HierSearch: A Hierarchical Enterprise Deep Search Framework Integrating Local and Web Searches

Jiejun Tan[1], Zhicheng Dou[1]*, Yan Yu[2], Jiehan Cheng[1], Lifeng Liu[2], Jian Xie[2], Ji-Rong Wen[1]

[1]Gaoling School of Artificial Intelligence, Renmin University of China
[2]Baichuan Intelligent Technology
{zstanjj, dou, jrwen}@ruc.edu.cn

Abstract

Recently, large reasoning models have demonstrated strong mathematical and coding abilities, and deep search leverages their reasoning capabilities in challenging information retrieval tasks. Existing deep search works are generally limited to a single knowledge source, either local or the Web. However, enterprises often require private deep search systems that can leverage search tools over both local and the Web corpus. Simply training an agent equipped with multiple search tools using flat reinforcement learning (RL) is a straightforward idea, but it has problems such as low training data efficiency and poor mastery of complex tools. To address the above issue, we propose a hierarchical agentic deep search framework, HierSearch, trained with hierarchical RL. At the low level, a local deep search agent and a Web deep search agent are trained to retrieve evidence from their corresponding domains. At the high level, a planner agent coordinates low-level agents and provides the final answer. Moreover, to prevent direct answer copying and error propagation, we design a knowledge refiner that filters out hallucinations and irrelevant evidence returned by low-level agents. Experiments show that HierSearch achieves better performance compared to flat RL, and outperforms various deep search and multi-source retrieval-augmented generation baselines in six benchmarks across general, finance, and medical domains.

Code — https://github.com/plageon/HierSearch
Extended version — https://arxiv.org/abs/2508.08088

Introduction

Recently, large reasoning models (LRMs) such as DeepSeek-R1 (DeepSeek-AI 2025) and OpenAI's O-series (Openai 2025) models have shown impressive capabilities in mathematics and coding. However, LRMs are troubled by higher hallucination rates (Sun et al. 2025) and restricted by limited internal knowledge in knowledge-intensive tasks. Thus, studies have combined LRMs with retrieval-augmented generation (RAG) to enable models to obtain external knowledge assistance, which is referred to as deep search (Li et al. 2025a,b).

Existing deep search works often equip LRMs with a local corpus search tool (Chen et al. 2025; DeepSeek-AI 2025; Song et al. 2025) or a Web search tool (Li et al. 2025a,b; Zheng et al. 2025). However, a common scenario for most enterprises is that their private deep search system interacts with both local knowledge sources and Web knowledge sources (Yu et al. 2025). To be specific, enterprises often possess private domain-specific documents. Existing methods for building private RAG systems usually involve processing them into a text chunk corpus and constructing knowledge graphs (Edge et al. 2024; Guo et al. 2024). Web knowledge sources generally include search engines and web pages. Generally speaking, local knowledge sources are more professional and targeted. Meanwhile, Web knowledge sources are more comprehensive and timely (Zhao et al. 2024b; Wang et al. 2024a). This deep search scenario with multiple knowledge sources poses challenges to existing methods: *Deep search agents need to selectively use different knowledge sources based on user questions and the characteristics of knowledge sources, and cross-supplement missing knowledge.*

A straightforward solution for the above challenge is equipping the deep search agent with all search tools for all knowledge sources and conducting flat reinforcement learning (RL). However, the flat RL solution is not suitable for the following reasons: (1) Numerous search tools result in a large action space during RL, leading to low training efficiency and instability. (2) Search tools within the same knowledge source have stronger synergy (e.g., browsing a Web page via a URL retrieved by a search engine or retrieving text chunks mentioning an entity from the knowledge graph), while that between tools across different knowledge sources is weaker. However, flat RL fails to effectively utilize this characteristic. (3) Moreover, preliminary experiments show that during flat RL, rewards encourage the agent to search more frequently in easily retrievable knowledge sources, while less frequently in hard ones (Web search is more difficult in our setting due to a wider search scope and more noise). Thus, the training efficiency of flat RL for the difficult knowledge source is poor due to limited exploration of the corresponding tools.

To address the above issues, we propose a hierarchical agentic deep search paradigm, HierSearch, which comprises a local deep search agent, a Web deep search agent, and a planner agent. Two deep search agents interact directly with search tools within their knowledge sources and re-

*Corresponding author.

trieve evidence for the planner agent. Specifically, the local deep search agent has access to the local text chunk corpus and the local knowledge graph. The Web deep search agent has access to the Web search engine and online web pages. Meanwhile, the planner agent drafts search plans, coordinates search agents, analyzes evidence provided by search agents, and provides the final answer.

Accordingly, we leverage a hierarchical reinforcement learning (HRL) (Pateria et al. 2022) algorithm to train this hierarchical agentic framework. Also, we use Group Relative Policy Optimization (GRPO) (Shao et al. 2024) and rule-based rewards. HRL overcomes the challenges above, mainly manifested in: (1) **In the first stage, we train low-level agents, the local deep search agent, and the Web deep search agent separately.** They master search tools within the same domain well, because the number of tools is limited and the tools are closely related. (2) **In the second stage, we train the high-level planner agent, equipped with both deep search agents.** Well-trained deep search agents mask the complex interaction process with search tools, and greatly lower the difficulty of knowledge acquisition. The planner agent can learn search planning and knowledge integration across multiple knowledge sources faster and better.

In the planner agent's training stage, we find that directly providing the complete trajectories of deep search agents would introduce irrelevant search results and the agents' hallucinatory reasoning contents. To address this, we design a reasoning-aware knowledge refiner. This refiner first selects the evidence that contributes to each round of reasoning by the deep search agent. Second, it selects the evidence helpful to the agent's conclusion from an overall perspective.

We conduct extensive experiments on six benchmarks from the general domain, the medical domain, and the financial domain. The results show that HierSearch outperforms baselines and the flat RL solution across all benchmarks.

In summary, our contributions are threefold: (1) We explore the deep search framework in multi-knowledge-source scenarios and propose a hierarchical agentic paradigm and train with HRL; (2) We notice drawbacks of the naive information transmission among deep search agents and developed a knowledge refiner suitable for multi-knowledge-source scenarios; (3) Our proposed approach for reliable and effective deep search across multiple knowledge sources outperforms existing baselines the flat-RL solution in various domains.

Related Work

Deep Search Traditional RAG combines large language models (LLMs) with information retrieval to provide external knowledge and mitigate hallucination (Tan et al. 2024). Traditional RAG methods generally follow a fixed retrieve-then-generate pipeline (Jin et al. 2025), while several works explore flexible agentic pipelines (Asai et al. 2024; Yao et al. 2023). Compared to traditional RAG, deep search combines LRM with search tools (Li et al. 2025c). Equipped with stronger reasoning abilities, deep search pushes iterative RAG further, and analyzes deeper for users' questions (Li et al. 2025b), which can "*search, read and reason until best answer found*". Several organizations have developed their representative products, such as Google, OpenAI, and Jina. Meanwhile, several researchers builds deep search on open-source LRMs (DeepSeek-AI 2025), like RAG-Star (Jiang et al. 2025), Search-o1 (Li et al. 2025a), and WebThinker (Li et al. 2025b). These works often have issues of excessive reasoning and inaccurate searching in search tasks, but they have the advantage of greater flexibility in choosing models and search tools. To make reasoning models perform better in deep search tasks, another branch of work trains LLMs to conduct deep search tasks following the RL paradigm introduced by DeepSeek-R1 (DeepSeek-AI 2025), like DeepResearcher (Zheng et al. 2025), R1-Searcher (Song et al. 2025), and ReCall (Chen et al. 2025). The aforementioned deep search works are all limited to a single knowledge source, and at most two search tools (Li et al. 2025b). However, enterprise private deep search often needs to access local and Web knowledge sources as well as multiple search tools. Existing methods cannot supplement knowledge and handle knowledge conflicts across multiple knowledge sources. Moreover, they produce a lot of unnecessary search tool calls, especially expensive Web search tool calls. In contrast, HierSearch uses multiple deep search agents to tackle different knowledge sources, and a planner agent that selectively calls agents of different knowledge sources as needed and integrates knowledge from these sources.

Multi-Knowledge Source RAG In traditional RAG research, some works have identified the challenges RAG faces in multi-knowledge-source scenarios and proposed solutions. PruningRAG (Yu et al. 2025) uses multi-granularity pruning strategies to integrate information from documents of different sources and mitigate the impact of misleading information. PrefRAG (Zhao et al. 2024b) introduces preference-driven adaptive retrieval to handle multi-retrieval source data, and calls web retrieval as a supplement when local retrieval does not satisfy knowledge requirements. HM-RAG (Liu et al. 2025) applies multi-source agents to conduct retrieval in parallel, and uses consistency voting to integrate multi-source answers. These works are still static RAG paradigms that need to follow a predefined pipeline. They use prompting or DPO methods to enable agents to learn multi-source RAG tasks. In contrast, we apply the GRPO RL algorithm to develop the agent's deep thinking and searching capabilities. Agents with deep thinking capabilities are not limited to a specific search path; instead, they can independently plan when to call search tools, when to interact with other agents, and when to terminate.

Hierarchical RL HRL decomposes complex tasks into simpler subtasks and uses a high-level policy to select subtasks and a low-level policy to perform specific actions (Vezhnevets et al. 2017; Dayan and Hinton 1992). HRL is effective and data-efficient when used in tasks with multiple turns, long horizons, and delayed rewards (Pateria et al. 2022; Hutsebaut-Buysse, Mets, and Latré 2022). HRL has performed well in robot control and game AI (Nachum et al. 2018; Kulkarni et al. 2016; Zhang, Yu, and Xu 2021). Recent works have also applied HRL to agents built on LLMs (Zhou et al. 2024; Zhao et al. 2024a). To the best of our knowl-

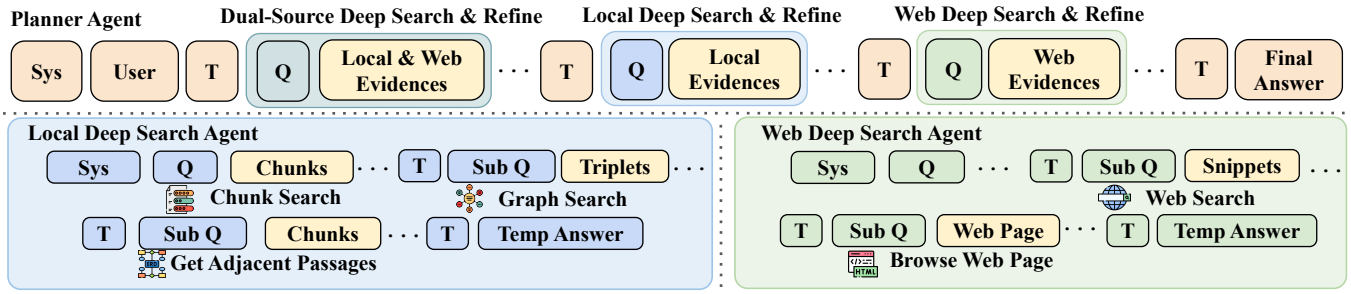

Figure 1: Illustration of the hierarchical agentic framework for HierSearch. We show exemplary trajectories of all low-level and high-level agents.

edge, this work is the first to use HRL in the deep search field. Multi-knowledge-source RAG task is broken down into two levels: in-knowledge-source deep search and cross-knowledge-source planning.

Methodology

We present HierSearch, a hierarchical agentic framework designed for enterprise-wide deep search across multiple knowledge sources. The framework comprises two levels: 1) low-level agents, including local and Web deep search agents, and 2) a high-level planner agent.

Problem Formulation

In a deep search task, the agent takes a user's question x, iteratively performs thinking processes or search tool calls, and finally outputs an answer $\hat{y}$. The optimization goal is to make the final answer as correct and helpful as possible. In the enterprise scenario, a deep search needs to access multiple knowledge sources before providing an answer. Given knowledge sources including a local text chunk corpus C, a local knowledge graph G, a Web search engine E, and accessible Web pages on the Internet P, the deep search framework is meant to maximize the probability of the golden answer y.

Hierarchical Agentic Deep Search

A straightforward idea for the multi-knowledge-source deep search task is equipping an agent with all search tools and conducting flat RL. However, our preliminary experiment shows that the flat RL displays drawbacks such as poor mastery of difficult Web search tools and low training data efficiency. Thus, we propose a hierarchical agentic deep search framework, HierSearch. As shown in Figure 1, HierSearch consists of a local deep search agent, a Web deep search agent, and a planner agent. We will discuss all three agents in the following sections in detail, including their accessible tools and their roles.

Preliminary: Tool-Augmented Reasoning We follow a commonly used synergized tool-augmented reasoning paradigm of current deep search methods (Li et al. 2025c). Our deep search agents and the planner agent roll out similarly. We use the following wrapping tags to distinguish different part in the trajectory: (1) The thinking processes are wrapped in `<think>...</think>`; (2) Tool calls are wrapped in `<tool_name>...</tool_name>` (The tool name varies). (3) Returned contents tool functions are wrapped in `<result>...</result>`. (4) The answer is wrapped in `<answer>...</answer>`. All tools accessible are demonstrated in the system prompt. The generation process pauses when the ending tags of tool calls are generated, and restarts until the tool call result is appended to the end of the sequence. The whole generation process ends when `</answer>` is generated or the number of tool call rounds reaches an upper limit.

Local Deep Search Agent The local deep search agent has access to two local knowledge sources: the text chunk corpus and the knowledge graph. The local agent accesses the text chunk corpus through `<chunk_search>` to retrieve chunks related to the input query. The local agent accesses knowledge graph by two tools: (1) `<graph_search>` retrieves triples (consisting of a subject, a predicate, and an object) related to the input query by calculating the similarity of semantic embeddings; (2) `<get_adjacent_passages>` returns relevant text chunks mentioning the input entity in the knowledge graph. The linking edges between graph entities and relevant chunks are identified and saved during the knowledge graph construction process.

Web Deep Search Agent The Web deep search agent accesses Web knowledge through two tools: (1) `<web_search>` calls a search engine API to retrieve web links and corresponding titles and snippets related to the input query; (2) `<browse_url>` takes both a web link and a query as inputs. We chunk the original HTML pages and only return query-relevant pieces, because the original HTML pages are generally lengthy and hard to read.

Multi-Knowledge Source Planner Agent Both the local deep search agent and the Web deep search agent are low-level agents that are manipulated by a high-level planner agent. The planner agent drafts search plans, integrates returned evidence from low-level agents, and provides the final answer. Low-level agents are packaged as tools for high-level agents to call, which includes the following: (1) `<local_search_agent>` calls the local deep search agent; (2) `<web_search_agent>` calls the Web deep

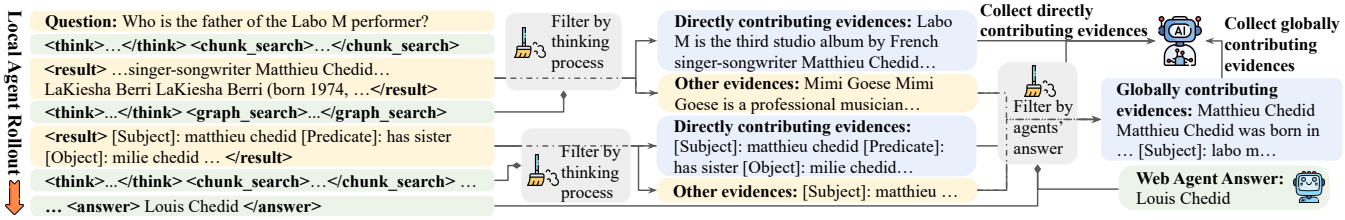

Figure 2: Illustration of the knowledge refining process from the local agent trajectory. The first step filters directly contributing evidence according to the subsequent thinking process of each round. The second step filters globally contributing evidence according to the local agent's answer and the web agent's answer (if available).

search agent; (3) <all_search_agent> calls both low-level agents simultaneously.

Hierarchical RL for Multi-Source Deep Search Considering the hierarchical framework and taking inspiration from HRL works, we employ HRL for HierSearch. That is, we first train two low-level search agents, and then the high-level planner agent. To be specific, we randomly sample the training set from MuSiQue (Trivedi et al. 2022), OmniEval (Wang et al. 2024b), and BioASQ (Nentidis et al. 2024). We mix these samples as the training data for agents.

We follow the GRPO algorithm introduced by DeepSeek-R1 (DeepSeek-AI 2025), and we use rule-based rewards, which are designed as follows. Agent trajectories with incorrect formats are punished with a zero reward. If the format is correct, we calculate the F1 score between the predicted answer $\hat{y}$ and the golden answer y. If the F1 score is larger than zero, we take the F1 score as the reward. If the rollout has a correct format but a zero F1 score, we encourage the agent to explore more tools. We calculate the proportion of the types of tools used during the rollout to the total types of tools accessible to the agent, and multiply it by a coefficient of 0.1 to serve as the reward. To sum up, the reward function can be formulated as:

$$R = \begin{cases} 0, & \text{if the format is incorrect,} \\ 0.1 \times t/T, & \text{if F1 = 0 and format is correct,} \\ \text{F1}(\hat{y}, y), & \text{if F1 } > 0 \text{ and format is correct.} \end{cases} \quad (1)$$

, where t is the number of tools used in the trajectory and T is the number of all tools accessible.

Reasoning-Aware Knowledge Refiner

This hierarchical framework requires information exchange between low-level deep search agents and the high-level planner agent. A straightforward idea is that low-level agents return the whole trajectory containing collected evidence (search results from search tools), thinking processes, and conclusions (temporary answers in answer tags). However, analytical experiments show that inputting all those information indiscriminately will be harmful for the planner, which mainly shows in: (1) Thinking processes and conclusions from low-level agents induce the planner agent directly copy them instead of thinking by itself; (2) Irrelevant evidence makes the contents low-level agents' returned lengthy and hard to read and confuses the planner agent; (3) The hallucinations generated by low-level agents lead to an error propagation to the planner agent.

Therefore, we design a knowledge refiner that filters key evidence contributing to the low-level agents' thinking processes and conclusions, as shown in Figure 2. The refiner filters evidence helpful for the thinking process in two steps. In the first refining step, we select evidence directly contributing to the next thinking process. Given a trajectory sequence S, which contains an input question x, and K rounds where thinking and tool calls alternate, and ends with a last thinking process t_{K+1} followed by a conclusion $\hat{c}$. The round k contains a thinking process t_k, a query q_k, and N returned evidence $\{e_{N(k-1)+1} \cdots e_{Nk}\}$. The trajectory sequence is like:

$$S = \{x, t_1, \cdots, t_k, q_k, e_{N(k-1)+1} \cdots e_{Nk}, \cdots, t_{K+1}, \hat{c}\} \quad (2)$$

The contribution score for each evidence in round k is given by its contribution to the next thinking process:

$$\text{Score}(e_i) = P(e_i | t_{k+1}),\ N(k-1)+1 \le i \le Nk \quad (3)$$

The contribution score is calculated by the embedding similarity score P between the evidence and the subsequent thinking process. In the first step, in each think & search round, top $\alpha\%$ evidence is selected.

In the second refining step, we distinguish evidence not selected in the first step but contributing globally to the agent's conclusion. As preparation, unselected evidence after the first step is gathered as candidates. If the planner agent calls only one low-level agent, we consider only that low-level agent's conclusion $\hat{c}$. If the planner agent calls both low-level agents, we concatenate $\hat{c}$ with the other agent's conclusion $\hat{c}^{'}$ as $\{\hat{c}, \hat{c}^{'}\}$, and consider them as a whole. The global contribution score for the conclusion is given by:

$$\text{Score}(e_i) = \begin{cases} P(e_i | \{\hat{c}, \hat{c}^{'}\}), & \hat{c}^{'} \text{exists,} \\ P(e_i | \hat{c}), & \text{otherwise.} \end{cases} \quad (4)$$

In the second step, the top $\beta\%$ of the remaining evidence is selected. Both α and β are hyperparameters of the refiner. Finally, evidence selected from the two steps is merged and tagged with its knowledge source. The planner receives a list of refined evidence collected by agents (e.g., "<result> Local Knowledge Graph: [Subject] matthieu chedid ... Search Engine: Labo M (2003) is the third studio album ... </result>").

Method	MuSiQue		OmniEval		BioASQ		NQ		HotpotQA		PubmedQA		# Searches	
	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	Local	Web
Local Search														
DeepSeek-R1	26.00	36.45	0.80	29.50	6.18	24.10	28.50	44.88	29.75	45.36	41.00	51.35	2.00	-
HippoRAG	30.25	43.36	0.00	29.27	9.71	36.87	43.25	59.71	35.25	52.23	68.50	70.95	2.00	-
R1-Searcher	44.50	55.86	2.93	9.85	34.12	50.87	44.50	56.92	48.25	63.88	64.00	64.12	1.68	-
ReCall	42.75	53.82	8.53	23.01	24.71	43.30	47.50	61.09	49.50	63.99	28.00	34.64	2.55	-
Web Search														
DeepSeek-R1	22.50	32.60	0.53	24.23	5.29	20.25	26.75	39.89	26.50	40.31	15.25	30.22	-	1.00
DeepResearcher	30.00	39.44	2.40	17.95	28.82	46.80	41.50	54.99	39.50	52.95	56.25	56.79	-	2.84
Search-o1	28.50	39.03	3.20	15.37	30.59	47.24	36.00	48.79	42.00	53.80	64.00	67.19	-	1.72
WebThinker	30.75	42.15	1.33	15.90	33.24	49.82	36.75	50.52	43.50	58.68	65.00	66.07	-	2.55
Parallel Search														
DeepSeek-R1	26.50	37.47	1.07	28.31	4.41	22.34	23.75	39.51	28.50	44.37	40.25	50.13	2.00	1.00
HippoRAG	33.25	46.39	0.00	29.69	10.29	37.29	43.00	59.88	39.75	57.70	70.25	70.93	2.00	1.00
HM-RAG	26.25	37.59	7.73	35.93	13.53	39.01	43.75	59.76	44.00	59.50	71.25	71.29	5.27	2.64
R1-Searcher	46.50	57.19	2.67	9.22	33.82	50.54	44.75	56.97	47.75	62.93	66.25	66.52	3.36	1.68
ReCall	43.00	52.69	9.33	22.02	26.18	42.45	48.25	61.13	47.00	62.12	31.75	39.34	4.62	2.31
DeepResearcher	33.75	44.94	6.40	24.96	32.94	52.44	46.25	59.76	45.75	60.23	64.75	65.50	4.20	2.10
Search-o1	36.25	47.53	5.60	18.82	32.06	50.18	39.25	53.36	44.00	59.13	65.50	68.93	3.10	1.55
WebThinker	33.00	44.53	5.60	19.87	33.82	50.54	40.25	53.39	46.75	61.04	67.75	69.04	4.38	2.19
Selective Search														
CRAG	26.50	36.89	1.07	28.50	5.88	23.76	25.00	42.25	30.00	45.43	41.50	51.76	2.00	0.61
PrefRAG	33.75	47.47	9.60	40.19	11.18	38.47	40.00	57.01	43.50	61.56	60.25	65.29	2.18	0.04
HierSearch$_{\text{w/o HRL}}$	46.00	56.34	7.73	39.49	39.41	62.42	47.75	59.65	42.00	57.99	67.50	69.31	4.82	1.02
HierSearch	**53.00**	**62.83**	**10.67**	**46.37**	**49.94**	**66.99**	**57.00**	**68.00**	**53.25**	**67.40**	**71.75**	**72.81**	5.74	1.06

Table 1: Main Results of HierSearch. The best and second best of each model are in **bold** and underlined.

Experiments

Benchmarks

We select three general-domain benchmarks, including: (1) MuSiQue (Trivedi et al. 2022): A synthetic multi-hop QA dataset; (2) Natural Questions (NQ) (Kwiatkowski et al. 2019): Real search engine questions collected by Google; (3) HotpotQA (Yang et al. 2018): A multi-hop QA dataset based on Wikipedia. We select one financial-domain benchmark, OmniEval (Wang et al. 2024b), a Chinese large-scale RAG benchmark targeting the financial domain with human annotations. We select two medical-domain benchmarks: (1) BioASQ (Nentidis et al. 2024): An annually updated biomedicine challenge with QA tasks; (2) PubMedQA (Jin et al. 2019): A human-annotated QA dataset based on research papers on PubMed. All benchmarks in finance and biomedicine include numerous queries that can only be answered using local knowledge. We randomly sample 373 samples for OmniEval, 340 samples for BioASQ, and 400 samples for other benchmarks from their corresponding test set (if available). We calculate Exact Match (EM) and F1 score as evaluation metrics for all benchmarks. Also, we count the average local search and Web search times (Web page browsing not included) required to process a query for each method.

Baselines

To demonstrate the effectiveness of our method, we select the following baselines:

• **Local Search**. (1) HippoRAG(Gutierrez et al. 2024): The graph RAG backbone method, with GPT-4o-mini (Openai 2024) as the base model. (2) DeepSeek-R1 (DeepSeek-AI 2025): A powerful reasoning model augmented by single-time chunk search and graph search; (3) R1-Searcher (Song et al. 2025) and (4) Recall (Chen et al. 2025): Both are deep search agents trained from scratch on QA datasets in local retrieval environments.

• **Web Search**. (1) A powerful reasoning model augmented by single-time Web search; (2) Search-o1 (Li et al. 2025a): A deep search method that incorporates Web search into reasoning in a single inference chain; (3) WebThinker (Li et al. 2025b): A deep search method that involves a deep web explorer in a main reasoning chain; (4) DeepResearcher (Zheng et al. 2025): A deep search agent trained from scratch in real-world web environments.

• **Parallel Search**. To align the knowledge sources and make a fair comparison, we reproduce the above baselines in a parallel search setting, where the same query is sent to both local and Web search tools in parallel. Also, we reproduce HM-RAG (Liu et al. 2025), which conducts parallel RAG based on text search, graph search, and Web search, and merges three answers with a majority vote.

• **Selective Search**. The agent autonomously decide which knowledge source to use or both, including: (1) PrefRAG (Zhao et al. 2024b): A multi-turn RAG pipeline that decides wether to involve Web search basing on local retrieval results; (2) CRAG (Yan et al. 2024): A plug-in discriminator that decides using Web search, local search or

Method	MuSiQue	OmniEval	BioASQ	NQ	HotpotQA	PubmedQA
HierSearch	**53.00**	**10.67**	**49.94**	**57.00**	**53.25**	**71.75**
w/o Local Agent	29.75 (23.25%↓)	3.20 (7.47%↓)	35.00 (14.94%↓)	36.00 (21.00%↓)	33.25 (20.00%↓)	65.00 (6.75%↓)
w/o Web Agent	47.50 (5.50%↓)	9.87 (0.80%↓)	46.18 (3.76%↓)	55.50 (1.50%↓)	51.50 (1.75%↓)	69.50 (2.25%↓)
w/o Refiner	50.75 (2.25%↓)	9.60 (1.07%↓)	48.82 (1.12%↓)	56.25 (0.75%↓)	48.50 (4.75%↓)	68.50 (3.25%↓)

Table 2: Ablation Study.

both basing on local retrieval results; (3) $\text{HierSearch}_{\text{w/o HRL}}$: A deep search agent equipped with all search tools and trained by flat RL.

Implementation Details

For local search, we prepare local knowledge bases separately for general, medical, and financial domains. For the general domain, we sample passages from the Wikipedia dump, and for the medical domain, we sample passages from the PubMed dump. The sampling passages consist of directly related passages for questions and hard negatives retrieved by BM25 (Robertson and Zaragoza 2009). This corpus sampling process for Wikipedia and PubMed is necessary because their original sizes are too large for constructing a graph upon them. For the financial domain, we use the original retrieval corpus of OmniEval. The knowledge graph is constructed upon the text chunk corpus. We follow HippoRAG (Gutierrez et al. 2024) and employ GPT-4o-mini (Openai 2024) and BGE-M3 (Chen et al. 2024) in graph construction. BGE-M3 is also the embedding model for all local search tools. As for the Web search, `<web_search>` uses Bing Search API for English queries and the Quark Search API for Chinese queries. `<browse_url>` accesses real-time Web pages and extract relevant evidence. For training settings, we collect training samples from Musique, OmniEval, and BioASQ. We train the local deep search agent, Web deep search agent, and the planner agent for 300 steps with a batch size of 64 and `Qwen2.5-7B-Instruct` (Yang et al. 2025) as the backbone. More implementation details are in the appendix.

Main Results

Main experimental results are shown in Table 1. "# Searches" is the average local or Web search tools (Web page browsing excluded) called to search for a user's question. Experiments demonstrate that HierSearch outperforms baselines without many additional search tool calls. Additionally, we make the following observations: (1) Baseline methods generally perform better if they have access to more knowledge sources. Local search has a larger augmentation than Web search because they are more professional and targeted. (2) Compared to methods using parallel search to access multiple knowledge sources, our method exhibits stronger deep search capabilities in multi-knowledge-source environments. Also, parallel search methods generate more Web search tool calls, which are slow and expensive. (3) Compared to multi-knowledge-source RAG baselines using selective search, our method is not constrained by a fixed workflow in knowledge source selection and integration, and makes deeper search and thinking. As for the comparison to flat RL ($\text{HierSearch}_{\text{w/o HRL}}$), we make a detailed analysis in the section below "*Effectiveness of Hierarchical RL*". (4) NQ, HotPotQA, and PubmedQA are not included in our training data, so the performances on them demonstrate our method's generalization ability in out-of-scope scenarios.

Further Analysis

Ablation Study We conduct an ablation study on key modules of our method, as shown in Table 2. We make the following observations: (1) We ablate the local deep search agent. In practice, we return an empty result when the planner calls the local deep search agent. Due to the lack of local information, the ultimate performance decreases. (2) Similar to the local deep search agent, when the Web deep search agent is ablated, the ultimate performance decreases due to a lack of Web knowledge. (3) We ablate the knowledge refiner. In practice, we directly provide the planner with the complete content of agent trajectories. Since the trajectories contain irrelevant search results and hallucinations from low-level agents, the overall performance is affected.

Effectiveness of Hierarchical RL To demonstrate that HRL has an edge over flat RL under the multi-knowledge-sources environment, we conduct a training comparison experiment. We start from an identical backbone model, `Qwen2.5-7B-Instruct`, and train with identical training samples. The training batch size is set to 64, and the total number of training steps is 300. We evaluate the checkpoint every 10 steps during training with a validation set sampled from MuSiQue and OmniEval. The results for the first 200 steps are shown in Figure 3 (due to space limitations). The green curve represents HierSearch using HRL, while the orange curve represents the flat-RL-trained agent. Comparing the reward curves, we can see: (1) At the initial stage, both methods' performances grow rapidly due to learning the tool-calling format, and the performance of HierSearch grows faster than flat RL. (2) Both methods' performance enters a plateau on MuSiQue after 20 steps, which is the same on OmniEval after 10 steps. During this period (steps 20 to 300), both methods are improving their deep search abilities slowly with fluctuations, and HierSearch consistently performs better than flat RL.

Additionally, through case analysis (more details in the Appendix), we find that: (1) The strong performance of HierSearch benefits from the low-level agents' stronger deep search ability compared to original search tools, as well as the refiner's ability to refine key evidence. (2) Flat RL faces multiple search tools and a larger action space, resulting in low sample utilization efficiency. Further analysis shows that

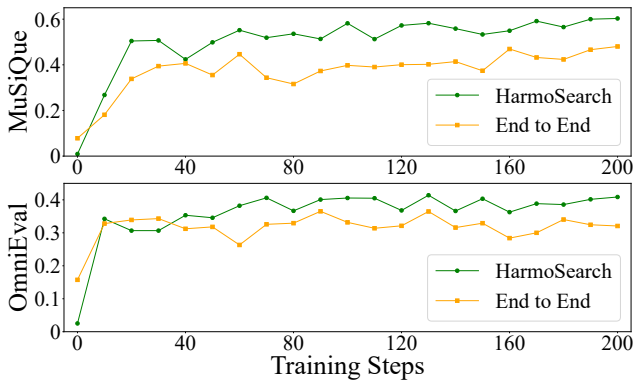

Figure 3: Rewards on Validation Sets during Training.

at step 290, the Web search tools only account for 18.5% of the total search tool calls, leading to low optimization efficiency.

Analysis of Multi-Knowledge Source Searching To further reveal the detailed reason that our method performs better in multi-knowledge-source environments, we analyze search success rates and reasoning success rates, and identify both of them according to different knowledge sources. To be specific, if the gold answer is contained in the returned local search results, it is regarded as a successful local search. This also applies to Web search, and "both" means both local and Web search are successful. The search success rate is calculated by dividing the number of search successful samples by the total number of samples. Meanwhile, under the premise of a successful local search, if the planner agent gives a correct final answer (EM = 1), it is regarded as successful reasoning. The reasoning success rate is calculated by dividing the number of reasoning successful samples by the number of search successful samples, which are the same for the Web and both.

Results are shown in Table 3, and we make the following observations: (1) Local search is easier than Web search, while Web search supplements some knowledge. (2) Specialized deep search agents have a higher search and reasoning success rate than agents built on general reasoning models. (3) Compared to deep search baselines, our method is better at searching as well as reasoning. (4) The flat RL solution ($\text{HarmoS}_{\text{w/o HRL}}$) outperforms all baselines in local search success rate and is close to our hierarchical method. However, its performance in web search success rate is unsatisfactory. This confirms our observation in preliminary experiments: the flat RL solution insufficiently explores and optimizes web search tools.

Efficiency Analysis Since we employ a hierarchical framework consisting of three agents, which may raise efficiency concerns, we make a comprehensive computational efficiency analysis, as shown in Table 4. We count the number of local search tool calls (# LS), Web search tool calls (# WS), Web page browsing tool calls (# WB), reasoning tokens (# Tokens), and the overall latency. For parallel search baselines, we call graph search, chunk search, and

Method	Search Success (%)			Reasoning Success (%)		
	Local	Web	Both	Local	Web	Both
R1-Searcher	84.75	51.00	47.75	49.85	58.82	59.16
ReCall	87.50	55.75	50.75	51.71	60.09	60.10
DeepResearcher	77.25	55.00	51.00	53.72	60.45	61.27
Search-o1	70.25	39.00	35.50	44.84	40.38	40.14
WebThinker	71.50	52.00	48.75	47.20	57.69	56.41
$\text{HierS}_{\text{w/o HRL}}$	89.75	23.50	22.25	51.81	62.77	61.80
HierSearch	**94.25**	**81.25**	**77.75**	**59.15**	**63.38**	**64.63**

Table 3: Multi-Knowledge-Source Utility Analysis on NQ.

Method	# LS	# WS	# WB	# Tokens	Latency (s)
Parallel Search					
R1-Searcher	4.26	2.13	-	297.62	8.84
ReCall	4.70	2.35	-	165.15	7.70
DeepResearcher	4.42	2.21	0.01	192.57	7.72
Search-o1	3.46	1.73	16.56	1,503.71	75.43
WebThinker	5.72	2.86	25.36	4,276.77	140.83
Selective Search					
CRAG	0.99	0.72	-	1,820.88	24.59
PrefRAG	2.08	0.05	-	1,077.02	13.63
$\text{HierS}_{\text{w/o HRL}}$	5.16	1.03	1.02	334.98	10.04
HierSearch	3.54	1.06	2.23	408.68	14.79

Table 4: Efficiency Analysis on MuSiQue.

Web search tools in parallel whenever the agent provides a query. The first two are local search tools, so their local search count is exactly twice that of Web search. For latency calculation, we estimate it with 43.99ms for a local search, 2.30s for a Web search, 3.16s for a Web page browsing, and 12.57ms for a reasoning token. In addition, we made the following observations: (1) Compared with the parallel search baselines, our method does not significantly increase search and reasoning cost. (2) Using the parallel search to integrate knowledge from different sources leads to unnecessary Web search tool calls, which are a lot more expensive and slower than local search tool calls. (3) Prompting reasoning models to build deep search agents significantly consumes more reasoning tokens, such as WebThinker and Search-o1. Such token consumption is of limited help for deep search tasks.

Conclusion

In this work, we propose a hierarchical agentic paradigm that integrates local and Web searches for enterprise deep search. Our method consists of a low-level local deep search agent and a Web deep search agent that conduct deep search in their corresponding knowledge sources, and a planner agent that coordinates low-level agents and provides the final answer. Furthermore, we devise a knowledge refiner that extracts helpful evidence from low-level agents' trajectories. Extensive experiments demonstrate that our method is effective and efficient across various domains, with better performance in searching and reasoning. This work explores the field of multi-knowledge-source deep search. We anticipate future research questions and research works in this field.

Acknowledgments

This work was supported by National Science and Technology Major Project No. 2022ZD0120103, National Natural Science Foundation of China No. 62272467. The work was partially done at the Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE.

References

Asai, A.; Wu, Z.; Wang, Y.; Sil, A.; and Hajishirzi, H. 2024. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net.

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. *CoRR*, abs/2402.03216.

Chen, M.; Li, T.; Sun, H.; Zhou, Y.; Zhu, C.; Wang, H.; Pan, J. Z.; Zhang, W.; Chen, H.; Yang, F.; Zhou, Z.; and Chen, W. 2025. ReSearch: Learning to Reason with Search for LLMs via Reinforcement Learning.

Dayan, P.; and Hinton, G. E. 1992. Feudal Reinforcement Learning. In Hanson, S. J.; Cowan, J. D.; and Giles, C. L., eds., *Advances in Neural Information Processing Systems 5, [NIPS Conference, Denver, Colorado, USA, November 30 - December 3, 1992]*, 271–278. Morgan Kaufmann.

DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *CoRR*, abs/2501.12948.

Edge, D.; Trinh, H.; Cheng, N.; Bradley, J.; Chao, A.; Mody, A.; Truitt, S.; and Larson, J. 2024. From Local to Global: A Graph RAG Approach to Query-Focused Summarization. *CoRR*, abs/2404.16130.

Guo, Z.; Xia, L.; Yu, Y.; Ao, T.; and Huang, C. 2024. LightRAG: Simple and Fast Retrieval-Augmented Generation. *CoRR*, abs/2410.05779.

Gutierrez, B. J.; Shu, Y.; Gu, Y.; Yasunaga, M.; and Su, Y. 2024. HippoRAG: Neurobiologically Inspired Long-Term Memory for Large Language Models. In Globersons, A.; Mackey, L.; Belgrave, D.; Fan, A.; Paquet, U.; Tomczak, J. M.; and Zhang, C., eds., *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*.

Hutsebaut-Buysse, M.; Mets, K.; and Latré, S. 2022. Hierarchical Reinforcement Learning: A Survey and Open Research Challenges. *Mach. Learn. Knowl. Extr.*, 4(1): 172–221.

Jiang, J.; Chen, J.; Li, J.; Ren, R.; Wang, S.; Zhao, X.; Song, Y.; and Zhang, T. 2025. RAG-Star: Enhancing Deliberative Reasoning with Retrieval Augmented Verification and Refinement. In Chiruzzo, L.; Ritter, A.; and Wang, L., eds., *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2025 - Volume 1: Long Papers, Albuquerque, New Mexico, USA, April 29 - May 4, 2025*, 7064–7074. Association for Computational Linguistics.

Jin, J.; Zhu, Y.; Dou, Z.; Dong, G.; Yang, X.; Zhang, C.; Zhao, T.; Yang, Z.; and Wen, J. 2025. FlashRAG: A Modular Toolkit for Efficient Retrieval-Augmented Generation Research. In Long, G.; Blumestein, M.; Chang, Y.; Lewin-Eytan, L.; Huang, Z. H.; and Yom-Tov, E., eds., *Companion Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025 - 2 May 2025*, 737–740. ACM.

Jin, Q.; Dhingra, B.; Liu, Z.; Cohen, W. W.; and Lu, X. 2019. PubMedQA: A Dataset for Biomedical Research Question Answering. In Inui, K.; Jiang, J.; Ng, V.; and Wan, X., eds., *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, 2567–2577. Association for Computational Linguistics.

Kulkarni, T. D.; Narasimhan, K.; Saeedi, A.; and Tenenbaum, J. 2016. Hierarchical Deep Reinforcement Learning: Integrating Temporal Abstraction and Intrinsic Motivation. In Lee, D. D.; Sugiyama, M.; von Luxburg, U.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, 3675–3683.

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

Li, X.; Dong, G.; Jin, J.; Zhang, Y.; Zhou, Y.; Zhu, Y.; Zhang, P.; and Dou, Z. 2025a. Search-o1: Agentic Search-Enhanced Large Reasoning Models. *CoRR*, abs/2501.05366.

Li, X.; Jin, J.; Dong, G.; Qian, H.; Zhu, Y.; Wu, Y.; Wen, J.; and Dou, Z. 2025b. WebThinker: Empowering Large Reasoning Models with Deep Research Capability. *CoRR*, abs/2504.21776.

Li, Y.; Zhang, W.; Yang, Y.; Huang, W.-C.; Wu, Y.; Luo, J.; Bei, Y.; Zou, H. P.; Luo, X.; Zhao, Y.; Chan, C.; Chen, Y.; Deng, Z.; Li, Y.; Zheng, H.-T.; Li, D.; Jiang, R.; Zhang, M.; Song, Y.; and Yu, P. S. 2025c. Towards Agentic RAG with Deep Reasoning: A Survey of RAG-Reasoning Systems in LLMs.

Liu, P.; Liu, X.; Yao, R.; Liu, J.; Meng, S.; Wang, D.; and Ma, J. 2025. HM-RAG: Hierarchical Multi-Agent Multimodal Retrieval Augmented Generation. *CoRR*, abs/2504.12330.

Nachum, O.; Gu, S.; Lee, H.; and Levine, S. 2018. Data-Efficient Hierarchical Reinforcement Learning. In Bengio, S.; Wallach, H. M.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, 3307–3317.

Nentidis, A.; Katsimpras, G.; Krithara, A.; Lima-López, S.; Farré-Maduell, E.; Krallinger, M.; Loukachevitch, N. V.; Davydova, V.; Tutubalina, E.; and Paliouras, G. 2024. Overview of BioASQ 2024: The Twelfth BioASQ Challenge on Large-Scale Biomedical Semantic Indexing and Question Answering. In Goeuriot, L.; Mulhem, P.; Quénot, G.; Schwab, D.; Nunzio, G. M. D.; Soulier, L.; Galuscáková, P.; de Herrera, A. G. S.; Faggioli, G.; and Ferro, N., eds., *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 15th International Conference of the CLEF Association, CLEF 2024, Grenoble, France, September 9-12, 2024, Proceedings, Part II*, volume 14959 of *Lecture Notes in Computer Science*, 3–27. Springer.

Openai. 2024. GPT-4o mini: advancing cost-efficient intelligence. Technical report, Openai.

Openai. 2025. OpenAI o3 and o4-mini System Card. Technical report, Openai.

Pateria, S.; Subagdja, B.; Tan, A.; and Quek, C. 2022. Hierarchical Reinforcement Learning: A Comprehensive Survey. *ACM Comput. Surv.*, 54(5): 109:1–109:35.

Robertson, S. E.; and Zaragoza, H. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Found. Trends Inf. Retr.*, 3(4): 333–389.

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

Song, H.; Jiang, J.; Min, Y.; Chen, J.; Chen, Z.; Zhao, W. X.; Fang, L.; and Wen, J. 2025. R1-Searcher: Incentivizing the Search Capability in LLMs via Reinforcement Learning. *CoRR*, abs/2503.05592.

Sun, Z.; Wang, Q.; Wang, H.; Zhang, X.; and Xu, J. 2025. Detection and Mitigation of Hallucination in Large Reasoning Models: A Mechanistic Perspective. *arXiv preprint arXiv:2505.12886*.

Tan, J.; Dou, Z.; Zhu, Y.; Guo, P.; Fang, K.; and Wen, J. 2024. Small Models, Big Insights: Leveraging Slim Proxy Models To Decide When and What to Retrieve for LLMs. In Ku, L.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, 4420–4436. Association for Computational Linguistics.

Trivedi, H.; Balasubramanian, N.; Khot, T.; and Sabharwal, A. 2022. MuSiQue: Multihop Questions via Single-hop Question Composition. *Trans. Assoc. Comput. Linguistics*, 10: 539–554.

Vezhnevets, A. S.; Osindero, S.; Schaul, T.; Heess, N.; Jaderberg, M.; Silver, D.; and Kavukcuoglu, K. 2017. FeUdal Networks for Hierarchical Reinforcement Learning. In Precup, D.; and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, 3540–3549. PMLR.

Wang, F.; Wan, X.; Sun, R.; Chen, J.; and Arik, S. Ö. 2024a. Astute RAG: Overcoming Imperfect Retrieval Augmentation and Knowledge Conflicts for Large Language Models. *CoRR*, abs/2410.07176.

Wang, S.; Tan, J.; Dou, Z.; and Wen, J. 2024b. OmniEval: An Omnidirectional and Automatic RAG Evaluation Benchmark in Financial Domain. *CoRR*, abs/2412.13018.

Yan, S.; Gu, J.; Zhu, Y.; and Ling, Z. 2024. Corrective Retrieval Augmented Generation. *CoRR*, abs/2401.15884.

Yang, A.; Yu, B.; Li, C.; Liu, D.; Huang, F.; Huang, H.; Jiang, J.; Tu, J.; Zhang, J.; Zhou, J.; Lin, J.; Dang, K.; Yang, K.; Yu, L.; Li, M.; Sun, M.; Zhu, Q.; Men, R.; He, T.; Xu, W.; Yin, W.; Yu, W.; Qiu, X.; Ren, X.; Yang, X.; Li, Y.; Xu, Z.; and Zhang, Z. 2025. Qwen2.5-1M Technical Report. *CoRR*, abs/2501.15383.

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 Riloff, E.; Chiang, D.; Hockenmaier, J.; and Tsujii, J., eds., *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, 2369–2380. Association for Computational Linguistics.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K. R.; and Cao, Y. 2023. ReAct: Synergizing Reasoning and Acting in Language Models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Yu, S.; Cheng, M.; Yang, J.; Ouyang, J.; Luo, Y.; Lei, C.; Liu, Q.; and Chen, E. 2025. Multi-Source Knowledge Pruning for Retrieval-Augmented Generation: A Benchmark and Empirical Study.

Zhang, J.; Yu, H.; and Xu, W. 2021. Hierarchical Reinforcement Learning by Discovering Intrinsic Options. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Zhao, Q.; Fu, H.; Sun, C.; and Konidaris, G. 2024a. EPO: Hierarchical LLM Agents with Environment Preference Optimization. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Miami, FL, USA, November 12-16, 2024*, 6401–6415. Association for Computational Linguistics.

Zhao, Q.; Wang, R.; Wang, X.; Zha, D.; and Mu, N. 2024b. Towards Multi-Source Retrieval-Augmented Generation via Synergizing Reasoning and Preference-Driven Retrieval. *CoRR*, abs/2411.00689.

Zheng, Y.; Fu, D.; Hu, X.; Cai, X.; Ye, L.; Lu, P.; and Liu, P. 2025. DeepResearcher: Scaling Deep Research via Reinforcement Learning in Real-world Environments. *CoRR*, abs/2504.03160.

Zhou, Y.; Zanette, A.; Pan, J.; Levine, S.; and Kumar, A. 2024. ArCHer: Training Language Model Agents via Hierarchical Multi-Turn RL. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net.