MULTISEARCH: DYNAMIC SEARCH TOOL ORCHESTRATION FOR INTELLIGENT AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advancements in Large Reasoning Models (LRMs) have enabled agents to perform information seeking involving complex reasoning and planning. Beyond deciding what to search for, agents must also determine which search sources best satisfy their evolving information needs. However, existing agents primarily rely on single general-purpose search engines. Furthermore, selecting suitable source from multiple similar options is far more challenging than distinguishing tools with vastly different capabilities. In this paper, we introduce `MultiSearch-Bench`, a comprehensive benchmark constructed from 319 search sources with overlapping functionality and diverse content coverage. Our analysis reveals a "High Recall, Low Quality" phenomenon: naive retrieval often achieves high recall by matching surface semantics yet fails to translate this into high-quality answers. To address this, we propose a reasoning-based pipeline `MultiSearch-R`. Our method treats tool selection as a dynamic planning task and employs a multi-granularity reward mechanism. Extensive experiments at 4B, 8B, and 14B model scales demonstrate that our approach significantly outperforms strong baselines across various metrics. Our work lays the foundation for handling massive similar tools, paving the way for precise source selection in open-ended, multi-source deep research environments. Our code is available at: https://anonymous.4open.science/r/MultiSearch.

1 INTRODUCTION

The evolution of Large Language Models (LLMs) towards Large Reasoning Models (LRMs) has fundamentally transformed the paradigm of information seeking. Recent commercial Deep Search systems, such as OpenAI Jaech et al. (2024) and Gemini Comanici et al. (2025), have moved beyond static RAG Lewis et al. (2020); Zhao et al. (2024); Asai et al. (2024) to autonomous agents capable of complex planning, multi-step execution, and self-correction Wei et al. (2022); Yao et al. (2022). By extending inference-time computation, these agents can theoretically tackle open-ended research tasks with unprecedented depth.

However, current agents typically rely on a single general-purpose search engine (*e.g.*, Google or Bing). While they can retrieve content from many websites, their search interfaces do not require agents to explicitly decide which sources to consult. For specialized or multi-source tasks, this decision matters. Sources covering the same topic may provide different evidence, serve different purposes, or offer complementary information. For instance, when analyzing specific market trends, a general engine might display popular or commercially driven content, whereas financial databases offer accurate raw data. Relying solely on a single general-purpose search engine can result in missing critical information stored in vertical databases or internal search systems.

When multiple search tools are available, selecting the right one becomes difficult. Previous studies Qin et al. (2023); Tang et al. (2023); Patil et al. (2024); Mialon et al. (2023); Schick et al. (2023); Shi et al. (2025); Wang et al. (2025); Qu et al. (2024) mainly focus on distinguishing between clearly distinct functions, for example, choosing a Calculator for math versus a Weather API for forecasts. In contrast, selecting the most suitable one from **multiple similar search tools** under different data sources is much harder. The model must understand the specific content coverage and rules of each source, rather than just its broad function. How to optimize agent behavior using various search tools remains an unexplored question.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

In order to bridge this gap, we introduce `MultiSearch-Bench`, a comprehensive benchmark designed to evaluate tool selection in multi-source search scenarios. Unlike prior datasets, `MultiSearch-Bench` is constructed from 319 search sources spanning 30 main categories. We employ a rigorous construction pipeline combining tree-based hierarchical exploration for single-source depth and scenario-driven synthesis for multi-source complexity, ensuring the dataset reflects realistic, high-noise search environments.

Obviously, navigating this massive, fine-grained tool space poses unique challenges. Our empirical analysis reveals a **"High Recall, Low Quality" phenomenon**: while traditional dense retrieval methods Xu et al. (2024); Chen et al. (2024); Molfetta et al. (2025) achieve high recall by identifying semantically relevant tools, they often fail in high-quality answers. This indicates that static embedding similarity is insufficient for agent workflows involving dynamic planning that require adherence to specific reasoning paths.

Therefore, we propose a novel reasoning-driven search pipeline `MultiSearch-R` to address this problem. Instead of selecting tools directly, our model first functions as a planner, generating abstract descriptions of ideal tools Fei et al. (2025). We also design a **multi-granularity reward system** that jointly optimizes tool selection, query formulation, and final answer generation. This method enables the model to adapt to the whole pipeline, yielding better results.

We conduct extensive experiments across model scales (4B, 8B, 14B), comparing our approach against internal baselines and SOTA agents like Search-o1 Li et al. (2025a), Search-R1 Jin et al. (2025); Song et al. (2025a), and WebThinker Li et al. (2025c). Our results demonstrate that, across all model scales, our RL fine-tuned model consistently outperform baselines in terms of answer quality. Beyond search, this work serves as a foundational exploration for future agents to leverage massive toolsets by identifying fine-grained content alignment of providers with similar functionalities.

In summary, our contributions are as follows:

(1) We construct `MultiSearch-Bench`, a large-scale benchmark dedicated to evaluating **similar tool selection** among hundreds of **search sources**.

(2) We propose a iterative reasoning-driven search pipeline `MultiSearch-R` with a **multi-granularity reward** mechanism. This method enables the model to balance active information seeking with execution efficiency.

(3) We provide an in-depth analysis of the **"High Recall, Low Quality"** phenomenon, offering new insights into why naive retrieval fails in agentic search and demonstrating the effectiveness of reasoning-driven planning.

2 Related Work

2.1 From RAG to Deep Search Agents

The paradigm of information retrieval has evolved from static RAG Lewis et al. (2020); Zhao et al. (2024) to autonomous agents. Early systems established the foundation by interleaving reasoning with iterative search actions Nakano et al. (2021); Yao et al. (2022); Trivedi et al. (2023); Xu et al. (2024); Chen et al. (2024); Asai et al. (2024). To scale these capabilities, various open-source frameworks have been proposed to facilitate robust web interaction Liu et al. (2023); Zhou et al. (2023b); Iong et al. (2024). Agents have further advanced to handle complex, multi-turn web navigation tasks Deng et al. (2024); Abuelsaad et al. (2024). In terms of multimodal models, some work leverages visual grounding to enable end-to-end exploration of realistic web environments Lai et al. (2024); He et al. (2025); Zheng et al. (2024); Furuta et al. (2023), while others explore API-based alternatives to streamline interaction Song et al. (2025b).

Current research emphasizes LRMs that prioritize inference-time compute. While proprietary systems lead in industry Jaech et al. (2024); Comanici et al. (2025); Team et al. (2025), the open-source community actively explores prompt-based reasoning Li et al. (2025a) and RL (*e.g.*, PPO Schulman et al. (2017) / GRPO Shao et al. (2024)) to optimize search trajectories Jin et al. (2025); Song et al. (2025a); Li et al. (2025c); Zheng et al. (2025). Furthermore, advanced planning mechanisms such as tree search and symbolic learning are being integrated to enable self-evolving search capabilities Koh et al. (2024); Ou et al. (2025); Chen et al. (2025); Sun et al. (2025); Li et al. (2025b).

Despite these advances in navigation and reasoning, existing agents primarily rely on a single, general-purpose search engine. Our research aims to provide a complementary approach by explicitly incorporating source selection into the search strategy. We investigate how agents can select from multiple sources with overlapping capabilities and coordinate queries across successive reasoning steps to access specialized, domain-specific knowledge.

2.2 Search Agent Benchmarks

Early benchmarks which evaluate search agents in dynamic web environments focused on text-based navigation and DOM parsing Xu et al. (2021); Humphreys et al. (2022); Kim et al. (2023); Zhou et al. (2023a); Tan et al. (2025), gradually shifting towards complex, long-horizon reasoning tasks Yoran et al. (2024); Tian et al. (2025). They have established rigorous standards for execution correctness and DOM-level interaction. Additionally, existing search-centric evaluations often assume a single general-purpose engine Nakano et al. (2021) or restrict tasks to navigation within pre-defined websites Deng et al. (2023). In contrast, `MultiSearch-Bench` focuses on the problem of selecting among multiple highly similar search tools with overlapping functionality. Distinguishing between these tools requires more fine-grained semantic understanding of their content coverage. Our work highlights the role of high-level planning in orchestrating information retrieval and evidence synthesis.

3 Construction of MultiSearch-Bench

In this section, we introduce `MultiSearch-Bench`, a novel benchmark designed to evaluate the ability of LRMs to select highly similar search tools and propose corresponding adaptive queries, including both single-source and multi-source retrieval scenarios. As illustrated in Figure 1, the construction process is structured into three phases: source repository curation, data collection, and task generation. More detailed comparison with existing benchmarks can be found in Appendix A.2.

3.1 Source Repository Curation

Considering comprehensiveness and universality of the search scope, we curated a diverse repository of 319 high-quality websites, spanning 30 main categories and 77 subcategories (Figure 3 for brief and Figure 4 for detail). To enable effective model interaction, we standardized each source into a search tool, defined by a tuple:

$$\mathcal{T} = \langle \textit{Category}, \textit{Subcategory}, \textit{Name}, \textit{Domain}, \textit{Description}, \textit{Root URL} \rangle. \tag{1}$$

Notably, to ensure objectivity and authority, the *Description* field was filled using Wikipedia Summary API,[1] which provides concise, entity-centric summaries for each platform. Due to the varying accessibility of web data, we adopted a hybrid implementation strategy to instantiate these tools.

Custom Search Wrappers For some representative platforms listed in Appendix A.3 with internal architectures, we developed custom wrappers. These wrappers simulate browser interactions to access native search interfaces, parsing HTML responses into unified structural formats.

Simulated Domain Search For the remaining sites, we leveraged the Google Search API with domain-restricted operators like "site:domain.com" to efficiently simulate site-specific search functionality.

3.2 Data Collection

We designed two distinct pipelines to generate search tasks with varying degrees of complexity, resulting in a total of 1,000 valid samples. Single and multi refer to the provenance of the reference evidence, rather than the number of tools available to the agent. In both settings, agents select sources from the repository. The former involves integrating one or more web pages restricted to a single domain, whereas the latter focuses on diverse information sources. The detailed structure of our benchmark and the results of statistical significance tests are presented in Appendix A.4.

[1] `https://en.wikipedia.org/api/rest_v1/page/summary/`

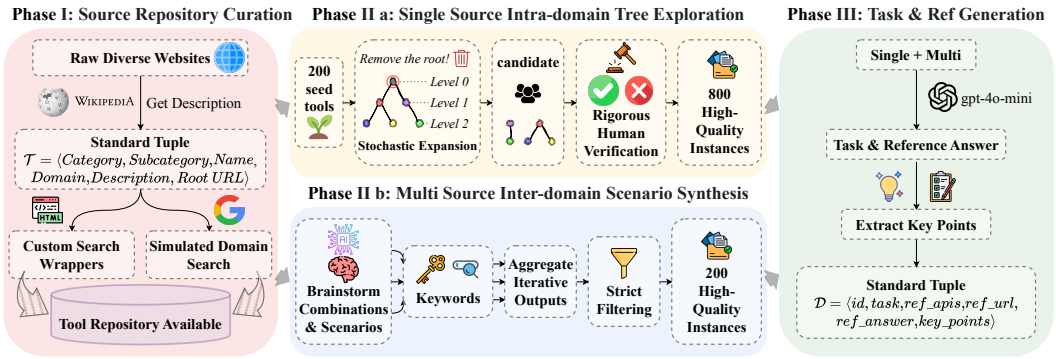

Figure 1: Overview of construction workflow of `MultiSearch-Bench`. The process comprises three phases: (a) Source Repository Curation, standardizing heterogeneous search tools using a hybrid strategy; (b) Data Collection, generating instances via Intra-domain Tree Exploration and Inter-domain Scenario Synthesis; (c) Task Generation, synthesizing user queries and reference answers with strict grounding constraints.

3.2.1 Single-Source Data

We introduce an Intra-domain Tree Exploration method to mine the depth and connectivity of pages within a specific domain. The process initializes with the homepage (Level 0) as the root and extracts valid internal URLs as children. Through stochastic expansion, we iteratively select URLs to extend either the depth (Level $n+1$) or breadth of the tree until a target set size is met. Crucially, we discard the homepage from the final collection, because it typically contains fragmented information, which prevents LLMs from focusing on key details. Starting with 200 distinct seed tools, we construct 5 instances for each, with evidence sets ranging in size from 1 to 5 URLs. Out of 1,000 resulting candidates, 800 high-quality instances are ultimately retained after rigorous manual verification.

3.2.2 Multi-Source Data

To simulate complex information needs that necessitate synthesis across diverse domains, we employ a scenario-driven strategy called Inter-domain Scenario Synthesis. Specifically, we prompt the LLM to brainstorm coherent search scenarios grounded in the names and functional descriptions of the collected tools. For each scenario, the model proposes a combination of 2–5 distinct tools required to address user's intent. Subsequently, we formulate keyword queries and execute Boolean-constrained searches like "query site:A OR site:B" to retrieve top-k relevant URLs. Through strict filtering for uniqueness and logical feasibility, it yielded 200 verified multi-source instances, each representing a distinct cross-domain search task.

3.3 Task and Reference Generation

After collecting the evidence sets, we leverage the LLM to synthesize naturalistic user queries and precise reference answers based on the extracted textual content. A critical constraint of this phase is strict grounding. The model is instructed to derive answers exclusively from the provided URL content. Such a design guarantees that every task is fully answerable within the retrieved context, thereby minimizing reliance on the model's internal parametric knowledge and ensuring the validity of the evaluation benchmarks.

4 Methodology

4.1 Iterative Multi-Source Search Pipeline

To empower LRMs to navigate complex multi-source information seeking, we propose an end-to-end iterative search framework called `MultiSearch-R`. As shown in Figure 2, it operates through four

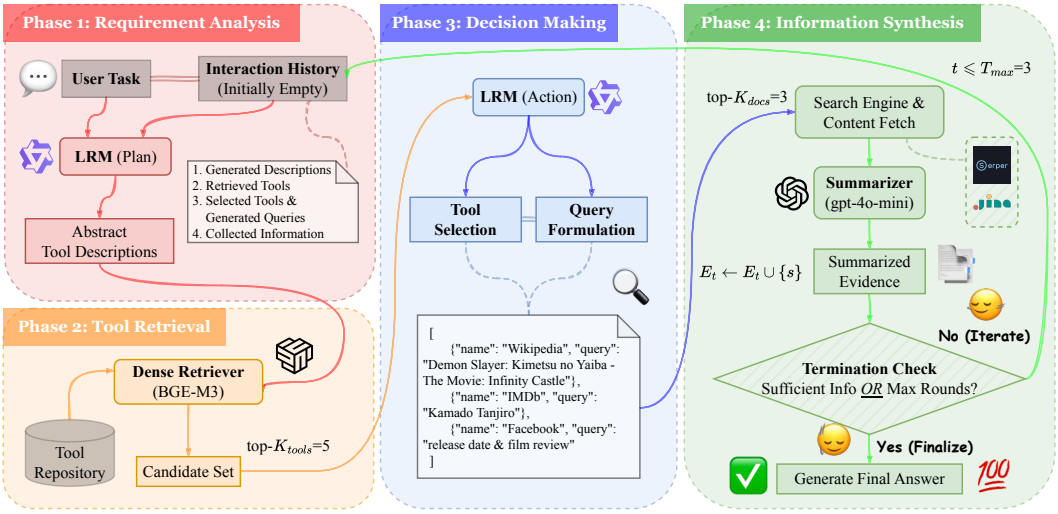

Figure 2: Overview of the `MultiSearch-R` framework. It operates iteratively through four phases: (1) **Requirement Analysis**, planning to generate abstract tool descriptions; (2) **Tool Retrieval**, mapping these descriptions to concrete candidates; (3) **Decision Making**, performing tool selection and query formulation simultaneously; (4) **Information Synthesis**, executing searches and summarizing retrieved content for reasoning and final answer generation.

distinct phases: Requirement Analysis, Tool Retrieval, Decision Making, and Information Synthesis. By explicitly separating reasoning from execution, abstract tool descriptions serve as an intermediate representation that connects the model's information needs with the available search sources.

4.1.1 Requirement Analysis

Given a user task, the LRM first functions as a strategic planner. Rather than selecting tools directly from a massive repository, the model analyzes the task requirements to generate a list of *Abstract Tool Descriptions*. Specifically, the model hypothesizes the ideal tools required to resolve the problem, ensuring that the generated descriptions are comprehensive and functionally non-overlapping.

4.1.2 Tool Retrieval

We leverage a dense retrieval mechanism to map the model's abstract intent to concrete tools. Utilizing `BGE-M3`, we encode both generated descriptions and repository's tool profiles into high-dimensional semantic vectors. Subsequently, we compute the cosine similarity between the query and candidate vectors. To manage the context window efficiently, we strictly limit the retrieved candidate pool size to top-$K_{\text{tool}} = 5$. These top-ranked instances constitute the *Candidate Set* for the next step.

4.1.3 Decision Making

Upon receiving the *Candidate Set* and original task, the LRM concurrently executes two sub-tasks: (1) **Tool Selection**, where it identifies the most appropriate tools from the candidate pool. (2) **Query Formulation**, where it crafts specific search queries tailored to the selected tools. This dual-process design tests the model's capability to ground its abstract needs into available resources.

4.1.4 Information Synthesis

We orchestrate the search environment using tools from previously curated repository and the Jina Reader API[2] for content extraction. For each query, the system retrieves top-$K_{\text{doc}} = 3$ URLs. To

[2] `https://jina.ai/`

prevent context overflow and filter noise, we employ `gpt-4o-mini` as a summarizer, condensing each webpage into a summary of at most 200 tokens while retaining task-relevant key points.

The pipeline operates iteratively. At the conclusion of each cycle, the LRM reviews the accumulated search history, spanning generated descriptions, selected tools, executed queries, and summarized evidence. Based on this context, the model determines whether to terminate and generate a final answer or, if evidence is deemed insufficient, to start a new search round. In the main experiment, the max round number is set to $T_{\text{max}} = 3$. To ensure reproducibility, we saved the entire trajectory and web page cache, enabling offline replay.

4.2 Multi-Granularity Reward Design

To adapt LRMs to this dynamic search environment, we design a multi-granularity reward mechanism and fine-tune models of varying scales (4B, 8B, and 14B) using GRPO. These models are optimized by maximizing various reward signals respectively derived from different stages of the pipeline.

Planning Reward (R_{plan}) To incentivize the generation of precise abstract descriptions, we evaluate the quality of the downstream retrieval. Let $\mathcal{T}_{\text{retrieved}}$ be the set of top-k tools retrieved via the generated descriptions, and $\mathcal{T}_{\text{reference}}$ be the ground-truth set. The reward is defined as:

$$R_{\text{plan}} = \text{F1}(\mathcal{T}_{\text{retrieved}}, \mathcal{T}_{\text{reference}}). \tag{2}$$

This actually rewards descriptions that are semantically consistent with the target tools. Moreover, we have also discussed about equivalent signals. Further analysis can be found in Appendix D.

Execution Reward (R_{exec}) This component evaluates the model's ability to select correct tools and formulate effective queries. We design a hybrid reward combining discrete selection accuracy and continuous semantic relevance:

$$R_{\text{exec}} = \mathbb{I}(|\mathcal{T}_{\text{select}}| \leq 5) \cdot [0.5 \cdot \text{F1}(\mathcal{T}_{\text{select}}, \mathcal{T}_{\text{reference}}) + 0.5 \cdot \text{Sim}(\mathcal{C}_{\text{retrieved}}, \mathcal{C}_{\text{target}})]. \tag{3}$$

Here, $\mathbb{I}(\cdot)$ is an indicator function that penalizes saturation attacks by setting the reward to 0 if the model selects more than 5 tools. $\text{Sim}(\cdot)$ denotes semantic similarity, $\mathcal{C}_{\text{retrieved}}$ is the aggregated content fetched by the queries, and $\mathcal{C}_{\text{target}}$ is the concatenation of user task and reference answer.

Response Reward (R_{resp}) To ensure the final output is helpful and truthful, we utilize normalized scores from our LLM-as-a-Judge mechanism (detailed in Section 5.1.3). The reward is the average score of Accuracy S_{Acc}, Completeness S_{Com}, and Relevance S_{Rel}:

$$R_{\text{resp}} = \frac{1}{3}(S_{\text{Acc}} + S_{\text{Com}} + S_{\text{Rel}}). \tag{4}$$

These three types of independent samples are obtained by splitting complete trajectories and are trained using the same mixing ratio.

5 Experiments

5.1 Evaluation Metrics

We establish a multi-dimensional evaluation framework to assess performance across four critical stages: Tool Selection, Query Formulation, Final Answer Generation, and Cost and Efficiency.

5.1.1 Tool Selection

We evaluate the alignment between selected tool set $\mathcal{T}_{\text{select}}$ and ground-truth set $\mathcal{T}_{\text{ref}}$ using standard **Precision**, **Recall**, and **F1-Score**. Additionally, we introduce the **Semantic Redundancy Rate (SRR)** to penalize the selection of functionally overlapping tools. A tool $t \in \mathcal{T}_{\text{select}}$ is deemed redundant if $t \notin \mathcal{T}_{\text{ref}}$ but exhibits high semantic similarity (cosine similarity > 0.7) to a correctly selected tool $t' \in (\mathcal{T}_{\text{select}} \cap \mathcal{T}_{\text{ref}})$. SRR is calculated as:

$$\text{SRR} = \frac{|\{t \in \mathcal{T}_{\text{select}} \mid \text{is_redundant}(t)\}|}{|\mathcal{T}_{\text{select}}|}. \tag{5}$$

5.1.2 QUERY FORMULATION

We assess the effectiveness of generated queries $\mathcal{Q} = \{q_i\}_{i \in \mathbb{N}^*}$ through two quantitative metrics:

Validity This metric measures the syntactic correctness and feasibility of the generated queries. Let $\mathcal{U}(q_i)$ denote the set of non-empty URLs retrieved by executing query q_i. The validity score S_{valid} is defined as the proportion of queries that yield at least one result:

$$S_{\text{valid}} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(|\mathcal{U}(q_i)| > 0). \tag{6}$$

Quality Considering the ever-changing web environment, we use semantic similarity instead of exact match to quantify the coverage of the retrieved content with the target information. Let $\mathcal{C}(q_i)$ be the textual content extracted from the URL set $\mathcal{U}(q_i)$. The quality score S_{qual} is calculated as the average cosine similarity between the retrieved content embedding and the target embedding:

$$S_{\text{qual}} = \frac{1}{N} \sum_{i=1}^{N} \text{Sim}\left(\mathcal{C}(q_i), \mathcal{C}_{\text{target}}\right). \tag{7}$$

High value of S_{qual} means the retrieved information is semantically consistent with the ground truth.

5.1.3 FINAL ANSWER GENERATION

We deploy a robust LLM-as-a-Judge mechanism Li et al. (2024) using `gpt-4o-mini`. To mitigate subjectivity, we implement a *Key Point Extraction* protocol. Prior to evaluation, LLM extracts 5–15 atomic key points from the ground-truth answer to serve as an objective checklist. Then the model's answer is scored across three dimensions: **Accuracy**, **Completeness** and **Relevance**. These indicator serve as the ultimate evaluation criteria. What's more, we employed `claude-sonnet-4.5` and `gemini-2.5-flash` as additional LLM judges and engaged independent human evaluators to assess the final answers from `MultiSearch-R`, providing detailed correlation analysis of Pearson (r) and Spearman (ρ) in Appendix E.

5.1.4 COST AND EFFICIENCY

Beyond response quality, we evaluate operational efficiency via three indicators:

Token Consumption Reasoning cost is quantified by the total number of output tokens generated across all steps, in order to isolate reasoning from architectural variances in input context length.

API Calls The total count of search actions, serving as a proxy for latency and monetary cost.

Step Count The number of iteration cycles ($0 \leq T \leq T_{\max}$), reflecting the model's ability to autonomously determine information sufficiency.

5.2 BASELINES

We compare our framework against two types of baselines: internal variants to validate architectural choices, and state-of-the-art deep search agents.

5.2.1 INTERNAL ARCHITECTURAL VARIANTS

Vanilla Single-Source The model is restricted to a single generic search interface. This acts as a control group to demonstrate the limitations of general-purpose engines in handling domain-specific, multi-hop queries. (*Single Search* in Table 1)

Full-Context Multi-Source The model receives complete list of tool names and descriptions directly in the context window. This setting tests the ability to handle massive search spaces without retrieval filtering. (*Multi Search Full* in Table 1)

Naive Retrieval We employ a standard dense retriever that matches the user task against tool descriptions. This baseline investigates whether deep reasoning via description generation is requisite to bridge the semantic gap between user intent and tool functionality. (*Multi Search Filter* in Table 1)

Table 1: Comprehensive evaluation results across different model scales and deep search agent frameworks. The best and second-best results in each category are indicated in **bold** and underlined, respectively. ***Closed-source Models*** means replacing the backbone in `MultiSearch-R` with Claude and Gemini. Please refer to Section 5.2.1 for the meaning of *Single Search*, *Multi Search Full* and *Multi Search Filter*. **Note:** Final Answer Score represents the ultimate evaluation criteria.

Method	Tool Selection				Query Formulation		Final Answer Generation				Cost and Efficiency		
	Prec.↑	Rec.↑	F1↑	Red.↓	Valid.↑	Qual.↑	Acc.↑	Comp.↑	Rel.↑	Avg.↑	Tokens↓	APIs↓	Steps↓
Closed-source Models													
Claude Sonnet 4.6	0.3670	0.6094	0.4333	0.0377	0.9386	0.6342	0.9140	0.9220	0.9210	0.9190	724.45	3.42	1.06
Gemini 2.5 Pro	0.3167	0.6285	0.3932	0.0227	0.9759	0.6306	0.8010	0.8640	0.8560	0.8403	704.38	5.40	1.87
4B Parameter Scale													
Single Search	–	–	–	–	**0.9845**	**0.6587**	0.5780	0.6800	0.7130	0.6570	205.66	1.93	1.93
Multi Search Full	0.2404	0.3685	0.2714	0.0170	0.6927	0.5864	0.4590	0.5640	0.6030	0.5420	188.30	3.71	1.58
Multi Search Filter	**0.4843**	**0.6180**	**0.5144**	**0.0150**	0.8231	0.6230	0.4890	0.5910	0.6320	0.5707	**182.50**	2.77	1.29
Search-o1-Qwen3-4B	0.2634	0.5017	0.3149	0.0237	0.6942	0.6140	0.7110	0.7690	0.7540	0.7447	592.96	6.77	2.07
`MultiSearch-R` Base	0.3766	0.5192	0.4100	0.0256	0.9161	0.5942	0.5900	0.7480	0.8040	0.7140	202.23	**1.43**	**0.60**
`MultiSearch-R`	0.3150	0.5966	0.3790	0.0234	0.9568	0.5842	**0.8478**	**0.8812**	**0.8971**	**0.8754**	424.19	4.03	1.43
7B/8B Parameter Scale													
Single Search	–	–	–	–	0.9895	**0.6697**	0.5250	0.6490	0.6710	0.6150	173.31	1.90	1.90
Multi Search Full	0.3773	0.5178	0.4116	0.0211	0.8394	0.6001	0.3770	0.4950	0.5350	0.4690	157.40	2.74	1.16
Multi Search Filter	**0.5498**	**0.6739**	**0.5810**	0.0054	0.8525	0.6307	0.4360	0.5490	0.5890	0.5247	160.57	3.05	1.28
Search-o1-Qwen3-8B	0.2382	0.6279	0.3167	0.0221	0.8244	0.5970	0.5730	0.6430	0.6210	0.6123	590.83	5.58	1.26
Search-R1-7B	0.1111	0.1414	0.1182	**0.0000**	0.6255	0.5796	0.3430	0.4670	0.5050	0.4383	**79.75**	2.43	2.43
WebThinker-Qwen3-8B	0.1722	0.1963	0.1637	0.0111	**1.0000**	0.5699	0.5790	0.6370	0.6610	0.6257	135.58	**0.87**	**0.73**
`MultiSearch-R` Base	0.3589	0.6364	0.4221	0.0290	0.9432	0.6014	0.7850	0.8380	0.8640	0.8290	399.04	5.11	1.85
`MultiSearch-R`	0.4421	0.6246	0.4873	0.0145	0.8972	0.6165	**0.8470**	**0.8550**	**0.8850**	**0.8623**	438.19	3.99	1.65
14B Parameter Scale													
Single Search	–	–	–	–	0.9765	**0.6634**	0.5220	0.6470	0.6780	0.6157	177.46	1.70	1.70
Multi Search Full	0.2466	0.3534	0.2750	0.0124	0.7091	0.5722	0.3840	0.5120	0.5270	0.4743	163.95	3.61	1.30
Multi Search Filter	0.4967	**0.6281**	0.5305	0.0118	0.8185	0.6255	0.4630	0.5880	0.6090	0.5533	170.57	3.36	1.42
Search-o1-Qwen3-14B	0.2294	0.5484	0.2962	0.0356	0.7242	0.6108	0.6790	0.7580	0.7440	0.7270	585.73	9.50	2.30
WebThinker-R1-14B	0.1758	0.1848	0.1721	**0.0091**	**1.0000**	0.5823	0.4880	0.5660	0.5840	0.5460	**85.94**	1.35	0.90
WebThinker-Qwen3-14B	0.1825	0.2619	0.1905	0.0238	**1.0000**	0.5728	0.5590	0.6200	0.6390	0.6060	165.34	**0.50**	**0.34**
`MultiSearch-R` Base	0.3906	0.5758	0.4317	0.0130	0.8580	0.6010	0.6570	0.7290	0.7860	0.7240	309.66	3.52	1.48
`MultiSearch-R`	**0.5311**	0.6158	**0.5488**	0.0096	0.8634	0.6240	**0.8210**	**0.8520**	**0.8790**	**0.8507**	366.43	2.27	1.32

5.2.2 Agent Frameworks

Search-o1 A prompting-based reasoning framework. We evaluate it using the official prompts on Qwen3 backbones to ensure fair comparison.

Search-R1 An RL-optimized search agent. We use the official checkpoint fine-tuned on Qwen2.5-7B.

WebThinker An advanced reasoning-enhanced agent. We evaluate both official checkpoint and a prompting-only variant with Qwen3 as backbones.

5.3 Results and Analysis

Table 1 presents a comprehensive performance comparison across various model scales. The metrics in the table integrate single-source and multi-source data, with the Final Answer Score serving as the primary outcome. Tool-level and query-level metrics are used to diagnose intermediate behaviors, rather than to replace end-to-end evaluation. We analyze these results to interpret the specific phenomena and trade-offs observed.

5.3.1 RL From Capability to Efficiency

As shown in Table 1, `MultiSearch-R` achieves Final Answer Scores of 0.8754, 0.8623, and 0.8507 at 4B, 8B, and 14B, respectively, delivering consistent improvements over baselines. For example, relative to Search-o1-Qwen3-14B, we achieve a higher answer score (0.8507 vs. 0.7270) with lower cost and efficiency, illustrating a favorable trade-off. WebThinker baselines perform extremely low API calls and step counts, indicating a tendency to prematurely terminate searches. As for `MultiSearch-R` Base, the accompanying changes in search activity and iteration steps differ by scale. At 4B, API calls increase from 1.43 to 4.03, and the corresponding steps count increase from 0.60 to 1.43, indicating that the model learns to utilize more resources for information seeking in order to bridge the reasoning gap. At 8B and 14B, higher answer scores accompany fewer calls and step count, showing reduced redundancy. The reported token consumption nevertheless increases at all three scales. Therefore, RL has altered the allocation of effort regarding search and generation for models of varying scales.

5.3.2 HIGH RECALL, LOW QUALITY

High tool-selection metrics do not necessarily translate into high answer quality. For example, at the 8B scale, *Multi Search Filter* achieves a higher tool-selection F1 than `MultiSearch-R` (**0.5810** vs. **0.4873**), but a substantially lower final answer score (**0.5247** vs. **0.8623**). To assess whether this gap can be reduced by expanding the retrieved candidate pool, we vary the number of candidate tools in *Multi Search Filter* (MSF) over $K \in \{5, 10, 20, 50\}$. Across these settings, its best final answer score on Qwen3-8B is **0.6017**, compared with **0.8623** for `MultiSearch-R` (Figure 7). Thus, enlarging the static candidate pool alone is insufficient to close the observed performance gap.

To investigate where the gap arises within this pipeline, we provide a complementary stage-wise diagnostic study in Appendix F, further identifing limitations beyond candidate retrieval. Replacing MSF's candidates with the reference tools raises its score only from 0.5247 to 0.5390. With reference URLs supplied, replacing truncated raw evidence with two-stage summaries raises the score from 0.6007 to 0.8343. As a result, downstream evidence handling is an important bottleneck. These findings highlight the importance of considering the combination of components like tool selection, downstream evidence acquisition, processing, and answer generation as a whole pipeline.

5.3.3 QUERY LEVEL DIAGNOSTICS

Our framework demonstrates superior query construction capabilities. While *Single Search* achieves high Validity and Quality due to the simplicity of keyword queries, this metric masks the difficulty of multi-source interaction. In contrast, our RL agent successfully adapts to complex constraints. Specifically, it boosts Query Validity at 4B and 14B scale, and Query Quality at 8B and 14B scale.

6 ABLATION STUDIES

Ablation studies assessed key `MultiSearch-R` components on the 8B model (Table 12). Compared to the full system, each ablation configuration yields a lower reference tool F1 score and requires more search calls. (1) **Necessity of Multi-Granularity Rewards:** Relying solely on the final outcome causes a sharp decline in F1 (-34.7%) and a significant increase in API Calls ($+29.6\%$). Removing **Planning Reward** and **Execution Reward** also results in worse performance, indicating that high-level planning and explicit validity feedback is crucial. (2) **Impact of Requirement Analysis:** Removing this module leads to degradation in all metrics. This suggests that planning before retrieval effectively narrows the search space, allowing the model to target correct tools more precisely.

We further investigated the impact of the inference budget by varying the maximum rounds allowed ($T_{\max} \in \{1, 3, 5, 10, 15\}$). As illustrated in Figure 6, relaxing the round constraint leads to a consistent improvement in performance, demonstrating the model utilizes additional compute for deeper reasoning. However, API Calls, Step Count, and Token Consumption exhibit a clear trend of diminishing marginal costs. When thresholds are set high, the average rounds remain well below, indicating that the system does not consistently exhaust allocated budget. Our RL strategy features an efficient autonomous termination mechanism that ensures a high benefit-to-cost ratio, even under relatively loose constraints, reflecting an empirical trade-off between quality and resource utilization.

7 CONCLUSION AND LIMITATIONS

In this work, we introduced `MultiSearch-Bench`, a benchmark dedicated to fine-grained selection among hundreds of functionally similar search tools. Our analysis revealed the "High Recall, Low Quality" phenomenon, demonstrating static retrieval often fails to support complex agentic workflows. To overcome this, we proposed `MultiSearch-R`, which combines iterative requirement analysis, source selection, and query formulation with multi-granularity GRPO training. Our findings advocate for a paradigm shift from simple keyword matching to reasoning-driven resource planning, treating the selection of internal website sources as a fundamental step in Deep Search.

While our work offers significant insights into multi-source agentic search, we acknowledge several limitations. The majority of sources are simulated using domain-restricted method, which may not fully capture the complex query syntax. What's more, it remains to be explored how our strategies scale when the candidate pool expands and context window constraints become more severe.

AI USE STATEMENT

We used pretrained language models in the data construction, search pipeline, and evaluation. GPT-4o-mini was used for scenario and task generation, reference-answer and key-point generation, webpage summarization, and LLM-based reward and evaluation. Claude Sonnet 4.5 and Gemini 2.5 Flash provided supplementary assessments, alongside human evaluation. We also used Codex for program assisting and manuscript editing. The authors reviewed all AI-assisted text and take responsibility for the final manuscript and all artifacts.

ETHICS STATEMENT

MultiSearch-Bench is intended for research on search source selection and evidence synthesis. Web sources and model generated references may contain errors, outdated information, or biases. Generated answers should not be treated as professional advice. Human assessment complements automated scoring. Reuse of retrieved content and deployment of the search interfaces must respect the original providers' licenses, terms of service, and access restrictions.

REPRODUCIBILITY STATEMENT

The benchmark construction, method, and evaluation are described in Sections 3, 4 and 5, with implementation details, pseudocode, and prompts in Appendices B, C and I. We retain query results, retrieved webpage content, intermediate summaries, and complete execution trajectories. These cached artifacts support offline replay of recorded runs. The source code is available through the anonymous repository linked in the paper.

REFERENCES

Tamer Abuelsaad, Deepak Akkil, Prasenjit Dey, Ashish Jagmohan, Aditya Vempaty, and Ravi Kokku. Agent-e: From autonomous web navigation to foundational design principles in agentic systems. *arXiv preprint arXiv:2407.13032*, 2024.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*, volume 2024, pp. 9112–9141, 2024.

Mingyang Chen, Linzhuang Sun, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z Pan, Wen Zhang, Huajun Chen, et al. Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025.

Yanfei Chen, Jinsung Yoon, Devendra Singh Sachan, Qingze Wang, Vincent Cohen-Addad, Mohammadhossein Bateni, Chen-Yu Lee, and Tomas Pfister. Re-invoke: Tool invocation rewriting for zero-shot tool retrieval. *arXiv preprint arXiv:2408.01875*, 2024.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114, 2023.

Yang Deng, Xuan Zhang, Wenxuan Zhang, Yifei Yuan, See Kiong Ng, and Tat-Seng Chua. On the multi-turn instruction following for conversational web agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8795–8812, 2024.

Xiang Fei, Xiawu Zheng, and Hao Feng. Mcp-zero: Active tool discovery for autonomous llm agents. *arXiv preprint arXiv:2506.01056*, 2025.

Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. *arXiv preprint arXiv:2305.11854*, 2023.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Hongming Zhang, Tianqing Fang, Zhenzhong Lan, and Dong Yu. Openwebvoyager: Building multimodal web agents via iterative real-world exploration, feedback and optimization. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27545–27564, 2025.

Peter C Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Adam Santoro, and Timothy Lillicrap. A data-driven approach for learning to control computers. In *International Conference on Machine Learning*, pp. 9466–9482. PMLR, 2022.

Iat Long Iong, Xiao Liu, Yuxuan Chen, Hanyu Lai, Shuntian Yao, Pengbo Shen, Hao Yu, Yuxiao Dong, and Jie Tang. Openwebagent: An open toolkit to enable web agents on large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pp. 72–81, 2024.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

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

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. *Advances in Neural Information Processing Systems*, 36:39648–39677, 2023.

Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language model agents. *arXiv preprint arXiv:2407.01476*, 2024.

Terry K Koo and Mae Y Li. A guideline of selecting and reporting intraclass correlation coefficients for reliability research. *Journal of chiropractic medicine*, 15(2):155–163, 2016.

Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, et al. Autowebglm: A large language model-based web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 5295–5306, 2024.

I Lawrence and Kuei Lin. A concordance correlation coefficient to evaluate reproducibility. *Biometrics*, pp. 255–268, 1989.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Haitao Li, Qian Dong, Junjie Chen, Huixue Su, Yujia Zhou, Qingyao Ai, Ziyi Ye, and Yiqun Liu. Llms-as-judges: a comprehensive survey on llm-based evaluation methods. *arXiv preprint arXiv:2412.05579*, 2024.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*, 2025a.

Xiaoxi Li, Wenxiang Jiao, Jiarui Jin, Guanting Dong, Jiajie Jin, Yinuo Wang, Hao Wang, Yutao Zhu, Ji-Rong Wen, Yuan Lu, et al. Deepagent: A general reasoning agent with scalable toolsets. *arXiv preprint arXiv:2510.21618*, 2025b.

Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yongkang Wu, Ji-Rong Wen, Yutao Zhu, and Zhicheng Dou. Webthinker: Empowering large reasoning models with deep research capability. *arXiv preprint arXiv:2504.21776*, 2025c.

Xiao Liu, Hanyu Lai, Hao Yu, Yifan Xu, Aohan Zeng, Zhengxiao Du, Peng Zhang, Yuxiao Dong, and Jie Tang. Webglm: towards an efficient web-enhanced question answering system with human preferences. In *Proceedings of the 29th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 4549–4560, 2023.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. Gaia: a benchmark for general ai assistants. In *The Twelfth International Conference on Learning Representations*, 2023.

Lorenzo Molfetta, Giacomo Frisoni, Nicolò Monaldini, and Gianluca Moro. Ports: Preference-optimized retrievers for tool selection with large language models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 10018–10041, 2025.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, et al. Webgpt: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Yixin Ou, Wangchunshu Zhou, Shengwei Ding, Long Li, Jialong Wu, Tiannan Wang, Jiamin Chen, Shuai Wang, Xiaohua Xu, Ningyu Zhang, et al. Symbolic learning enables self-evolving agents. *AI Open*, 2025.

Shishir G Patil, Tianjun Zhang, Xin Wang, and Joseph E Gonzalez. Gorilla: Large language model connected with massive apis. *Advances in Neural Information Processing Systems*, 37: 126544–126565, 2024.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, et al. Toolllm: Facilitating large language models to master 16000+ real-world apis. *arXiv preprint arXiv:2307.16789*, 2023.

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. Towards completeness-oriented tool retrieval for large language models. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 1930–1940, 2024.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *Advances in Neural Information Processing Systems*, 36:68539–68551, 2023.

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

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Zhengliang Shi, Yuhan Wang, Lingyong Yan, Pengjie Ren, Shuaiqiang Wang, Dawei Yin, and Zhaochun Ren. Retrieval models aren't tool-savvy: Benchmarking tool retrieval for large language models. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 24497–24524, 2025.

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

Yueqi Song, Frank F Xu, Shuyan Zhou, and Graham Neubig. Beyond browsing: Api-based web agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 11066–11085, 2025b.

Shuang Sun, Huatong Song, Yuhao Wang, Ruiyang Ren, Jinhao Jiang, Junjie Zhang, Fei Bai, Jia Deng, Wayne Xin Zhao, Zheng Liu, et al. Simpledeepsearcher: Deep information seeking via web-powered reasoning trajectory synthesis. *arXiv preprint arXiv:2505.16834*, 2025.

Jiejun Tan, Zhicheng Dou, Wen Wang, Mang Wang, Weipeng Chen, and Ji-Rong Wen. Htmlrag: Html is better than plain text for modeling retrieved knowledge in rag systems. In *Proceedings of the ACM on Web Conference 2025*, pp. 1733–1746, 2025.

Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, Boxi Cao, and Le Sun. Toolalpaca: Generalized tool learning for language models with 3000 simulated cases. *arXiv preprint arXiv:2306.05301*, 2023.

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, et al. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*, 2025.

Shulin Tian, Ziniu Zhang, Liang-Yu Chen, and Ziwei Liu. Mmina: Benchmarking multihop multimodal internet agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 13682–13697, 2025.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 10014–10037, 2023.

Shuai Wang, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. Resllm: Large language models are strong resource selectors for federated search. In *Companion Proceedings of the ACM on Web Conference 2025*, pp. 1360–1364, 2025.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

Nancy Xu, Sam Masling, Michael Du, Giovanni Campagna, Larry Heck, James Landay, and Monica Lam. Grounding open-domain instructions to automate web support tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1022–1032, 2021.

Qiancheng Xu, Yongqi Li, Heming Xia, and Wenjie Li. Enhancing tool retrieval with iterative feedback from large language models. *arXiv preprint arXiv:2406.17465*, 2024.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*, 2022.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. Assistantbench: Can web agents solve realistic and time-consuming tasks? *arXiv preprint arXiv:2407.15711*, 2024.

Penghao Zhao, Hailin Zhang, Qinhan Yu, Zhengren Wang, Yunteng Geng, Fangcheng Fu, Ling Yang, Wentao Zhang, Jie Jiang, and Bin Cui. Retrieval-augmented generation for ai-generated content: A survey. *arXiv preprint arXiv:2402.19473*, 2024.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v (ision) is a generalist web agent, if grounded. *arXiv preprint arXiv:2401.01614*, 2024.

Yuxiang Zheng, Dayuan Fu, Xiangkun Hu, Xiaojie Cai, Lyumanshan Ye, Pengrui Lu, and Pengfei Liu. Deepresearcher: Scaling deep research via reinforcement learning in real-world environments. *arXiv preprint arXiv:2504.03160*, 2025.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*, 2023a.

Wangchunshu Zhou, Yuchen Eleanor Jiang, Long Li, Jialong Wu, Tiannan Wang, Shi Qiu, Jintian Zhang, Jing Chen, Ruipu Wu, Shuai Wang, et al. Agents: An open-source framework for autonomous language agents. *arXiv preprint arXiv:2309.07870*, 2023b.

APPENDIX

A MULTISEARCHBENCH DETAILS

A.1 THE PROPORTION OF TOOL CATEGORY AND SUBCATEGORY

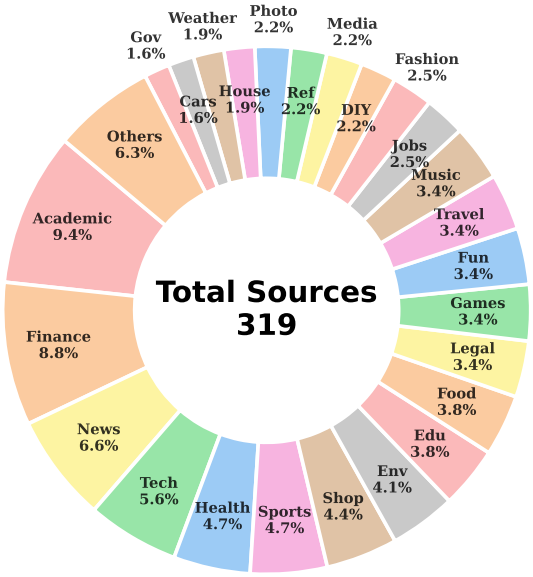

Figure 3: Distribution of search tools across 30 main categories in `MultiSearch-Bench`.

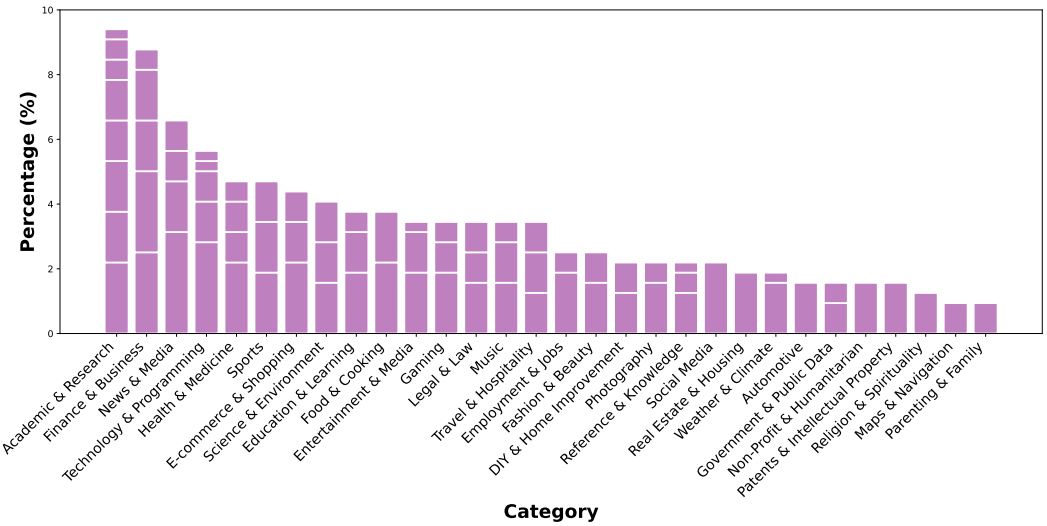

Figure 4: Detailed Distribution of MultiSearchBench

For details on the specific number of search tools included in each Category and Subcategory, please refer to Table 2 and Table 3.

Table 2: Search Tools Repository Details: Part I

Category	Category Count	Subcategory	Subcategory Count
Academic & Research	30	Astrophysics & Physics	1
		Computer Science & Tech Research	4
		Economics & Social Sciences	2
		General Academic	5
		Life Sciences & Medicine	5
		Mathematics	2
		Multidisciplinary Databases	4
		Preprints & Open Access	7
Automotive	5	Car Reviews & News	5
DIY & Home Improvement	7	Home Decor	3
		Home Improvement	4
E-commerce & Shopping	14	Fashion & Apparel	3
		Online Marketplaces	7
		Price Comparison & Reviews	4
Education & Learning	12	Online Learning Platforms	6
		Test Prep	4
		University Rankings	2
Employment & Jobs	8	Job Search	6
		Salary Information	2
Entertainment & Media	11	Books & Literature	1
		Movie & TV	6
		Streaming Platforms	4
Fashion & Beauty	8	Beauty & Cosmetics	3
		Fashion News	5
Finance & Business	28	Business News & Analysis	8
		Cryptocurrency	2
		Financial News & Data	8
		Market Data & Tools	5
		Personal Finance	5
Food & Cooking	12	Recipe Sites	7
		Restaurant Reviews	5
Gaming	11	Gaming Communities	2
		Gaming News	6
		Gaming Platforms	3
Government & Public Data	5	International Organizations	2
		U.S. Government	3
Health & Medicine	15	Drug Information	3
		Medical Information	7
		Mental Health	2
		Public Health	3
Legal & Law	11	Government & Courts	3
		Legal News	3
		Legal Research	5
Maps & Navigation	3	Mapping Services	3
Music	11	Music Information	2
		Music News & Reviews	4
		Music Streaming	5
News & Media	21	International News	10
		Investigative Journalism	3
		News Aggregators	5
		Political Analysis	3
Non-Profit & Humanitarian	5	International Organizations	5
Parenting & Family	3	Parenting Resources	3
Patents & Intellectual Property	5	Patent Databases	5
Photography	7	Photography Communities	5
		Photography Resources	2
Real Estate & Housing	6	Real Estate Listings	6
Reference & Knowledge	7	Dictionaries & Thesaurus	4
		Encyclopedias	2
		Q&A Communities	1
Religion & Spirituality	4	Religious Texts & Resources	4
Science & Environment	13	Environment	4
		General Science	5
		Space & Astronomy	4
Social Media	7	Major Platforms	7
Sports	15	League-Specific	6
		Sports News	5
		Stats & Fantasy	4

Table 3: Search Tools Repository Details: Part II

Category	Category Count	Subcategory	Subcategory Count
Technology & Programming	18	AI & Machine Learning	1
		Developer Communities & Resources	4
		Product Reviews	1
		Programming Documentation	3
		Tech News	9
Travel & Hospitality	11	Airlines	3
		Travel Booking	4
		Travel Guides	4
Weather & Climate	6	Climate Data	1
		Weather Services	5

A.2 Benchmarks Comparison

Table 4 situates `MultiSearch-Bench` within broader landscape of tool learning and web navigation benchmarks. Unlike existing datasets that focus on massive but functionally distinct API calls or generic web interactions, `MultiSearch-Bench` is uniquely designed to evaluate fine-grained decision-making. Specifically, it introduces **Homogeneous Distractors** to test an agent's ability to distinguish between highly similar vertical search sources. Additionally, it provides evaluation of **Adaptive Querying**, measuring ability to adjust queries based on different search sources.

Table 4: Comparison with existing agentic search and tool learning benchmarks.

	Information Source Scale	Homogeneous Distractors	Adaptive Querying	Deep Web Access	Inter-domain Synthesis
Toolformer	5 APIs	✗	✗	✗	✗
WebArena	4 categories	✗	✗	✗	✓
Mind2Web	137 websites	✗	✗	✓	✓
ToolGen	16k tools, 47k APIs	✗	✗	✓	✓
ToolkenGPT	Hundreds of tools	✗	✗	✓	✓
ToolLLM	3451 tools, 16464 APIs	✗	✗	✓	✓
`MultiSearch-Bench` **(Ours)**	319 sources	✓	✓	✓	✓

A.3 Custom Search Wrappers

We choose 10 representative platforms wrapped as insite search source for complex internal architectures. As for the rest websites, we use Serper Search API[3] for simulated domain search. All automated scraping complies with the website's terms of service.

- Wikipedia: https://www.wikipedia.org/
- Stack Overflow: https://stackoverflow.com/
- GitHub: https://github.com/
- arXiv: https://arxiv.org/
- Amazon: https://www.amazon.com/
- IMDb: https://www.imdb.com/
- Goodreads: https://www.goodreads.com/
- Medium: https://medium.com/
- BBC News: https://www.bbc.com/
- Nature: https://www.nature.com/

A.4 Detailed Analysis of MultiSearch-Bench

As illustrated in Figure 1, we have collected 800 single source data and 200 multi source data for the entire benchmark. Then we shuffled the dataset and utilized 80% for training, 10% for validating, and the rest 10% for testing.

[3] `https://serper.dev/`

Table 5: Detailed Structure of MultiSearch-Bench

MultiSearch-Bench	Single	Multi	Total
Training Set	640	160	800
Validation Set	80	20	100
Test Set	80	20	100
Total	800	200	1000

Regarding the size of the test set split, we conducted statistical significance tests to quantify uncertainty. Specifically, for the various methods across all model sizes, we calculated the bootstrap 95% confidence intervals (10,000 resamples) and conducted paired permutation tests (10,000 permutations) on the Final Answer Score.

The mean scores with a bootstrap 95% CI of MultiSearch-R at 4B, 8B, and 14B are 0.8754 ([0.8464, 0.9010]), 0.8623 ([0.8383, 0.8837]), and 0.8507 ([0.8217, 0.8763]), respectively. Critically, the upper bound of the highest baseline 95% CI (0.7790 of Search-o1-Qwen3 4B) falls below the lower bound of MultiSearch-R (0.8217). The forest plot of bootstrap CIs is reported below.

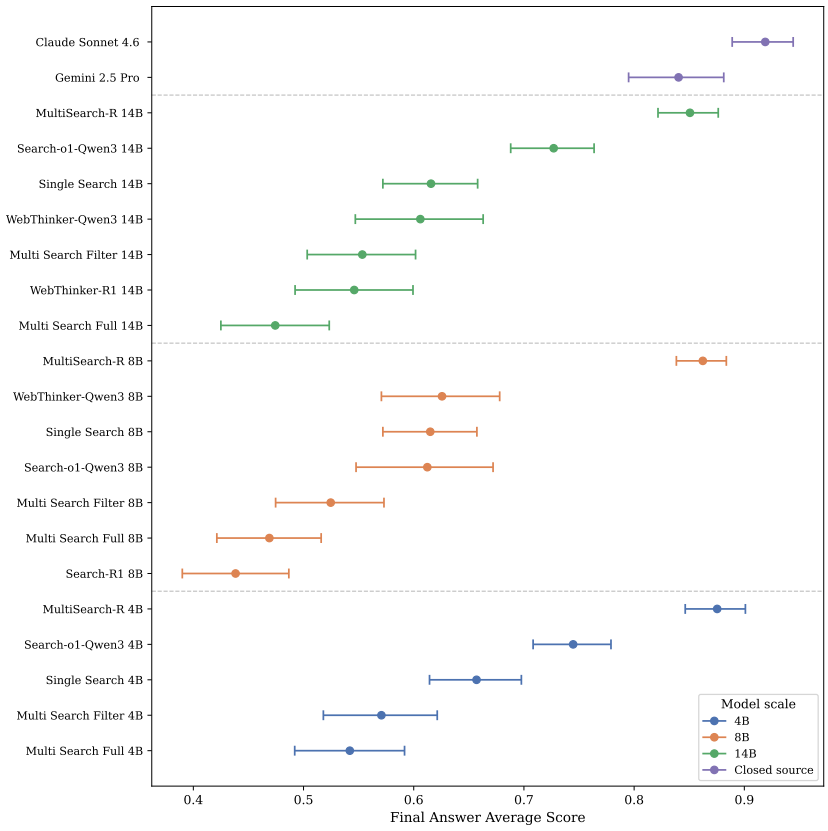

Figure 5: Statistical Significance Tests

Across all comparisons in Table 6, MultiSearch-R achieves positive paired mean differences ranging from 0.1237 to 0.4240, with Cohen's d_z ranging from 0.61 to 1.95 and Monte Carlo p-values of approximately 10^{-4}. These results provide statistical evidence of consistent improvements over the evaluated open-source baselines on the available test instances.

Table 6: Paired comparisons of `MultiSearch-R` against the evaluated open-source baselines. The Δ Score is the average difference between the Final Answer Score of `MultiSearch-R` and that of baseline, computed over n shared test instances. All permutation tests yield unadjusted $p \approx 10^{-4}$. Cohen's d_z reports the paired effect size.

Scale	Baseline	Δ Score	Cohen's d_z
4B	Single Search	+0.2184	0.98
	Multi Search Full	+0.3334	1.22
	Multi Search Filter	+0.3047	1.30
	Search-o1-Qwen3	+0.1307	0.72
8B	Single Search	+0.2473	1.30
	Multi Search Full	+0.3933	1.76
	Multi Search Filter	+0.3376	1.46
	Search-o1-Qwen3	+0.2500	0.82
	Search-R1	+0.4240	1.95
	WebThinker-Qwen3	+0.2366	0.92
14B	Single Search	+0.2350	1.21
	Multi Search Full	+0.3764	1.54
	Multi Search Filter	+0.2974	1.38
	Search-o1-Qwen3	+0.1237	0.61
	WebThinker-R1	+0.3047	1.05
	WebThinker-Qwen3	+0.2447	0.91

B Implementation Details

For the backbone large reasoning model in `MultiSearch-R`, we utilize the open-sourced Qwen3-4B, Qwen3-8B, and Qwen3-14B Yang et al. (2025). For generation settings, we use a maximum of 32,768 tokens, temperature of 0.7, top_p of 0.8, top_k of 20, and a repetition penalty of 1.05 across all models. For retrieval, we employ the Serper Google Search API, setting the region to US-EN and the top-k retrieved documents to 10. We use Jina Reader API to fetch the content of web pages for given URLs. Detailed instructions for all models are provided in Appendix I. All experiments are conducted on eight NVIDIA A800-80GB GPUs.

C Pipeline Pseudocode

The pseudocode for the entire pipeline of our algorithm `MultiSearch-R` is detailed in Algorithm 1.

D Equivalent F1 Metrics

The annotation set records the tools used to construct the reference solution rather than an exhaustive collection of all valid solutions. Consequently, Exact-F1 measures consistency with the reference tools. Nevertheless, this reward mechanism may penalize valid alternative sources. We supplement this with Subcategory-F1 and Category-F1 metrics, calculated after mapping tools to their corresponding category label sets. The final answer score remains our primary evaluation metric.

For task i, let P_i be the set of distinct tools selected and R_i be the reference tool set. For granularity $g \in \{\text{exact}, \text{subcat}, \text{cat}\}$, let π_g map a tool to its identity, subcategory label, or category label, respectively. Using the repository taxonomy, we form

$$P_i^{(g)} = \{\pi_g(t) : t \in P_i\}, \qquad R_i^{(g)} = \{\pi_g(t) : t \in R_i\}, \tag{8}$$

and compute

$$F_{1,i}^{(g)} = \frac{2\,|P_i^{(g)} \cap R_i^{(g)}|}{|P_i^{(g)}| + |R_i^{(g)}|}, \qquad \overline{F}_1^{(g)} = \frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} F_{1,i}^{(g)}. \tag{9}$$

Algorithm 1 Iterative Multi-Source Search Pipeline

Input: User task u, Tool repository $\mathcal{T}$
Output: Final answer A

$H \leftarrow \emptyset, t \leftarrow 0, T_{\max} \leftarrow 3, K_{\text{tool}} \leftarrow 5, K_{\text{doc}} \leftarrow 3$, finished $\leftarrow$ False
while $t < T_{\max}$ **and** $\neg$finished **do**
  // Phase 1: Requirement Analysis
  $\mathcal{D}_{\text{abs}} \leftarrow \text{LRM}_{\text{plan}}(u, H)$ ▷ Generate abstract tool descriptions
  // Phase 2: Tool Retrieval
  $\mathcal{T}_{\text{retrieve}} \leftarrow \text{Retrieve}(\mathcal{T}, \mathcal{D}_{\text{abs}}, K_{\text{tool}})$ ▷ Retrieve candidate tools
  // Phase 3: Decision Making
  $(\mathcal{T}_{\text{sel}}, Q) \leftarrow \text{LRM}_{\text{decide}}(u, \mathcal{T}_{\text{retrieve}}, H)$ ▷ Tool selection and query formulation
  // Phase 4: Information Synthesis
  $E_t \leftarrow \emptyset$
  for each query $q \in Q$ **do**
    $\mathcal{U}(q) \leftarrow$ Custom Search Wrappers *or* SerperAPI(q, K_{doc}) ▷ Search and retrieve URLs
    $\mathcal{C}(q) \leftarrow$ JinaReaderAPI$(\mathcal{U}(q))$ ▷ Extract content from URLs
    $s \leftarrow$ Summarize$(\mathcal{C}(q))$ ▷ Summarize content
    $E_t \leftarrow E_t \cup \{s\}$
  end for
  $H \leftarrow H \cup \{\mathcal{D}_{\text{abs}}, \mathcal{T}_{\text{sel}}, Q, E_t\}$ ▷ Update history context
  $(\text{finished}, A) \leftarrow \text{LRM}_{\text{review}}(u, H)$ ▷ Check termination and generate answer
  $t \leftarrow t + 1$
end while
return A

This is F1 over sets of labels. Multiple tools with the same labels are merged into a single element. Shared subcategories or categories serve as indicators of functional relevance.

We reuse stored trajectories and answer scores for the Multi Search Full, Multi Search Filter, and MultiSearch-R systems at the 4B, 8B, and 14B scales. To reproduce the Exact-F1 metrics reported, the set $\mathcal{I}$ comprises available records where both P_i and R_i are non-empty. The same set $\mathcal{I}$ is used for all three granularity settings within a given system. Table 7 presents the corresponding results.

Table 7: Equivalent F1 with Final Answer Score. Full, Filter and MSR denote Multi Search Full, Multi Search Filter and MultiSearch-R, respectively. The best and second-best results in each category are indicated in **bold** and underlined, respectively.

Scale	Method	Exact-F1	Subcat-F1	Cat-F1	Final Answer Score
4B	Full	0.2714	0.3693	0.5063	0.5420
	Filter	**0.5144**	**0.6468**	**0.8360**	<u>0.5707</u>
	MSR	<u>0.3790</u>	<u>0.4834</u>	<u>0.6769</u>	**0.8754**
8B	Full	0.4116	0.5474	0.7218	0.4690
	Filter	**0.5810**	**0.7151**	**0.8601**	<u>0.5247</u>
	MSR	<u>0.4873</u>	<u>0.5911</u>	<u>0.8114</u>	**0.8623**
14B	Full	0.2750	0.3771	<u>0.5361</u>	0.4743
	Filter	<u>0.5305</u>	**0.6497**	**0.8182**	<u>0.5533</u>
	MSR	**0.5488**	<u>0.6312</u>	**0.8182**	**0.8507**

Across all system-level average metrics, Subcat-F1 consistently outperforms Exact-F1, and Cat-F1 scores show further improvement. These differences indicate that the measured alignment results are sensitive to the granularity of the reference labels. To some extent, these alternative metrics help mitigate the misjudgments associated with Exact-F1. Replacing Exact-F1 supervision with validated equivalence-aware rewards requires a separate retraining study and remains future work.

E CORRELATION ANALYSIS

To mitigate the potential bias inherent in relying on a single LLM as an evaluator, we utilized three human annotators and three model judges (GPT-4o-mini, Claude Sonnet 4.5, and Gemini 2.5 Flash) to evaluate answers generated by MultiSearch-R 8B. All evaluators applied the same scoring criteria. Human annotators scored the answers independently. For each example and dimension, we defined the arithmetic mean of the three human annotators' scores as the human consensus result.

Table 8 reports Pearson (r) and Spearman (ρ) between each model judge and the human consensus. Claude and Gemini exhibit high overall correlations with the human consensus, with Pearson correlations of 0.852 and 0.841 and Spearman correlations of 0.812 and 0.824, respectively. GPT also exhibits positive overall agreement ($r = 0.699$, $\rho = 0.669$).

Table 8: Correlation analysis of the LLM-as-a-Judge evaluation. This table shows Pearson (r) and Spearman (ρ) correlation coefficients between each LLM judge and human consensus.

LLM Judge	Accuracy	Completeness	Relevance	Overall
GPT-4o-mini	0.677 / 0.627	0.690 / 0.669	0.632 / 0.419	0.699 / 0.669
Claude Sonnet 4.5	0.818 / 0.784	0.834 / 0.812	0.837 / 0.813	0.852 / 0.812
Gemini 2.5 Flash	0.820 / 0.834	0.841 / 0.831	0.740 / 0.696	0.841 / 0.824

Futhermore, we assess inter-annotator reliability using mean pairwise Pearson/Spearman correlations and two-way random-effects, absolute-agreement intraclass correlation coefficients Koo & Li (2016). ICC(2,1) describes the reliability of an individual annotator, whereas ICC(2,3) describes the reliability of the mean of three annotators. As shown in Table 9, the overall ICC increases from 0.634 for an individual rating to 0.839 for the three-annotator mean, supporting the use of aggregated human ratings as a more reliable reference.

Table 9: Agreement among three human annotators. Mean r and ρ are arithmetic averages across the three annotator pairs. ICC(2,1) and ICC(2,3) measure absolute-agreement reliability for an individual rating and the mean of three ratings, respectively. Brackets denote bootstrap 95% confidence intervals.

Dimension	Mean r	Mean ρ	ICC(2,1) [95% CI]	ICC(2,3) [95% CI]
Accuracy	0.617	0.606	0.533 [0.423, 0.624]	0.774 [0.688, 0.833]
Completeness	0.714	0.719	0.634 [0.530, 0.718]	0.839 [0.772, 0.884]
Relevance	0.520	0.524	0.496 [0.383, 0.589]	0.747 [0.651, 0.812]
Overall	0.681	0.690	0.634 [0.527, 0.717]	0.839 [0.770, 0.884]

Among the three scoring dimensions, relevance yielded the lowest values across both the pairwise human correlations and the two ICC metrics. It exhibits a higher degree of subjectivity at the level of human annotation. The evidence therefore supports both dimension-dependent variability and differences among evaluators.

Correlation alone does not establish agreement in absolute scores. We therefore additionally report mean signed differences, mean absolute error (MAE), and Lin's concordance correlation coefficient (CCC), which accounts for departures from agreement in location and scale as well as correlation Lawrence & Lin (1989).

Table 10 shows that GPT assigns overall scores +2.242 points higher than the human consensus on average and has a CCC of 0.290. Claude and Gemini exhibit smaller mean differences (+0.356 and +0.139) and higher CCC values (0.802 and 0.769). Thus, independent Claude and Gemini judges show substantially stronger human alignment and absolute calibration. GPT exhibits substantial leniency and score compression while retaining moderate rank agreement with human consensus. Therefore, when weighing effectiveness against cost, GPT is a rather good choice. The statistically significant positive alignment across the board validates the reliability of LLM-as-a-Judge employed in our experiments.

Table 10: Absolute agreement between model judges and human consensus for overall scores. Δ is the average signed difference between the evaluator and the human consensus. Brackets denote bootstrap 95% confidence intervals.

LLM Judge	Mean	Δ	MAE	CCC [95% CI]
GPT-4o-mini	8.623	+2.242	2.242	0.290 [0.168, 0.403]
Claude Sonnet 4.5	6.737	+0.356	0.978	0.802 [0.722, 0.857]
Gemini 2.5 Flash	6.520	+0.139	1.123	0.769 [0.684, 0.831]

F Stage-wise Diagnosis of the Answer-Quality Gap

We diagnose the gap of *Multi Search Filter* (MSF) with `MultiSearch-R` at 8B scale. C1 is the original MSF result, using the tool candidates retrieved from the user task. C2 replaces candidates with the reference tools, where the model still selects tools and generates queries. C3 is a separate control using the tools selected by MultiSearch-R. C4 directly injects the reference URLs, bypassing tool selection and query generation, and passes truncated raw evidence to the answer generator. C5 uses the same reference URLs with two-stage processing: `gpt-4o-mini` summarizes each raw evidence and then summarizes the concatenated summaries.

Table 11: Stage-wise diagnosis on the test split at 8B scale.

ID	Configuration	Acc.	Comp.	Rel.	Avg.
C1	Original Multi Search Filter	0.4360	0.5490	0.5890	0.5247
C2	Reference Tool Candidate Pool	0.4440	0.5800	0.5930	0.5390
C3	MultiSearch-R Tool Candidate Pool	0.4780	0.5870	0.6090	0.5580
C4	Reference URLs + Raw Evidence	0.4970	0.6520	0.6530	0.6007
C5	Reference URLs + Two-stage Summaries	0.7930	0.8430	0.8670	0.8343
–	MultiSearch-R	**0.8470**	**0.8550**	**0.8850**	**0.8623**

Source selection, query formulation, and evidence processing form a dependent search pipeline. Selecting appropriate sources determines where relevant evidence can be sought, while subsequent stages determine whether that evidence is recovered and effectively used. Giving the golden tool set changes little **(C2-C1=+0.0143)**, because the Multi Search Filter itself is already capable of retrieving the majority of correct sources. The improvement in C3 indicates that, after RL our model has learned to find tools consistent with the reasoning path, and even replacing the same candidate pool shows improvement **(C3-C1=+0.0333)**. The residual gap also emerges downstream. Replacing URLs adds a little **(C4-C2=+0.0617)**, and the largest gain is evidence processing **(C5-C4=+0.2337)**. Even with indeed golden sources in hand, a naive pipeline fails to convert them into complete answers. Table 11 shows that success requires both correct selection and faithful evidence conversion, and MultiSearch-R achieves both ends at once. Therefore, these results support the effectiveness of the entire pipeline and the interdependence of its various stages.

Table 12: Ablation studies of `MultiSearch-R` (8B).

Method	Tool F1	Query Quality	Answer Score	API Calls
`MultiSearch-R`	**0.487**	**0.617**	**0.862**	**3.99**
w/o Planning Reward	0.442	0.600	0.855	4.30
w/o Execution Reward	0.309	0.596	0.851	5.71
Sparse Reward Only	0.318	0.612	0.852	5.17
w/o Requirement Analysis	0.461	0.614	0.857	4.32

G Ablation Studies

Table 12 shows some relevant indicators when removing certain multi-granularity rewards and components in `MultiSearch-R`.

Figure 6 shows the parameter sensitivity analysis about the maximum rounds allowed.

Figure 7 shows how the Final Answer Score varies with the size of the retrieved tool candidate pool.

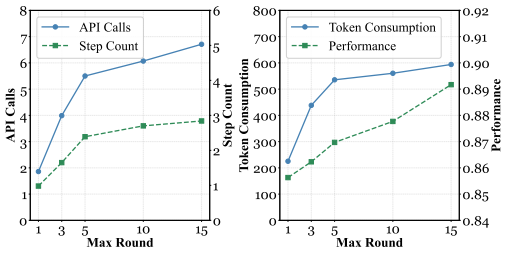

Figure 6: Performance across Max Rounds.

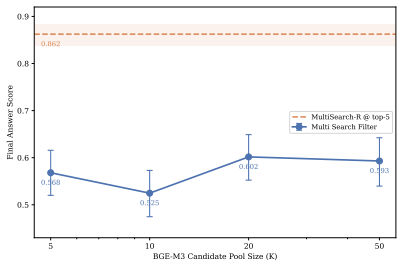

Figure 7: Score across Candidate Pool Size.

H Case Study

In this section, we present two representative examples selected from the evaluation framework for `MultiSearch-R`, with **Qwen3-8B** as backbone, to testify our findings in Section 5.3.

H.1 Case Study 1: Optimized Queries and Precise Execution

GOAL Demonstrate how the model correctly identifies vertical tools and facilitates higher-quality queries.

ID 933 Box H.1 illustrates a legal query requiring specific procedural knowledge ("filing a lawsuit") and access to restricted records ("court records").

- **Tool Selection**: The model accurately bypasses general encyclopedias, selecting **PACER** for records, **Westlaw** and **FindLaw** for procedures. This aligns perfectly with the ground truth, achieving a Recall of 1.0. Specifically, the model choose **Westlaw** and **PACER** in round 1, **FindLaw** and **Cornell Legal Information Institute** in round 2, achieving a Precision of 0.75. Although only one answer is incorrect, it is still a useful search tool.
- **Adaptive Querying**: Crucially, the model sends different queries to different tools. For **Westlaw**, it issues a broad procedural query ("steps to file..."), recognizing Westlaw's capability to retrieve unstructured legal guides. For **PACER**, it switches to a functional query ("accessing... after filing"), aligning with PACER's specific database structure.
- **Outcome**: This precise tool-query alignment results in a perfect evaluation score (**10/10/10**), validating that our RL model learns to navigate the specific "search personalities" of vertical platforms.

Optimized Queries and Precise Execution

{
"id": 933,
"task": "What steps do I need to take to file a federal lawsuit, and how can I access relevant court records once my case is filed?",
"reference_apis": [
"Westlaw",
"PACER",
"FindLaw"

],
"reference_urls": [
"https://pacer.uscourts.gov/find-case",
"https://www.findlaw.com/litigation/filing-a-lawsuit/
starting-the-case-initial-court-papers.html",
"http://www.westlaw.com/Link/Document/FullText?findType=L&pubNum=
1000600&cite=USFRCPR17&originatingDoc=I21c5e1a04d8f1",
"https://www.findlaw.com/legal/law-students/surviving-law-school/
westlaw-for-law-students.html",
"https://www.findlaw.com/state/education-laws.html",
"https://pacer.uscourts.gov/"
],
"reference_answer": "To file a federal lawsuit, you need to complete specific initial court documents, which typically include a complaint and summons. You can find detailed guidance on the necessary initial court papers for civil lawsuits on the FindLaw website. Once your case is filed, you can access relevant court records through the Public Access to Court Electronic Records (PACER) system. First, you will need to register for a PACER account, which will allow you to search for your case either by specific court or via a nationwide index. If you need to access court records by phone, certain bankruptcy courts offer limited information through their voice case information systems. Additionally, the PACER system charges $0.10 per page for access to documents, but fees are waived if your total charges are $30 or less in a given quarter.",
"final_answer": "To file a federal lawsuit and access court records, follow these steps:### **Steps to File a Federal Lawsuit**1. **Determine the Appropriate Court**: Identify the federal court with jurisdiction over your case (e.g., U.S. District Court). Check requirements for venue and subject-matter jurisdiction.2. **Prepare Legal Documents**: Draft a complaint outlining your claims, facts, and relief sought, along with a summons to notify the defendant. Use official forms from the [Federal Judicial Center](https://www.fjc.gov/) or [PACER](https://www.pacer.gov/).3. **File with the Court**: Submit the complaint and summons to the court clerk. Pay the filing fee (or request a waiver if eligible). Courts may accept paper or electronic filings (e.g., via PACER).4. **Serve the Defendant**: Ensure the defendant receives a copy of the complaint and summons via certified mail or a process server.5. **Track Your Case**: Use PACER or the court's case management system to monitor filings and deadlines.### **Accessing Federal Court Records After Filing**1. **Register for PACER**: Create a free account at [PACER](https://www.pacer.gov/) to access public court records, including complaints, orders, and filings. Note: There is a per-page fee ($0.10–$0.30, capped at $3.00 per document). 2. **Search via PACER Case Locator**: Use case numbers, party names, or keywords to find documents. 3. **Explore Court Websites**: Visit the specific district court's website (e.g., [NDCA](https://www.ndca.us/)) for local rules and additional resources.4. **Check for Fee Waivers**: If you qualify for indigent status, request fee waivers for PACER or court filings.**Resources**: - [FindLaw](https://www.findlaw.com/) provides procedural checklists and forms.- [Cornell Legal Information Institute](https://www.law.cornell.edu/) offers free federal rules and statutes.- [PACER](https://www.pacer.gov/) is the primary tool for accessing court records. Always verify requirements with the specific court handling your case, as procedures may vary slightly by jurisdiction.",
"history_log": [
{
"round": 1,
"descriptions": [
"Official U.S. federal court procedures and forms for initiating litigation",
"Database for accessing public federal court records and case filings",
"Legal resource portal for tutorials and guidelines on civil procedure"
],
"retrieved_tools": [
{
"PACER": "Electronic public access service for U.S. federal court documents and case information."
},
{
"Westlaw": "Westlaw is an online legal research service and proprietary database for lawyers and legal professionals available in over 60 countries. Information resources on Westlaw include more than 40,000 databases of case law, state and federal statutes, administrative codes, newspaper and magazine articles, public records, law journals, law reviews, treatises, legal forms and other information resources."
},
{
"Data.gov": "Data.gov is a U.S. government website launched in late May 2009 by the federal chief information officer of the United States, Vivek Kundra. Data.gov aims to improve public access to high

value, machine-readable datasets generated by the Executive Branch of the federal government. The site is a repository for federal, state, local, and tribal government information made available to the public."
},
{
"Congress.gov": "Congress.gov is the online database of United States Congress legislative information. Congress.gov is a joint project of the Library of Congress, the House, the Senate and the Government Publishing Office."
},
{
"LexisNexis": "LexisNexis is an American data analytics company headquartered in New York, New York. Its products are various databases that are accessed through online portals, including portals for computer-assisted legal research (CALR), newspaper search, and consumer information. During the 1970s, LexisNexis began to make legal and journalistic documents more accessible electronically. As of 2006, the company had the world's largest electronic database for legal and public-records–related information. The company is a subsidiary of RELX."
}
],
"selected_interface": [
{
"name": "Westlaw",
"query": "steps to file a federal lawsuit"
},
{
"name": "PACER",
"query": "accessing federal court records after filing a case"
}
],
"search_urls": [
"https://pacer.uscourts.gov/",
"https://pacer.uscourts.gov/find-case",
"https://pacer.uscourts.gov/help/faqs"
],
"summarized_content": "To file a federal lawsuit, register for a PACER account and follow the filing instructions on the PACER website. After filing, access court records through PACER by searching via the PACER Case Locator or specific court. Access costs $0.10 per page, capped at $3.00 per document. For assistance, contact the PACER Service Center at (800) 676-6856."
},
{
"round": 2,
"descriptions": [
"Official U.S. federal court guidelines and forms for civil litigation procedures",
"Database of federal legal forms and procedural checklists for plaintiffs",
"Tutorial resource on federal court rules and filing requirements"
],
"retrieved_tools": [
{
"PACER": "Electronic public access service for U.S. federal court documents and case information."
},
{
"Westlaw": "Westlaw is an online legal research service and proprietary database for lawyers and legal professionals available in over 60 countries. Information resources on Westlaw include more than 40,000 databases of case law, state and federal statutes, administrative codes, newspaper and magazine articles, public records, law journals, law reviews, treatises, legal forms and other information resources."
},
{
"Cornell Legal Information Institute": "The Legal Information Institute (LII) is a non-profit public service of Cornell Law School that provides no-cost access to current American and international legal research sources online. Founded in 1992 by Peter Martin and Tom Bruce, LII was the first law site developed on the internet. LII electronically publishes on the Web the U.S. Code, U.S. Supreme Court opinions, Uniform Commercial Code, the US Code of Federal Regulations, several Federal Rules, and a variety of other American primary law materials. LII also provides access to other national and international sources, such as treaties and United Nations materials. According to its website, the LII serves over 40 million unique visitors per year."

```
},
{
"FindLaw": "FindLaw is a business of Internet Brands that provides online legal information in the form of state laws, case law and codes, legal blogs and articles, a lawyer directory, DIY legal services and products, and other legal resources. The company also provides online marketing services for law firms. FindLaw was created by Stacy Stern, Martin Roscheisen, and Tim Stanley in 1995, and was acquired by Thomson West in 2001."
},
{
"Tasty": "Recipe website with short video recipes and easy cooking instructions."
}
],
"selected_interface": [
{
"name": "FindLaw",
"query": "steps to file a federal lawsuit"
},
{
"name": "Cornell Legal Information Institute",
"query": "accessing federal court records after filing a case"
}
],
"search_urls": [
"https://www.findlaw.com/litigation/filing-a-lawsuit/
starting-the-case-initial-court-papers.html",
"https://www.findlaw.com/litigation/filing-a-lawsuit/
how-to-file-a-class-action-lawsuit.html",
"https://www.findlaw.com/litigation/filing-a-lawsuit.html",
"https://www.law.cornell.edu/rules/frcp/rule_44",
"https://www.law.cornell.edu/rules/frcp/rule_5.2",
"https://www.law.cornell.edu/wex/pacer"
],
"summarized_content": "To file a federal lawsuit, determine the appropriate court, prepare a complaint and summons, file them with the court, pay the filing fee, and serve the defendant. After filing, access court records via the PACER system by registering for an account. As of June 2024, document access costs $0.30 per page, with a maximum of $3.00 per document. Fee waivers may be available."
}
],
"evaluation": {
"completeness": 10,
"accuracy": 10,
"relevance": 10
}
}
```


H.2 Case Study 2: Functional Equivalency over Semantic Matching

GOAL Demonstrate why F1 score is imperfect and how the model succeeds by choosing functionally equivalent tools.

ID 938 Box H.2 presents a complex comparative health task (COVID-19 vs. Legionellosis).

- **Selection Deviation**: The Ground Truth includes *WHO*, *Red Cross* and *MedlinePlus*. However, our model selects *WHO* and *CDC*. While this lowers the standard Tool Selection Index, it is a strategically superior choice.
- **Reasoning Path**: The model recognizes that for "emergency response guidelines" and "comparative protocols," the *CDC* serves as a functional superset of the *Red Cross* in terms of technical authority. It substitutes a general aid organization with a specialized national public health agency.
- **Outcome**: Despite the deviation in tool selection, the model constructs a comprehensive answer, correctly distinguishing between "behavioral interventions" (COVID) and "envi-

ronmental controls" (Legionellosis). The high final evaluation scores (**9/9/9**) reinforce our finding in Section 5.3: *functional alignment* for the reasoning path is more critical than superficial *semantic alignment* with a fixed ground truth.

Architectural Awareness and Precise Execution

{
"id": 938,
"task": "What are the latest safety recommendations for preventing the spread of COVID-19, and how do they compare to the guidelines for dealing with other respiratory diseases like Legionellosis, particularly in emergency response efforts?",
"reference_apis": [
"MedlinePlus",
"World Health Organization",
"Red Cross"
],
"reference_urls": [
"https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public",
"https://medlineplus.gov/ency/patientinstructions/000973.htm",
"https://medlineplus.gov/",
"https://www.who.int/news-room/fact-sheets/detail/legionellosis"
],
"reference_answer": "To prevent the spread of COVID-19, the World Health Organization (WHO) advises several measures: staying at least 6 feet away from others, wearing face masks indoors in public if transmission rates are high, ensuring proper ventilation in indoor spaces, and getting vaccinated. The American Red Cross also emphasizes the importance of hand hygiene, wearing masks in crowded or poorly ventilated settings, and staying home if symptomatic or tested positive. In contrast, Legionellosis, caused by inhaling water droplets containing Legionella bacteria, requires different preventative measures, such as maintaining water systems to prevent bacterial growth and avoiding exposure to contaminated water sources. Emergency response efforts for both diseases involve coordinating health services and ensuring access to medical care, but the strategies differ based on the transmission modes and environmental factors specific to each disease. The WHO and American Red Cross collaborate to improve emergency responses by emphasizing the need for emergency medical teams to be adequately trained and equipped to handle such public health threats.",
"final_answer": "The latest safety recommendations for preventing COVID-19 spread emphasize vaccination (including updated boosters for emerging variants), mask-wearing in crowded or indoor settings, physical distancing, hand hygiene, and improved ventilation. These measures prioritize individual behavior changes to curb high transmissibility. In contrast, Legionellosis prevention focuses on environmental controls, such as maintaining water system temperatures, regular maintenance of cooling towers and hot tubs, and disinfection to prevent bacterial growth (e.g., Legionella pneumophila). Emergency responses for COVID-19 prioritize rapid vaccination drives, contact tracing, and public health messaging, while Legionellosis outbreaks require immediate water system inspections and remediation. Both diseases stress protecting vulnerable populations and clear communication, but COVID-19 guidelines are broader and more adaptable to community settings, whereas Legionellosis protocols are highly specific to infrastructure management. Key differences in emergency efforts include: (1) COVID-19 relies on behavioral interventions and scalability, while Legionellosis demands targeted environmental interventions; (2) COVID-19 responses often involve mass vaccination campaigns, whereas Legionellosis requires specialized water safety audits. Sources: CDC (2024 guidelines), WHO (emergency response frameworks), and CDC outbreak reports (2015–2019).",
"history_log": [
{
"round": 1,
"descriptions": [
"Latest public health guidelines from CDC/WHO on COVID-19 prevention and control measures",
"Academic papers or comparative analyses of respiratory disease protocols (e.g., Legionellosis vs. COVID-19)",
"Emergency response frameworks from health organizations (e.g., FEMA, WHO) for infectious disease outbreaks"
],
"retrieved_tools": [
{

"World Health Organization": "The World Health Organization (WHO) is a specialized agency of the United Nations which coordinates responses to international public health issues and emergencies. It is headquartered in Geneva, Switzerland, and has six regional offices and 150 field offices worldwide. Only sovereign states are eligible to join, and it is the largest intergovernmental health organization at the international level."
},
{
"NAMI": "Mental health advocacy organization providing education, support, and resources for mental illness."
},
{
"FDA": "The Food and Drug Administration (FDA) is a federal agency of the United States Department of Health and Human Services. The FDA is responsible for protecting and promoting public health through the control and supervision of food safety, tobacco products, caffeine products, dietary supplements, prescription and over-the-counter pharmaceutical drugs (medications), vaccines, biopharmaceuticals, blood transfusions, medical devices, electromagnetic radiation emitting devices (ERED), cosmetics, animal foods & feed, and veterinary products."
},
{
"Centers for Disease Control": "The Centers for Disease Control and Prevention (CDC) is the national public health agency of the United States. It is a United States federal agency under the Department of Health and Human Services (HHS), and is headquartered in Atlanta, Georgia."
},
{
"Doctors Without Borders": "Médecins Sans Frontières, sometimes known as Doctors Without Borders, is a charity that provides humanitarian medical care. It is a non-governmental organisation (NGO) of French origin known for its projects in conflict zones and in countries affected by endemic diseases. The organisation provides care for diabetes, drug-resistant infections, HIV/AIDS, hepatitis C, tropical and neglected diseases, tuberculosis, vaccines and COVID-19. In 2019, the charity was active in 70 countries with over 35,000 personnel; mostly local doctors, nurses and other medical professionals, logistical experts, water and sanitation engineers, and administrators. Private donors provide about 90% of the organisation's funding, while corporate donations provide the rest, giving MSF an annual budget of approximately US$1.63 billion."
}
],
"selected_interface": [
{
"name": "World Health Organization",
"query": "latest safety recommendations for preventing spread of COVID-19"
},
{
"name": "Centers for Disease Control",
"query": "guidelines for Legionellosis in emergency response"
},
{
"name": "World Health Organization",
"query": "comparison of COVID-19 and Legionellosis safety guidelines in emergency response"
}
],
"search_urls": [
"https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public",
"https://www.who.int/teams/risk-communication/covid-19-transmission-package",
"https://www.who.int/health-topics/coronavirus",
"https://www.cdc.gov/control-legionella/php/toolkit/cooling-towers-module.html",
"https://www.cdc.gov/control-legionella/media/pdfs/toolkit.pdf",
"https://www.cdc.gov/control-legionella/php/toolkit/potable-water-systems-module.html",
"https://www.who.int/europe/news-room/fact-sheets/item/water--sanitation-and-hygiene",

"https://www.who.int/docs/default-source/coronaviruse/clinical-management-of-novel-cov.pdf",
"https://www.who.int/docs/default-source/food-safety/arsenic/9789241549950-eng.pdf?sfvrsn=bad6319a_2"
],
"summarized_content": "Latest COVID-19 safety recommendations emphasize vaccination, mask-wearing in crowded places, physical distancing, hand hygiene, and good ventilation. In contrast, Legionellosis prevention focuses on managing water systems to control bacterial growth, ensuring proper water temperatures, and regular maintenance. Both require rapid identification and control measures in emergency responses, but COVID-19 guidelines prioritize individual behavior changes due to its higher transmissibility, while Legionellosis emphasizes environmental management. Both highlight the importance of protecting vulnerable populations and public health communication."
},
{
"round": 2,
"descriptions": [
"Comparative analysis articles from peer-reviewed journals on respiratory disease management protocols",
"Emergency response case studies from FEMA or WHO detailing COVID-19 vs. Legionellosis interventions",
"Public health policy documents from the CDC outlining specific emergency preparedness frameworks"
],
"retrieved_tools": [
{
"Centers for Disease Control": "The Centers for Disease Control and Prevention (CDC) is the national public health agency of the United States. It is a United States federal agency under the Department of Health and Human Services (HHS), and is headquartered in Atlanta, Georgia."
},
{
"PACER": "Electronic public access service for U.S. federal court documents and case information."
},
{
"Drugs.com": "Drugs.com is an online pharmaceutical encyclopedia that provides drug information for consumers and healthcare professionals, primarily in the United States. It self-describes its information as "accurate and independent" yet limited to being "for educational purposes only and is not intended for medical advice, diagnosis or treatment.""
},
{
"WebMD": "WebMD is an American corporation which publishes online news and information about human health and well-being. The WebMD website also includes information about drugs and is an important healthcare information website and the most popular consumer-oriented health site."
},
{
"FDA": "The Food and Drug Administration (FDA) is a federal agency of the United States Department of Health and Human Services. The FDA is responsible for protecting and promoting public health through the control and supervision of food safety, tobacco products, caffeine products, dietary supplements, prescription and over-the-counter pharmaceutical drugs (medications), vaccines, biopharmaceuticals, blood transfusions, medical devices, electromagnetic radiation emitting devices (ERED), cosmetics, animal foods & feed, and veterinary products."
}
],
"selected_interface": [
{
"name": "Centers for Disease Control",
"query": "comparison of COVID-19 and Legionellosis emergency response guidelines"
},
{
"name": "Centers for Disease Control",
"query": "safety recommendations for Legionellosis versus COVID-19 in public health emergencies"
}
],
"search_urls": [
"https://www.cdc.gov/mmwr/volumes/73/wr/pdfs/mm7342-H.pdf",

"https://www.cdc.gov/infection-control/hcp/isolation-precautions/scientific-review.html",
"https://www.cdc.gov/mmwr/volumes/70/wr/pdfs/mm7020-H.pdf",
"https://www.cdc.gov/infection-control/hcp/isolation-precautions/scientific-review.html",
"https://www.cdc.gov/infection-control/media/pdfs/guideline-environmental-h.pdf",
"https://www.cdc.gov/mmwr/index2024.html"
],
"summarized_content": "As of October 2024, CDC recommends COVID-19 prevention through updated vaccinations targeting Omicron variants, mask-wearing in crowded settings, and good ventilation. In contrast, Legionellosis prevention focuses on controlling water systems to prevent bacterial growth, particularly in hot tubs and cooling towers. Emergency responses for both diseases involve rapid identification and containment, but COVID-19 prioritizes vaccination and public health measures, while Legionellosis emphasizes environmental controls. From 2015-2019, 208 outbreaks linked to treated recreational water were reported, with 65 (42%) caused by Legionella, resulting in 354 cases and 13 deaths."
}
],
"evaluation": {
"completeness": 9,
"accuracy": 9,
"relevance": 9
}
}

I PROMPTS

This section outlines the instructions utilized in this work, including those for benchmark construction and evaluation.

I.1 PROMPT IN BENCHMARK CONSTRUCTION

Prompt for Combination & Scenario Generation

Your Task
Generate as many diverse, reasonable and high-quality API combinations with realistic usage scenarios as possible.
Each combination should represent a coherent information retrieval workflow that a user might need.
Combination Requirements
- **Size**: 2-5 APIs per combination
- **Uniqueness**: Each API must provide irreplaceable information, no redundant APIs that serve the same purpose
- **Coherence**: APIs should form a logical workflow, not random selections
- **Diversity**: Cover different domains, use cases, and patterns across your generated combinations
- **Quality over similarity**: Avoid multiple APIs from the same organization or narrow category
Scenario Requirements
- Describe a realistic user situation where this specific API combination would be necessary
- Should clearly explain why ALL APIs in the combination are needed and what unique contribution each makes
- Use few sentences to keep it concise but specific
- Should reflect practical information needs, not artificial scenarios
- Must demonstrate that removing any API would leave a gap in the information workflow
Combination Patterns Use for Inspiration
Explore diverse patterns to maximize variety:
- **Sequential workflows**: Start broad → narrow down → get specifics
- **Cross-validation**: Multiple perspectives on the same topic
- **Complementary coverage**: Fill different information gaps
- **Multi-domain synthesis**: Combine insights across fields
- **Depth variations**: Expert + accessible sources

- **Temporal analysis**: Track evolution over Time
Input
Available APIs:
{tool_list}
Output Format (JSON only)
Generate a JSON array where each element contains an API combination and its scenario:
```json
[{"apis": ["api_name_1", "api_name_2", "api_name_3"], "scenario": "A clear, specific description of when and why a user would need this exact combination of APIs"}, {"apis": ["api_name_4", "api_name_5"], "scenario": "Another realistic scenario requiring these specific APIs"}]
```

Prompt for Task Generation

Your Task
Generate one high-quality realistic user question and its corresponding reference answer pair based on multiple webpage contents.
Your generation should meet the following criteria:
Question Requirements
- Must require information from **ALL** provided webpages to answer completely, missing any one webpage would result in an incomplete answer
- Does not need to use every detail from each webpage, only the relevant parts needed for a coherent question
- Should sound natural and authentic, like something a real user would ask with a clear purpose
- Should have strong intent, avoid vague or overly broad questions
- Can be any type: fact-finding, multi-hop reasoning, comparative analysis, causal reasoning, synthesis, procedural questions, etc.
- Should be specific enough to have a definitive answer, but complex enough to require cross-referencing multiple sources
- When webpages have weak logical connections and lack strong thematic coherence, you may generate parallel/composite questions to ensure all pages are covered.
Answer Requirements
- Must be based **entirely** on the provided webpage content, no fabrication, speculation, or external knowledge
- Should be comprehensive and directly address the question
- Must incorporate relevant information from all webpages in a coherent way
- Should be factual, accurate, and verifiable against the source material
- Can organize information logically (e.g., step-by-step, comparative structure, causal chains) to improve clarity
Quality Guidelines
- **Coherence**: The question should have a clear logical connection between the different webpages, not just randomly combining unrelated facts
- **Practical relevance**: Prefer questions that reflect realistic information needs users might have
- **Appropriate complexity**: The question should justify needing multiple sources, not artificially complex, but genuinely requiring synthesis
- **Answer completeness**: The reference answer should feel complete and satisfying, not leaving obvious gaps
- **Specificity**: Include specific details, numbers, names, or constraints where appropriate to make the question concrete
- **Avoid redundancy**: Don't ask for information that's repeated across all pages, leverage the unique contribution of each source
Input
{webpages}
Output Format (JSON only)
```json
{"task": "your generated question here", "reference_answer": "your reference answer here"}
```

Prompt for Keyword Extraction

Your Task
Generate search keywords that a user would most likely use in the given scenario to retrieve information from the specified APIs.
Your keywords will be used to search each API domain and retrieve the top-1 result from each.
Keyword Requirements
- **Realistic**: Generate keywords that real users would actually type in search engines for this scenario
- **Effective**: Keywords should be likely to return relevant, high-quality results from the specified APIs
- **Concise**: Typically few words per keyword query, natural search phrases, not full sentences
- **Targeted**: Consider what information the user needs and what each API specializes in
- **Multiple queries allowed**: You can generate 1-3 different keyword queries to cover different aspects of the information need
Keyword Strategies
- **Specific terms**: Include key entities, names, concepts, or technical terms relevant to the scenario
- **Natural phrasing**: Use how people actually search
- **Avoid over-complexity**: Don't use overly long or awkward phrases that would return poor results
- **Consider API specialization**: Think about what type of content each API domain provides and what keywords work well there
- **Information intent**: Match the user's goal, finding facts, comparisons, tutorials, news, research, etc.
Input
APIs:
{apis}
Scenario:
{scenario}
Output Format (JSON only)
Generate a JSON array of keyword query strings:
```json
{"keywords": ["keyword query 1", "keyword query 2", "keyword query 3"]}
```

I.2 Prompt in Evaluation Pipeline

Prompt for Description Generation

You are an intelligent search agent tasked with helping users gather comprehensive information to their request through strategic use of search APIs.
WORKFLOW OVERVIEW
The process follows these steps in a single round:
1. **Generate Tool Descriptions OR Final Answer** (CURRENT STEP) - Decide if more information is needed; if yes, describe what types of search tools you need; if no, provide final answer
2. Retrieve Tool Subset - System retrieves relevant APIs based on your descriptions
3. Select Tools & Generate Queries - Choose specific tools and create search queries
4. Execute Search - System collects information from selected sources you chose
KNOWN INFORMATION
Now you are in the **round {round_num}** of the interaction.
The max number of rounds allowed is {max_rounds}.
User's Request:
{task}
History:
{history}
Understanding the History:
• Each round shows: Generated Descriptions → Retrieved Tools → Selected Tools and Generated Queries → Collected Information
• "Generated Descriptions" are the tool descriptions you requested in that round
• "Retrieved Tools" are the APIs the system found matching your descriptions
• "Selected Tools and Generated Queries" are the specific tools you chose and the queries you created
• "Collected Information" summarizes what was found from those sources
YOUR MISSION
You need to decide whether to:

(A) CONTINUE SEARCHING
Consider this if previous search results are incomplete or need verification from other sources and different perspectives or complementary information.
(B) PROVIDE FINAL ANSWER
Consider this if You have collected enough comprehensive information to fully address the user's request and provide a complete and well-rounded answer.
OUTPUT FORMAT
Option A: If you decide to CONTINUE SEARCHING
Generate clear, specific, and diverse tool descriptions that capture different aspects of the information you need.
Each description should:
• Be concise and focused on a specific type of information or functionality
• Cover different angles, domains, or perspectives relevant to the user's request
• Be distinctive enough to retrieve different sets of relevant APIs
• Avoid repeating what you've already searched for (check history carefully)
Output in JSON format:
```json
["tool_description_1", "tool_description_2", "tool_description_3",...]
```
Example tool descriptions:
• "Academic database for peer-reviewed research papers in computer science"
• "News platform covering recent developments in artificial intelligence"
• "Technical documentation and code repository for open-source projects"
• "Financial data provider for market trends and company analysis"
—
Option B: If you decide to PROVIDE FINAL ANSWER
Synthesize the information you've gathered into a comprehensive, well-structured response that directly addresses the user's request.
Your answer should:
• Be informative and complete, well-organized and easy to understand
• Cite or reference the information sources appropriately
• Directly address all aspects of the user's original request
• Avoid fabricating information not supported by the gathered data
Output in JSON format:
```json
{"final_answer": "Your comprehensive answer to the user's request based on the information gathered from previous searches."}
```
IMPORTANT REMINDERS
• Your output must be VALID JSON only - no additional text, explanations, or markdown
• Choose only ONE option: either continue searching OR provide final answer
• Do NOT include control characters in your JSON output

Prompt for Tool Selection & Query Formulation

You are an intelligent search agent tasked with helping users gather comprehensive information to their request through strategic use of search APIs.
WORKFLOW OVERVIEW
The process follows these steps in a single round:
1. Generate Tool Descriptions OR Final Answer - Decide if more information is needed; if yes, describe what types of search tools you need; if no, provide final answer
2. Retrieve Tool Subset - System retrieves relevant APIs based on descriptions
3. **Select Tools & Generate Queries** (CURRENT STEP) - Choose specific tools and create search queries
4. Execute Search - System collects information from selected sources
KNOWN INFORMATION
Now you are in the **round {round_num}** of the interaction.
The max number of rounds allowed is {max_rounds}.
User's Request:
{task}
History:
{history}

Available Search APIs (Retrieved in Step 2):
{tools}
Understanding the History:
• Each round shows: Generated Descriptions → Retrieved Tools → Selected Tools and Generated Queries → Collected Information
• "Generated Descriptions" are the tool descriptions you requested in that round
• "Retrieved Tools" are the APIs the system found matching your descriptions
• "Selected Tools and Generated Queries" are the specific tools you chose and the queries you created
• "Collected Information" summarizes what was found from those sources
TOOL SELECTION AND QUERY GENERATION GUIDELINES
1. API Selection:
• Select APIs from the available list above that best address the user's information needs
• Focus on covering gaps in previously collected information (check history)
• Avoid APIs already searched with similar queries in previous rounds
2. Query Generation:
• Create specific, concise search queries
• Target distinct aspects of the information need
• Ensure queries differ from those in previous rounds
OUTPUT FORMAT
You MUST output a valid JSON array containing objects for each selected API.
Each object must have:
• "name": The exact API name from the available list (must match exactly)
• "query": A search query string for that API
Output ONLY the JSON array, with no additional text, explanations, or markdown:
```json
[{"name": "API Name 1", "query": "query 1"}, {"name": "API Name 2", "query": "query 2"}, {"name": "API Name 3", "query": "query 3"}]
```

Prompt for Content Summarization

You are a helpful assistant to extract key information from content while reducing length.
User's Request:
{task}
Content:
{content}
Your task:
Extract key information according to user's request into a single, concise summary.
Your answer should be no more than {threshold} tokens.
Requirements:
• Focus on information that directly helps answer the user's request.
• Preserve key facts, findings, and specific details relevant to the task.
• Keep important numbers, dates, names, and technical terms.
• Remove redundancy, verbose explanations, and irrelevant content.
• Maintain logical flow and readability.
Output format should be JSON only with no additional text, explanations, or markdown:
```json
{"summary": "Your concise summary here."}
```

Prompt for Final Answer Evaluation

You are an expert evaluator assessing the quality of an AI search agent's final answer.
User Question: {task}
Reference Answer: {reference_answer}
Key Points to Check:
{key_points}
AI Answer: {final_answer}
Evaluation Instructions
Key Points Hit Analysis
First, evaluate which key points from the reference answer are covered in the AI Answer. For each key point, determine if it is:

- **HIT**: The AI answer contains this information (may be paraphrased but factually equivalent)
- **MISS**: The AI answer does not contain this information or contradicts it
Dimension Scores
Evaluate the AI Answer based on the following 3 dimensions.
Assign a score from 1 to 10 for each dimension using the key points hit ratio as a reference.
1. **Completeness (1-10)**
- Does the answer fully address the user's question?
- Does it include all key information points found in the Reference Answer?
- Consider the key points hit ratio as a major factor
2. **Accuracy (1-10)**
- Is the information factually correct and consistent with the Reference Answer?
- Are there any contradictions or hallucinations?
- Penalize for any key points that are contradicted
3. **Relevance & Conciseness (1-10)**
- Is the answer directly focused on the user's intent?
- Is it well-structured and free of irrelevant fluff?
Output the scores in JSON format ONLY, with no additional text or markdown:
```json
{"completeness": <integer 1-10>, "accuracy": <integer 1-10>, "relevance": <integer 1-10>}
```