 2025. Synergizing rag and reasoning: A systematic review. *arXiv preprint arXiv:2504.15909*.

Glass, M.; Rossiello, G.; Chowdhury, M. F. M.; Naik, A. R.; Cai, P.; and Gliozzo, A. 2022. Re2G: Retrieve, rerank, generate. *arXiv preprint arXiv:2207.06300*.

Grattafiori, A.; Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Vaughan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

He, B.; Chen, N.; He, X.; Yan, L.; Wei, Z.; Luo, J.; and Ling, Z.-H. 2024. Retrieving, rethinking and revising: The chain-of-verification can improve retrieval augmented generation. *arXiv preprint arXiv:2410.05801*.

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

Huang, L.; Yu, W.; Ma, W.; Zhong, W.; Feng, Z.; Wang, H.; Chen, Q.; Peng, W.; Feng, X.; Qin, B.; et al. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2): 1–55.

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

Jiang, H.; Wu, Q.; Lin, C.-Y.; Yang, Y.; and Qiu, L. 2023a. Llmlingua: Compressing prompts for accelerated inference of large language models. *arXiv preprint arXiv:2310.05736*.

Jiang, Z.; Xu, F. F.; Gao, L.; Sun, Z.; Liu, Q.; Dwivedi-Yu, J.; Yang, Y.; Callan, J.; and Neubig, G. 2023b. Active retrieval augmented generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 7969–7992.

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

Jin, J.; Zhu, Y.; Dou, Z.; Dong, G.; Yang, X.; Zhang, C.; Zhao, T.; Yang, Z.; and Wen, J.-R. 2025b. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In *Companion Proceedings of the ACM on Web Conference 2025*, 737–740.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P. S.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP (1)*, 6769–6781.

Kim, J.; Nam, J.; Mo, S.; Park, J.; Lee, S.-W.; Seo, M.; Ha, J.-W.; and Shin, J. 2024. Sure: Summarizing retrievals using answer candidates for open-domain qa of llms. *arXiv preprint arXiv:2404.13081*.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474.

Li, X.; Xu, W.; Zhao, R.; Jiao, F.; Joty, S.; and Bing, L. 2024. Can we further elicit reasoning in llms? critic-guided planning with retrieval-augmentation for solving challenging tasks. *arXiv preprint arXiv:2410.01428*.

Ma, X.; Gong, Y.; He, P.; Zhao, H.; and Duan, N. 2023. Query rewriting in retrieval-augmented large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 5303–5315.

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

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

Robertson, S.; Zaragoza, H.; et al. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4): 333–389.

Roy, N.; Ribeiro, L. F.; Blloshmi, R.; and Small, K. 2024. Learning when to retrieve, what to rewrite, and how to respond in conversational QA. *arXiv preprint arXiv:2409.15515*.

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

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

Shi, Z.; Sun, W.; Gao, S.; Ren, P.; Chen, Z.; and Ren, Z. 2024. Generate-then-ground in retrieval-augmented generation for multi-hop question answering. *arXiv preprint arXiv:2406.14891*.

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

Su, J.; Healey, J.; Nakov, P.; and Cardie, C. 2025. Fast or better? balancing accuracy and cost in retrieval-augmented generation with flexible user control. *arXiv preprint arXiv:2502.12145*.

Su, W.; Tang, Y.; Ai, Q.; Wu, Z.; and Liu, Y. 2024. DRAGIN: dynamic retrieval augmented generation based on the information needs of large language models. *arXiv preprint arXiv:2403.10081*.

Tang, Y.; and Yang, Y. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *arXiv preprint arXiv:2401.15391*.

Teng, F.; Yu, Z.; Shi, Q.; Zhang, J.; Wu, C.; and Luo, Y. 2025. Atom of thoughts for markov llm test-time scaling. *arXiv preprint arXiv:2502.12018*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; Batra, S.; Bhargava, P.; Bhosale, S.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

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

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

Wang, L.; Yang, N.; Huang, X.; Jiao, B.; Yang, L.; Jiang, D.; Majumder, R.; and Wei, F. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Wang, X.; He, J.; Chen, L.; Yang, R. H. Z.; Wang, Y.; Meng, X.; Pan, K.; and Sui, Z. 2024a. SG-FSM: A Self-Guiding Zero-Shot Prompting Paradigm for Multi-Hop Question Answering Based on Finite State Machine. *arXiv preprint arXiv:2410.17021*.

Wang, Z.; Yuan, H.; Dong, W.; Cong, G.; and Li, F. 2024b. Corag: A cost-constrained retrieval optimization system for retrieval-augmented generation. *arXiv preprint arXiv:2411.00744*.

Xu, S.; Pang, L.; Shen, H.; Cheng, X.; and Chua, T.-S. 2024. Search-in-the-chain: Interactively enhancing large language models with search for knowledge-intensive tasks. In *Proceedings of the ACM Web Conference 2024*, 1362–1373.

Yang, D.; Rao, J.; Chen, K.; Guo, X.; Zhang, Y.; Yang, J.; and Zhang, Y. 2024a. Im-rag: Multi-round retrieval-augmented generation through learning inner monologues. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 730–740.

Yang, S.; Gribovskaya, E.; Kassner, N.; Geva, M.; and Riedel, S. 2024b. Do large language models latently perform multi-hop reasoning? *arXiv preprint arXiv:2402.16837*.

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

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Ye, L.; Lei, Z.; Yin, J.; Chen, Q.; Zhou, J.; and He, L. 2024. Boosting conversational question answering with fine-grained retrieval-augmentation and self-check. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2301–2305.

Ye, L.; Yu, L.; Lei, Z.; Chen, Q.; Zhou, J.; and He, L. 2025. Optimizing Question Semantic Space for Dynamic Retrieval-Augmented Multi-hop Question Answering. *arXiv preprint arXiv:2506.00491*.

Zhang, N.; Zhang, C.; Tan, Z.; Yang, X.; Deng, W.; and Wang, W. 2025. Credible plan-driven rag method for multi-hop question answering. *arXiv preprint arXiv:2504.16787*.

Zheng, Y.; Zhang, R.; Zhang, J.; Ye, Y.; Luo, Z.; Feng, Z.; and Ma, Y. 2024. LlamaFactory: Unified Efficient Fine-Tuning of 100+ Language Models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*. Bangkok, Thailand: Association for Computational Linguistics.

Zhu, Y.; Zhou, H.; Hong, W.; Liu, T.; and Wang, N. 2025. REAP: Enhancing RAG with Recursive Evaluation and Adaptive Planning for Multi-Hop Question Answering. *arXiv preprint arXiv:2511.09966*.