

# WebAggregator: Enhancing Compositional Reasoning Capabilities of Deep Research Agent Foundation Models

Rui Wang<sup>†</sup>, Ce Zhang, Jun-Yu Ma, Jianshu Zhang, Hongru Wang<sup>†</sup>, Yi Chen, Boyang Xue<sup>†</sup>, Tianqing Fang<sup>‡\*</sup>, Zhisong Zhang<sup>‡</sup>, Hongming Zhang<sup>‡</sup>, Haitao Mi<sup>‡</sup>, Dong Yu<sup>‡</sup>, Kam-Fai Wong<sup>†\*</sup>

<sup>†</sup>MoE Lab, The Chinese University of Hong Kong <sup>‡</sup>Tencent AI Lab

<https://github.com/Tencent/CognitiveKernel-Pro>

<https://github.com/Tencent/WebAggregator>

## Abstract

The hallmark of Deep Research agents lies in compositional reasoning, the capacity to aggregate distributed, heterogeneous information into coherent logical insights. However, current agentic systems are often retrieval-heavy but reasoning-light, where success is predominantly determined by simple entity-seeking rather than the multi-step aggregation of scattered evidence. To address this, we propose a data synthesis pipeline **WebAggregator**, designed to shift the agentic paradigm from retrieval-centric to compositional aggregation. Our approach first employs *Proactive Explorer* to collect interconnected knowledge, then *Compositional Logic Proposer* to weave knowledge into complex questions using over 12 composition guidelines derived from a rigorous deconstruction of the Deep Research problem setting. By leveraging 10K verifiable QA pairs grounded on 50K websites, we curate a high-quality SFT dataset via rejection sampling. Fine-tuning on this corpus fundamentally transforms agent behavior, fostering deliberate composition reasoning and reduced tool redundancy. The resulting WebAggregator-32B surpasses GPT-4.1 and matches Claude-3.7-Sonnet on GAIA, WebWalkerQA, and Xbench. To address the lack of benchmarks that emphasize both reasoning and retrieval, we introduce the WebAggregatorQA testbed, which reveals that even with perfect retrieval, top-tier models still underperformed. These results demonstrate that compositional reasoning, not retrieval, is the true performance ceiling for next-generation research agents.

## 1 Introduction

Deep Research agent systems (OpenAI, 2025; Monica.Im, 2025) aim to automate scientific discovery using foundational large language models (LLMs).

\*Correspondence to Rui [ruiwangnlp@outlook.com](mailto:ruiwangnlp@outlook.com), Tianqing [fangtq229@gmail.com](mailto:fangtq229@gmail.com), Kam-Fai [kfwong@cuhk.edu.hk](mailto:kfwong@cuhk.edu.hk)

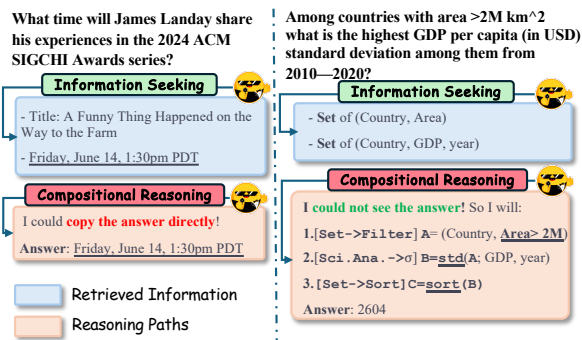


Figure 1: Comparison between WebWalkerQA (left) (Wu et al., 2025b) and WebAggregatorQA (right, ours). WebAggregatorQA necessitates both *Information Seeking* (IS) and intensive *Compositional Reasoning* load, which reveals a critical finding: *accurate retrieval could not guarantee task success*.

Achieving this requires not merely effective *information seeking*, grounding responses with accurate knowledge, but more critically, *compositional reasoning*: the ability to integrate distributed materials into coherent insights by decoding underlying logical structures (Bereiter and Scardamalia, 1987; Flower and Hayes, 1981; Mialon et al., 2023).

However, current resources fall short of fostering this capability. Traditional multi-hop datasets (Yang et al., 2018) offer only shallow logical hops, while recent web agent benchmarks (Shi et al., 2025; Wu et al., 2025b) tend to prioritize entity-seeking puzzles on static pages. There are two fundamental gaps: (1) insufficient environmental complexity, such as the dynamic content (e.g., JavaScript) and files essential for research; and (2) an imbalance in task difficulty, which relies on search effort rather than reasoning depth, as shown in Figure 1 & 2.

Specifically, 30.79% of WebWalkerQA (Deng et al., 2023) and 43.2% of TaskCraft (Shi et al., 2025) tasks only require simple entity localization. This observation aligns with other resources (Wu et al., 2025a; Tao et al., 2025), suggesting that ex-

Resource	IS	CR	Train set	Information Source	#Dom.
GAIA (Mialon et al.)	✓	✓	N/A	Text, Figure, Audio, File, Dynamic Elements	-
BrowseComp (Wei et al.)	✓	✗	N/A	Text	9
WebWalker (Wu et al.)	✓	✗	Y	Text	4
TaskCraft (Shi et al.)	✓	✗	Y	Text, PDF, Figure	8
WebShaper (Tao et al.)	✓	✗	N	Text	11
<b>WebAggregatorQA (Ours)</b>	✓	✓	Y	Text, Figure, File, Dynamic Elements	12

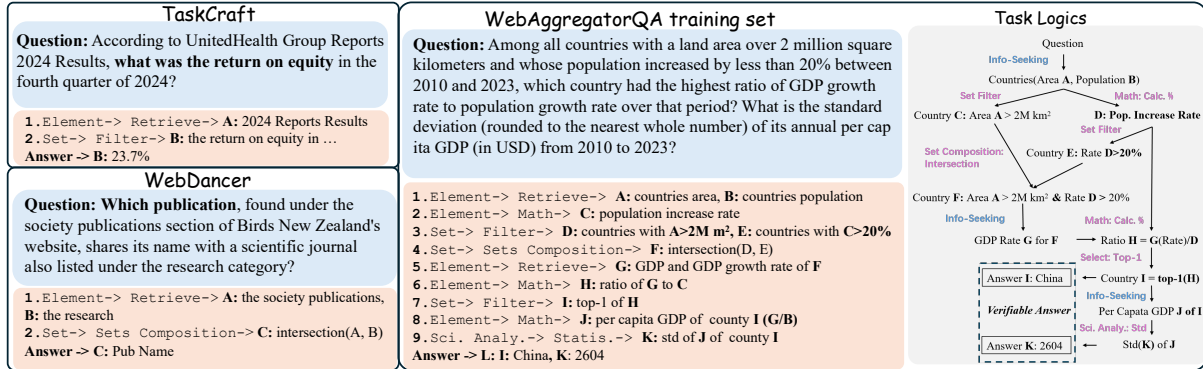


Figure 2: (Top) The table illustrates a feature comparison between our benchmark and existing resources. It highlights our dataset’s unique focus on **Compositional Reasoning (CR)** as a progression beyond traditional **Information-Seeking (IS)** needs. (Bottom) Examples comparison of the simple logic in prior works with the complex compositional reasoning demanded by WebAggregatorQA.

isting tasks may not fully test high-level reasoning. This tendency encourages agents to rely on frequent tool execution rather than necessary reasoning steps (Section 4.2), leaving compositional reasoning as a critical but underexplored frontier.

To bridge these gaps, we introduce **WebAggregator**, an automated framework designed to generate verifiable training data that pushes agents beyond information seeking towards deep compositional reasoning (Figure 3). The workflow consists of two phases: the *Proactive Explorer* and the *Compositional Logic Proposer*. Equipped with a comprehensive toolkit, spanning search, static parsing, dynamic interaction, file processing, and visual perception, the agent is capable of navigating in realistic web environments. *Proactive Explorer* initiates exploration from anchor URLs to autonomously gather interconnected resources. Subsequently, the *Compositional Logic Proposer* synthesizes complex tasks that explicitly necessitate multi-hop aggregation operations across these collected materials. Drawing on prior multi-hop reasoning studies (Chang et al., 2022; Yang et al., 2018; Talmor et al., 2021; Wu et al., 2025c; Wong et al., 2025), this module employs a rigorous taxonomy categorized into Element, Set, Scientific Analysis, and Temporal Reasoning, which are further refined into 12 granular subtypes (Table 4) to tailor the system to the analytical depth required by Deep Research.

We utilize WebAggregator to construct WebAg-

gregatorQA, a dataset of approximately 10K QA pairs designed to enforce rigorous compositional reasoning. As illustrated in Figure 2, WebAggregatorQA are designed to be non-trivial. Even with all necessary references provided, an agent must perform genuine compositional reasoning to derive the answer, rather than relying on direct extraction. Our rigorous quality control ensures the resulting tasks demand genuine compositional reasoning rather than shallow pattern matching. By leveraging rejection sampling to train the WebAggregator model family, we demonstrate the high effectiveness of this training corpus. Extensive experiments on GAIA, WebWalkerQA, and XBench show that WebAggregator-8B significantly surpasses GPT-4.1, while the 32B variant outperforms all strong baselines. Furthermore, fine-tuning on WebAggregatorQA fundamentally improves agent behavior by encouraging deeper reasoning steps and clearly reducing redundant tool usage, leading to robust performance gains.

The manually annotated WebAggregatorQA test set serves as a challenging benchmark, where even Claude-3.7-Sonnet achieves only 28.3%. Crucially, our error analysis reveals a critical limitation: *accurate reference retrieval does not guarantee task success*. Even with the full reference context provided, Claude-3.7-Sonnet scores only 42.1%, highlighting severe challenges in compositional reasoning. Our further analysis attributes this failure to gaps

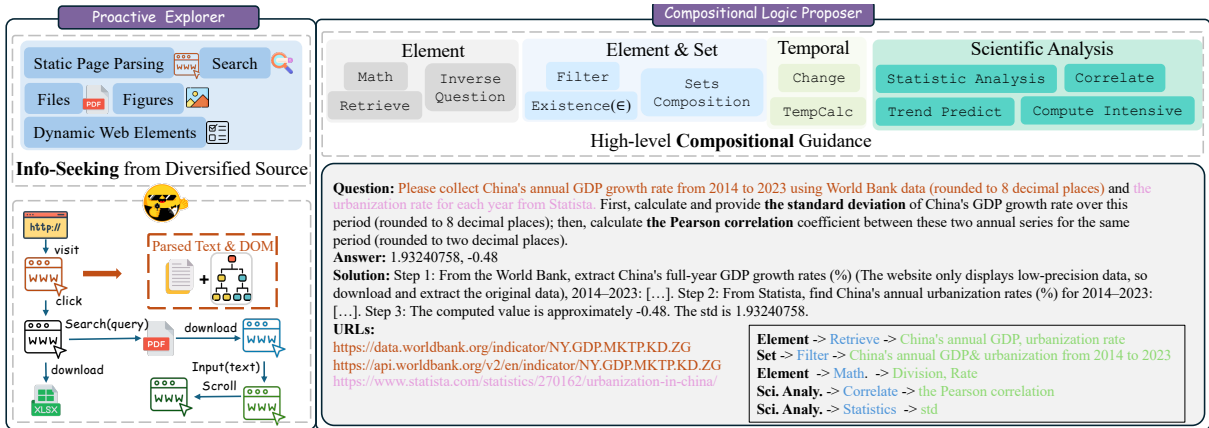


Figure 3: **Pipeline of WebAggregator for WebAggregatorQA construction.** (1) Proactive Explorer gathers comprehensive information by interacting with the web environment through tools (more details in Figure 10). (2) Compositional Logic Proposer synthesizes QA pairs based on the collected knowledge by instantiating the high-level composition guidance into concrete operations, e.g., *Statistic Analysis*  $\rightarrow$  *standard deviation*.

in advanced agentic capabilities, such as ineffective self-refinement, and faulty information utilization. These findings underscore the enduring value of WebAggregatorQA as a testbed, suggesting that future progress must transcend retrieval-centric paradigms to foster agents capable of the rigorous compositional reasoning required for complex tasks.

## 2 WebAggregator

We aim to automatically synthesize scalable, challenging QA pairs grounded in real web resources, requiring *minimal human involvement*. Unlike fact-lookup puzzles (Wei et al., 2025; Wu et al., 2025b), these tasks demand both complex retrieval and compositional reasoning. We propose an automatically verifiable pipeline, WebAggregator (Figure 3), that frames synthesis as an agentic task:<sup>1</sup> starting from anchor URLs, **Proactive Explorer** harvests interconnected information from diversified sources, which is formulated by **Compositional Logic Proposer** into reasoning-intensive QAs, which then undergo rigorous automated quality control.

### 2.1 Anchor URL Collection

By selecting seed URLs across multiple domains, we can effectively regulate the dataset’s domain distribution, thereby enabling effective distribution balance. We sampled 5,000 topic-diverse queries from QA and scientific datasets (Yang et al., 2018; Jin et al., 2019; Trivedi et al., 2022) and retrieved URLs via Google Search, resulting in over 160,000 anchor links from 11+ domains (Figure 4).

<sup>1</sup>Agent details in Appendix A.

### 2.2 WebAggregator: From Exploration to Composition

In this section, we introduce our approach to synthesizing target QA pairs by framing *task synthesis as a specialized form of web-agent tasks*. The agent is given a task prompt consisting of two components: Proactive Explorer and Compositional Logic Proposal, enabling it to complete the task construction in an end-to-end manner.

#### 2.2.1 Proactive Explorer

**Proactive Explorer** aims to collect diverse information as the foundation for task construction by exploration on the real web. During this phase (Figure 3), the agent is prompted to start from a single anchor URL and employ various tools to navigate across web pages to discover any unknown but relevant information. Interactions include navigating heterogeneous content types such as text, files, and images, as well as dynamic element interactions, which are still challenging for even state-of-the-art vision language models (Zhang et al., 2025). To control task difficulty and ensure the comprehensiveness of the knowledge scope, a minimum number of web page visits is enforced (Appendix B.3).

This unconstrained exploration proves highly effective at sourcing diverse information. Analysis of 5,296 exploration trajectories (Figure 4) reveals significant interaction diversity. Every trajectory leverages at least three foundational tools: Search, Visit, and ScreenShot. Crucially, a majority of trajectories employ a wider toolkit: 48.36% use 4 distinct tools, and 28.55% use 5, while 13.41% engage with 6 or more. This breadth of interaction enriches the knowledge pool and directly translates into more

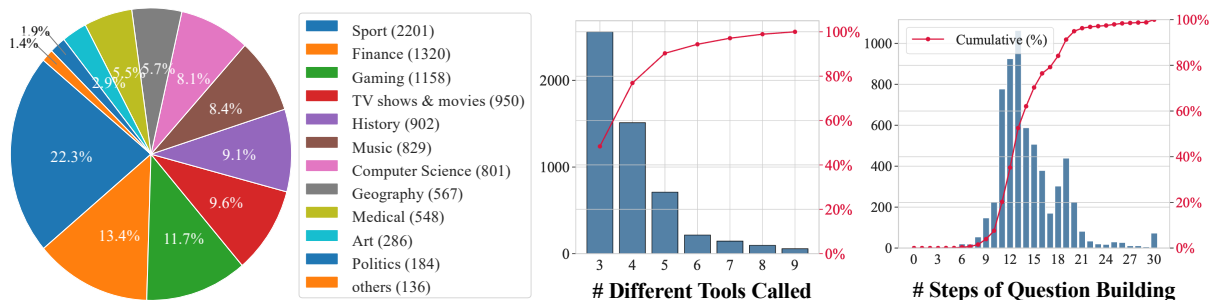


Figure 4: The distribution of domains in WebAggregatorQA, tasks categorized by the number of different tools involved during construction, and steps (an action-observation round) in data synthesis.

complex downstream tasks. For instance, questions derived from file-based information inherently test an agent’s file-processing capabilities.

## 2.2.2 Compositional Logic Proposer

**Compositional Logic Proposer** transforms raw, unstructured information into sophisticated reasoning challenges. To systematically generate diverse and challenging reasoning chains, we developed a taxonomy of logical operations inspired by prior work on multi-hop and compositional QA (Sen et al., 2022; Talmor et al., 2021; Wu et al., 2025c; Fang et al., 2024; Krishna et al., 2025). As shown in Figure 3, this taxonomy is organized into four high-level categories: Element, Set, Scientific Analysis, and Temporal Reasoning, which are further refined into 12 granular subtypes. Element and Set operations are the basis of compositional reasoning behaviors among knowledge snippets, such as *mathematical calculation* among elements and *Filtering* among sets. While Scientific Analysis and Temporal Reasoning are advanced applications of them, involving computation-intensive and data science tasks, to reflect real but challenging user cases.

Crucially, these 12 subtypes act as high-level conceptual guides within the prompt, not as rigid templates. They instruct the Proposer to instantiate these patterns into concrete, context-specific questions. For example, *Element*->*mathematical calculation* guide can be instantiated as addition, subtraction, or ratio calculation, depending on the available data. This flexible, guided approach ensures the generation of novel and diverse reasoning challenges (examples provided in Table 4).

## 2.2.3 Quality Control

**QA Alignment Checking** We implement a two-stage verification process to ensure data integrity. First, the Proposer employs a self-refinement tool with a checklist to revise questions (Appendix B.5). Second, another agent reviews the data by solv-

ing it, verifying reference URLs to guarantee strict alignment between questions, answers, and sources (Appendix B.4). This rigorous post-hoc filtering removes 11.72% of the initial samples.

**Diversity Constraint** We actively balance the dataset across domains and compositional types.<sup>2</sup> First, we normalize the distribution of anchor URL domains annotated by GPT-4.1 (Figure 4). Second, we dynamically adjust prompts to upweight under-represented composition operations (e.g., calculating average) identified via preliminary analysis. As shown in Figure 8, this strategy successfully spawns diverse low-level operations tailored to high-level guidance, such as intersection for Set tasks and table processing for Scientific Analysis.

## 2.2.4 Trajectory Sampling

After the task synthesis, we collect the trajectory that completes these tasks. We utilize the agent based on GPT-4.1 with SmolAgents and collect the plain text trajectories. To ensure the quality of the collected trajectories, we conduct a further rejection sampling and finally collected 6,184 trajectories for the foundation model training. (1) **Correctness**: We retain those trajectories with correct predictions according to the reference answers. (2) **Format**: Data with format errors (e.g., undefined tool name or parameters) is filtered out. (3) **Exception Handling**: Anomalies in observations (e.g., page failures) are kept to improve the model’s generalization, since similar situations would occur in the real web.

## 2.2.5 Statistics of WebAggregatorQA

WebAggregatorQA comprises 9,883 tasks (with 200 reserved for testing), covering 54,064 unique URLs across 12 domains. Figure 4 shows the distribution of domains and steps for QA construction trajectories. Most QA pairs are constructed with around 15 steps, demonstrating that the gener-

<sup>2</sup>See Appendix C for contamination analysis.

Methods	GAIA-text				WebAggregatorQA			
	Level-1	Level-2	Level-3	Average	Level-1	Level-2	Level-3	Average
<i>Non-Agentive</i>								
GPT-4.1	10.3	13.5	8.3	11.7	15.4	4.0	2.8	5.6
Claude-3.7-sonnet	35.9	17.3	0.0	22.3	18.5	5.1	2.8	6.8
Qwen2.5-7B	12.8	3.8	0.0	6.8	4.2	1.0	0.0	1.3
Qwen2.5-32B	20.5	9.6	8.3	13.6	4.2	1.0	0.0	1.3
Qwen3-8B	12.8	3.8	0.0	6.8	4.2	1.0	2.8	1.9
Qwen3-32B	17.9	3.8	0.0	8.7	8.3	1.0	0.0	1.9
<i>Zero-shot Foundations</i>								
GPT-4.1	51.3	44.2	16.7	43.7	62.4	22.2	11.1	25.8
GPT-5	71.8	57.7	33.3	60.2	58.3	24.5	<b>22.2</b>	<b>28.9</b>
Claude-3.7-sonnet	<b>74.4</b>	<b>55.8</b>	<b>33.3</b>	<b>60.2</b>	<b>66.7</b>	<b>25.3</b>	11.1	28.3
Qwen2.5-7B	23.1	15.4	0.0	16.5	27.3	3.4	2.8	6.3
Qwen2.5-32B	46.1	21.2	0.0	28.2	25.0	10.1	5.6	11.3
Qwen3-8B	33.3	11.5	0.0	18.4	30.8	5.1	5.6	9.4
Qwen3-32B	48.7	40.4	16.7	40.8	45.8	10.1	5.6	14.5
<i>Fine-tuned Foundations</i>								
<i>WebThinker</i>								
Qwen2.5-32B	56.4	50.0	16.7	48.5				
<i>WebDancer</i>								
Qwen2.5-7B	41.0	30.7	0.0	31.0				
Qwen2.5-32B	46.1	44.2	8.3	40.7				
<i>WebSailor</i>								
Qwen2.5-7B	-	-	-	37.9				
Qwen2.5-32B	-	-	-	53.2				
<i>WebShaper</i>								
Qwen2.5-32B	61.5	53.8	16.7	52.2				
<i>CogKernal-Pro</i>								
Qwen3-8B	56.4	42.3	8.3	43.7				
<i>WebAggregator</i>								
Qwen2.5-7B	53.8	30.8	16.7	40.8	37.5	11.1	8.3	14.5
- pass@3	74.4	63.5	25.0	63.1	54.2	22.2	19.4	26.4
Qwen2.5-32B	66.7	44.2	<b>33.3</b>	51.5	54.2	15.2	<b>11.1</b>	20.1
- pass@3	79.5	67.3	50.0	69.9	70.8	22.2	19.4	28.9
Qwen3-8B	61.5	34.6	16.7	42.7	54.2	11.1	5.6	16.4
- pass@3	82.1	53.8	33.3	62.1	62.4	21.2	11.1	25.2
Qwen3-32B	<b>69.2</b>	<b>55.8</b>	16.7	<b>56.3</b>	<b>62.4</b>	<b>24.2</b>	8.3	<b>26.4</b>
- pass@3	79.5	67.3	50.0	69.9	66.7	35.4	13.9	35.2

Table 1: The Pass@1 & Pass@3 performance of agents on GAIA-text and WebAggregatorQA. The best performance of different settings is in bold.

ated data points are not hastily created from only a few reasoning steps, thus avoiding overly simplistic questions.

### 2.3 WebAggregatorQA testset for Compositional Reasoning Evaluation

Existing benchmarks (Wu et al., 2025b; Wei et al., 2025) primarily assess information-seeking, often neglecting compositional reasoning. To address this critical gap, we introduce the test set of WebAggregatorQA, which is meticulously annotated to evaluate both retrieval and reasoning capabilities.

**Annotation Details** We uniformly split 200 tasks as seeds from WebAggregatorQA across different domains to ensure high task diversity. We aim to ensure every question is unambiguous, with the reference answer correct, other candidates are verifiably incorrect. First, human annotators solve the

QA pairs, then review the QA and references to eliminate ambiguities by adding clear constraints. To further enhance sample reliability and reduce bias from the solver’s perspective, this process is repeated twice: tasks are solved, ambiguities identified, and revisions made by annotators. In the final cross-validation stage, each question was answered by at least two annotators, yielding 155 consistently aligned samples. The other 4 samples were abandoned by annotators due to difficulty, but their references and questions were verified to ensure data quality and thus were retained. More details are shown in Appendix B.2.

This yielded 159 samples, including those in text and multimodal, categorized by difficulty into Level 1 (24), Level 2 (99), and Level 3 (36). Each sample contains a question, reference answer, solution, and supporting URLs. A text example is shown in

Model	WebWalkerQA	XBench
GPT-4.1	39.9	37.0
<i>WebDancer</i>		
Qwen2.5-7B	36.0	-
Qwen2.5-32B	38.4	-
<i>WebSailor</i>		
Qwen2.5-7B	-	34.3
Qwen2.5-32B	-	53.3
<i>WebAggregator</i>		
Qwen2.5-7B	44.7	37.0
Qwen2.5-32B	<b>45.7</b>	56.0
Qwen3-8B	41.2	40.0
Qwen3-32B	43.5	<b>58.0</b>

Table 2: Performance on WebWalkerQA and XBench.

Figure 2 and a multimodal one is in Figure 11.

### 3 Experiments

#### 3.1 Experimental Setups

**Models and Benchmarks** We construct the WebAggregator models by SFT varied Qwen2.5 (Yang et al., 2024) and Qwen3 (Yang et al., 2025) models on the training set of WebAggregatorQA. We evaluate the baselines and our methods on the GAIA-text (Mialon et al., 2023) following Li et al. (2025a,b), XBench (Chen et al., 2025), WebWalkerQA (Wu et al., 2025b), and WebAggregatorQA.

**Training Configs** We formalize the trajectory we sampled as (question,  $a_1, o_1, \dots, a_n, o_n$ , answer).  $a_i$  stands for the *action* code the agent generated to perform tool calling, and  $o_i$  is the *observation* returned by the web environment. The question and observations are masked during training.

**Baselines and Metrics** We mainly compare WebAggregator with three types of prior works. **a.** Non-agentic foundation models that answer questions using their internal knowledge. **b.** Zero-shot foundation models initialized as agents via the SmolAgents framework. **c.** Strong fine-tuned foundation models: WebThinker (Li et al., 2025b), WebDancer (Wu et al., 2025a), CognitiveKernel-Pro (Fang et al., 2025b), WebSailor (Li et al., 2025a) and WebShaper (Tao et al., 2025). We report the pass@1 and pass@3 scores evaluated by GPT-4.1 following previous works (Wu et al., 2025a).

#### 3.2 Main Results

##### Effects of WebAggregatorQA Training Set

The experiment results are shown in Table 1 and Table 2. In the zero-shot setting, the closed-sourced models surpass Qwen models on benchmarks. However, after tuning on WebAggregatorQA, Qwen models exhibit clear and steady improvements on all

of the benchmarks and approach the performance of these strong baselines. Specifically, the WebAggregator based on Qwen2.5-32B and Qwen3-32B surpasses most of the strong baselines, including GPT-4.1 and WebShaper. The pass@3 performance of WebAggregator-32B achieves 69.9% on GAIA-text. These observations prove the quality of WebAggregatorQA and the effectiveness of our data construction paradigm.

**Difficulty of WebAggregatorQA Test Set** The complex aggregation needs in WebAggregatorQA pose a new challenge for current agent systems. GPT-4.1-powered SmolAgents attain 43.7% accuracy on GAIA-text but drop to 25.8% on WebAggregatorQA. Claude-3.7-sonnet shows a similar decline. Furthermore, the performance gap between Claude and GPT-4.1 is smaller on WebAggregatorQA than on GAIA-text. This suggests that for the harder questions in WebAggregatorQA, neither model can solve them effectively, which leads to the reduced gap. These results and our following analysis (Section 4) highlight **the substantial gap between current agent capabilities and the demands of compositional reasoning needed** for multi-hop web tasks.

**Data Efficiency and Small Foundations** The construction of WebAggregatorQA also enables highly efficient model training. By fine-tuning Qwen3-8B on small subsets of 500 and 1,200 trajectories, we achieve 36.9% and 38.83% accuracy on GAIA-text, respectively, demonstrating that even limited high-quality data from WebAggregatorQA can yield significant performance gains. This efficiency translates to superior performance in smaller foundation models; for instance, WebAggregator-8B achieves parity with GPT-4.1 on GAIA-text and significantly outperforms previous baselines on WebWalkerQA and XBench (Table 2). Despite these gains, a performance gap persists between 8B and 32B models on the more complex WebAggregatorQA tasks, suggesting that while small foundations are increasingly viable and cost-effective, breaking through the performance bottleneck of small foundations on hard tasks remains a vital direction.

## 4 Analysis

### 4.1 Requirements for Solving WebAggregatorQA

To better understand the challenges in WebAggregatorQA, Figure 5 illustrates the distribution of infor-

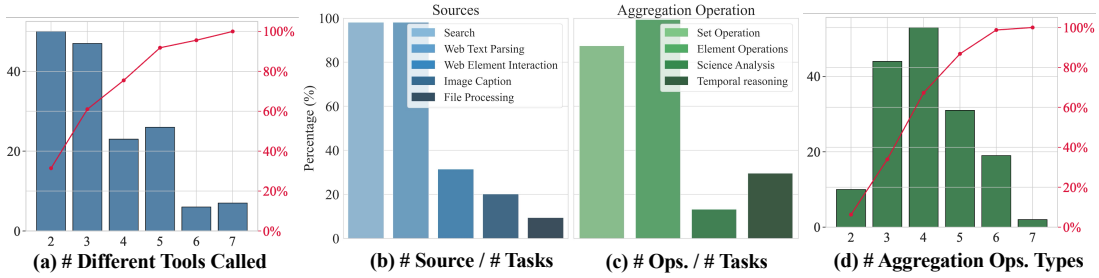


Figure 5: Distributions of tasks required different numbers of tools (a) and aggregation operations (d). Proportion of information source (b) and aggregation operations (c) that are needed across tasks.

mation sources and the compositional operations required for task completion.

**Diversified Information Source Reliance** Our analysis reveals that tasks in WebAggregatorQA necessitate access to at least Search and Web Text. Furthermore, these tasks frequently require information from Files, indicating a multi-source dependency rather than reliance on a single medium. Successfully addressing WebAggregatorQA demands advanced web-browsing capabilities to retrieve knowledge. The benchmark proves exceptionally challenging for models constrained by internal knowledge alone. Even strong base models like Claude-3.7 and GPT-4.1 solve fewer than 7% of the questions.

**Diversified Composition Requirements** We observe that the compositional reasoning requirements of WebAggregatorQA challenge the agent systems. Figure 5 illustrates that the solving of these tasks requires diversified composition operations, and many of them contain multiple operations. We conduct further detailed analysis in Section 4.2.

## 4.2 Challenges from Compositional Reasoning

In this section, we scrutinize the behaviors of research agents and highlight a compound challenge: compositional reasoning on the retrieved information to derive the final answer.

**Success in Seeking, Fail in Reasoning** We further investigate the impact of compositional reasoning by analyzing a specific failure mode: **perfect reference retrieval but fails the task**. In Table 3, although accuracy improves when all reference URLs are visited, the performance gain remains modest. This suggests that while successful information retrieval is a prerequisite, the subsequent reasoning burden remains a significant bottleneck. Notably, Qwen3-32B shows a sharp gap between its strong retrieval and weak accuracy, highlighting a deficit in compositional reasoning. WebAggregator-

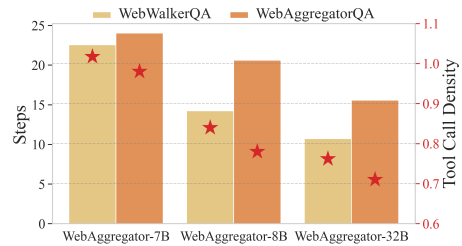


Figure 6: Steps and tool use density of two models across test sets. More results are shown in Table 5.

Model	Counts	Acc.
GPT-4.1	30	33.3
Claude	38	42.1
Qwen3-32B	22	9.7
WebAggregator-32B	28	35.7

Table 3: Counts and accuracy of trajectories that visited all reference URLs and obtained the gold knowledge.

32B, however, significantly closes this gap, reaching performance levels comparable to top-tier baselines. Still, the lack of a high score across all models, even with full information, underscores that the primary ceiling remains the complex composition logic.

## Reasoning-Intensive Nature of Composition

We evaluate the impact of compositional reasoning on agents by analyzing their tool usage patterns across WebAggregatorQA and WebWalkerQA in Figure 6. We define tool call density as the number of tool calls per step. We observe that while tasks in WebAggregatorQA require more total steps to complete, the tool call density is notably lower. This pattern suggests that in WebAggregatorQA, models rely more heavily on reasoning steps to execute compositional reasoning, enabling deeper synthesis and analysis of existing information, rather than predominantly invoking tools to acquire new external knowledge.

## 4.3 Behavior Evolution of WebAggregator

**Evolution of Tool Usage Behaviors** To understand how fine-tuning on WebAggregatorQA alters

agent behavior, we analyzed their action sequences and performed a semantic analysis on 500 randomly sampled steps per model. Following the classification by GPT-5, we categorize internal Thought processes into: (1) **Procedural & Planning (Type A)**: routine instructions (e.g., "I will search for X") and state updates; (2) **Compositional Reasoning (Type B)**: deep operations including information synthesis, logical deduction, and error correction. Our findings, illustrated in Table 5 and Figure 9, reveal three key trends:

First, **WebAggregators exhibit reduced tool dependency alongside enhanced reasoning density**. As shown in Table 5, the Reasoning Ratio (Type B) increased across all scales (e.g., from 18.6% to 26.2% for the 8B model), while Tool Call Density markedly declined. This inverse correlation refutes the verbosity hypothesis; rather than generating procedural fluff, the model prioritizes internal logical synthesis over mechanical tool invocation.

Second, **model scaling further amplifies evidence aggregation capacity**. Post-tuning, despite fewer tool calls, total token consumption rose by 12–17%, a growth driven exclusively by Type B reasoning. This indicates that larger models (32B) leverage deeper internal synthesis, integrating fragmented data into coherent insights, to resolve complex queries with higher efficiency and fewer environment interactions.

Finally, **agent behavior shifts toward a more diverse and balanced operational distribution**. Prior to tuning, tool-calling patterns were concentrated and simplistic (e.g., 40% of trajectories relied on only two tools, as seen in Figure 9). WebAggregatorQA facilitates a transition from a retrieval-centric paradigm to a reasoning-heavy one, prioritizing multi-step aggregation of scattered evidence.

**Evolution on Composition Reasoning Capabilities** We investigate how model performance changes across different compositional logic types. Logic labels are labeled by an LLM via analyzing the reference solutions, consistent with the dataset construction method (Figure 3). We compare Qwen series models before and after fine-tuning. We compute the Phi Coefficient ( $\psi$ ), measuring correlation between logic type presence and answer correctness. More negative  $\psi$  indicates greater task difficulty. Figure 7 reveals that computation-intensive logic, particularly Scientific Analysis and Set Operations, remains the key challenge.

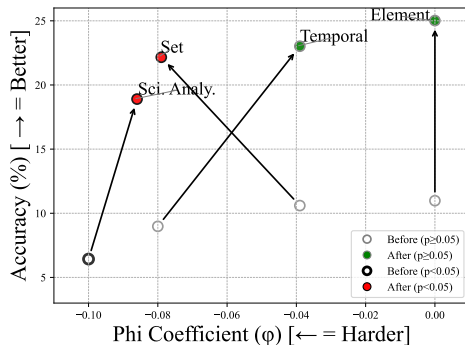


Figure 7: Phi Coefficients ( $\psi$ ) by composition types across models before and after tuning on WebAggregatorQA.

#### 4.4 Screening of Failure Modes

To elucidate the inherent challenges of WebAggregatorQA, we conduct a qualitative analysis of failure trajectories. We manually examine 50 representative cases from the models listed in Table 3 where all references were retrieved, comprising: 28 instances from WebAggregator-32B, 6 each from Claude and GPT-4.1, and 10 additional failure cases. We reveal a taxonomy of three failure categories:

**Information Seeking Failures** These errors preclude access to valid evidence. A primary issue is **Source Location Failure**, where agents misled by initial uncertainty or sub-optimal search queries, fail to locate gold references and settle for incomplete secondary sources.

**Agent Foundation Capability Failures** Operational errors also derail tasks. Common issues include *Invalid Tool Calls* (e.g., malformed queries) and *Exceeding Max Context*, often resulting from inefficient planning or error-correction loops.

**Compositional Reasoning Failures** This category covers errors during the composition of retrieved information. (1) **Faulty Composition**: agents misinterpret evidence or apply flawed reasoning, such as making unsupported assumptions, despite accessing the correct sources (Figure D.1). (2) **Error Accumulation**: minor inaccuracies (e.g., premature rounding) compound across steps to yield a significantly incorrect result (Figure D.3). (3) **Ineffective Refinement**: a critical failure where agents fail to self-correct, proceeding down the erroneous logical path while being unable to revise previous mistakes (Figure D.2).

The above failures are rarely isolated. As shown in Figure D.2, an initial *Faulty Composition* can propagate errors downstream, exacerbated by *Ineffective Refinement*. This cascade consumes the context window (*Exceeding Max Steps*), increasing

the probability of hallucinations or timeouts.

## 5 Related Work

**Resources for Deep Research Agents** Robust information retrieval and precise question answering form the bedrock of deep research agents. Consequently, developing multi-hop datasets for training agent foundation models (Tongyi, 2025; Qiao et al., 2025) necessitates sophisticated tool-use and reasoning, making manual construction challenging. Existing datasets like HotpotQA (Yang et al., 2018; Trivedi et al., 2022) lack the complexity of authentic web interactions. While some works synthesize request-action pairs (Xu et al., 2025; Chen et al., 2024; He et al., 2024), they often lack goal-oriented depth. Recent methods construct tasks from logical flows over knowledge snippets (Wu et al., 2025a; Li et al., 2025a; Shi et al., 2025; Xia et al., 2025), such as using KGs from *offline* pages (Shi et al., 2025; Wu et al., 2025a) or entity expansion (Xia et al., 2025; Tao et al., 2025). However, these remain limited by static content and often neglect knowledge composition (Figure 3), where complexity stems more from reasoning than entity tracing.

**Benchmarking Deep Research Agents** Most existing benchmarks prioritize information-seeking and reference filtering (Wu et al., 2025b; Wei et al., 2025), while few (Li et al., 2025c) assess compositional reasoning. Krishna et al. (2025) evaluates aggregation but is confined to Wikipedia, and Wong et al. (2025) focuses on long sequences of simple actions. GAIA (Mialon et al., 2023), though most relevant, is becoming saturated by recent agents (Fang et al., 2025b; Qiu et al., 2025). This underscores the need for challenging benchmarks that jointly evaluate information-seeking and complex reasoning.

### Training and Optimization for Research Agents

Recent advancements focus on sophisticated training and test-time optimization. Besides standard rejection-sampling based SFT, which is adopted in most agent post-training works (Tongyi, 2025; Fang et al., 2025b; Team et al., 2025), there are more advanced techniques leveraging reinforcement learning and test-time self-evolving. In addition to simply applying the training QA pairs as RL input queries, self-synthesized queries can be useful for providing process-level supervision signals for better credit assignment (Fang et al., 2025b; Feng et al., 2025; Wang et al., 2026; Li et al., 2026; Fang et al., 2026; Fu et al., 2026). More specifi-

cally, execution errors during tool interaction can also be converted into corrective supervision signals within the RL loop to improve multi-turn tool robustness (Zhang et al., 2026). Regarding test-time self-evolving (Fang et al., 2025a), synthesized data could help building agent failure taxonomy and corresponding rubrics for better evaluating the generation of actions (Wan et al., 2026).

## 6 Conclusion

In this work, we introduce WebAggregator, a framework designed to bridge the gap between information seeking and compositional reasoning in Deep Research agents. Through the construction of WebAggregatorQA and the WebAggregator models, we demonstrate that verifiable, reasoning-intensive training data significantly boosts agent performance across diverse benchmarks. Crucially, our analysis reveals a fundamental bottleneck: *accurate retrieval does not guarantee task success*, as current models struggle to integrate distributed information. By mitigating these deficits through targeted fine-tuning, WebAggregator paves the way for future research to transcend retrieval-centric paradigms and move closer to autonomous scientific discovery.

### Limitations

To ensure a fair and controlled comparison, all evaluation protocols and experimental details are strictly aligned with established methodologies (Wu et al., 2025a; Li et al., 2025b; Wu et al., 2025b; Tao et al., 2025), precluding any interference from prior knowledge. Following these standards, we evaluated WebAggregator across several well-recognized benchmarks (Mialon et al., 2023; Wu et al., 2025b; Chen et al., 2025) to demonstrate its generalization capabilities. Furthermore, we conducted a rigorous data contamination analysis on WebAggregatorQA; as detailed in Appendix C, the results confirm zero overlap with existing benchmarks. Given our automated pipeline, we implemented strict post-construction filtering and utilized a rejection sampling procedure during training to eliminate misaligned QA pairs, thereby guaranteeing high data quality. While the data construction of WebAggregator requires environmental interaction and execution time, it remains highly cost-effective compared to human annotation, which often demands several hours of expert labor per task. The code, data and models will be open-sourced.

## References

- Carl Bereiter and Marlene Scardamalia. 1987. *The psychology of written composition*.
- Yingshan Chang, Guihong Cao, Mridu Narang, Jianfeng Gao, Hisami Suzuki, and Yonatan Bisk. 2022. *Webqa: Multihop and multimodal QA*. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 16474–16483. IEEE.
- Kaiyuan Chen, Yixin Ren, Yang Liu, Xiaobo Hu, Haotong Tian, Tianbao Xie, Fangfu Liu, Haoye Zhang, Hongzhang Liu, Yuan Gong, Chen Sun, Han Hou, Hui Yang, James Pan, Jianan Lou, Jiayi Mao, Jizheng Liu, Jinpeng Li, Kangyi Liu, and 14 others. 2025. *xbench: Tracking agents productivity scaling with profession-aligned real-world evaluations*. *Preprint*, arXiv:2506.13651.
- Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. 2024. *Agent-FLAN: Designing data and methods of effective agent tuning for large language models*. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 9354–9366, Bangkok, Thailand. Association for Computational Linguistics.
- Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. *Mind2web: Towards a generalist agent for the web*. *Preprint*, arXiv:2306.06070.
- Tianqing Fang, Zeming Chen, Yangqiu Song, and Antoine Bosselut. 2024. *Complex reasoning over logical queries on commonsense knowledge graphs*. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 11365–11384. Association for Computational Linguistics.
- Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. 2025a. *Webevolver: Enhancing web agent self-improvement with co-evolving world model*. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8970–8986.
- Tianqing Fang, Zhisong Zhang, Xiaoyang Wang, Rui Wang, Can Qin, Yuxuan Wan, Jun-Yu Ma, Ce Zhang, Jiaqi Chen, Xiyun Li, Hongming Zhang, Haitao Mi, and Dong Yu. 2025b. *Cognitive kernel-pro: A framework for deep research agents and agent foundation models training*. *Preprint*, arXiv:2508.00414.
- Yangyi Fang, Jiaye Lin, Xiaoliang Fu, Cong Qin, Haolin Shi, Chang Liu, and Peilin Zhao. 2026. *Proximity-based multi-turn optimization: Practical credit assignment for llm agent training*. *arXiv preprint arXiv:2602.19225*.
- Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. 2025. *Group-in-group policy optimization for llm agent training*. *arXiv preprint arXiv:2505.10978*.
- Linda S. Flower and J. R. Hayes. 1981. *A cognitive process theory of writing*. *College Composition & Communication*.
- Xiaoliang Fu, Jiaye Lin, Yangyi Fang, Binbin Zheng, Chaowen Hu, Zekai Shao, Cong Qin, Lu Pan, Ke Zeng, and Xunliang Cai. 2026. *Maspo: Unifying gradient utilization, probability mass, and signal reliability for robust and sample-efficient llm reasoning*. *arXiv preprint arXiv:2602.17550*.
- Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. *WebVoyager: Building an end-to-end web agent with large multimodal models*. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6864–6890, Bangkok, Thailand. Association for Computational Linguistics.
- Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. *Pubmedqa: A dataset for biomedical research question answering*. *Preprint*, arXiv:1909.06146.
- Satyapriya Krishna, Kalpesh Krishna, Anhad Mohanaray, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, and Manaal Faruqi. 2025. *Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation*. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 4745–4759, Albuquerque, New Mexico. Association for Computational Linguistics.
- Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, Weizhou Shen, Junkai Zhang, Dingchu Zhang, Xixi Wu, Yong Jiang, Ming Yan, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025a. *WebSailor: Navigating Superhuman Reasoning for Web Agent*. *arXiv preprint*. ArXiv:2507.02592 [cs].
- Mukai Li, Qingcheng Zeng, Tianqing Fang, Zhenwen Liang, Linfeng Song, Qi Liu, Haitao Mi, and Dong Yu. 2026. *Verified critical step optimization for llm agents*. *arXiv preprint arXiv:2602.03412*.
- Xiaoxi Li, Jiajie Jin, Guanting Dong, Hongjin Qian, Yutao Zhu, Yongkang Wu, Ji-Rong Wen, and Zhicheng Dou. 2025b. *Webthinker: Empowering large reasoning models with deep research capability*. *Preprint*, arXiv:2504.21776.
- Zijian Li, Xin Guan, Bo Zhang, Shen Huang, Houquan Zhou, Shaopeng Lai, Ming Yan, Yong Jiang, Pengjun Xie, Fei Huang, Jun Zhang, and Jingren Zhou. 2025c. *Webweaver: Structuring web-scale evidence with dynamic outlines for open-ended deep research*. *Preprint*, arXiv:2509.13312.

- Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2023. *Gaia: a benchmark for general ai assistants*. *Preprint*, arXiv:2311.12983.
- Monica.Im. 2025. *Manus ai*. Technical report, Monica.Im.
- OpenAI. 2025. *Introducing deep research | OpenAI*.
- Zile Qiao, Guoxin Chen, Xuanzhong Chen, Donglei Yu, Wenbiao Yin, Xinyu Wang, Zhen Zhang, Baixuan Li, Hui Feng Yin, Kuan Li, Rui Min, Minpeng Liao, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025. *Webresearcher: Unleashing unbounded reasoning capability in long-horizon agents*. *Preprint*, arXiv:2509.13309.
- Jiahao Qiu, Xuan Qi, Tongcheng Zhang, Xinzhe Juan, Jiacheng Guo, Yifu Lu, Yimin Wang, Zixin Yao, Qihan Ren, Xun Jiang, Xing Zhou, Dongrui Liu, Ling Yang, Yue Wu, Kaixuan Huang, Shilong Liu, Hongru Wang, and Mengdi Wang. 2025. *Alita: Generalist agent enabling scalable agentic reasoning with minimal predefinition and maximal self-evolution*. *Preprint*, arXiv:2505.20286.
- Aymeric Roucher, Albert Villanova del Moral, Thomas Wolf, Leandro von Werra, and Erik Kaunismäki. 2025. ‘smolagents’: a smol library to build great agentic systems. <https://github.com/huggingface/smolagents>.
- Priyanka Sen, Alham Fikri Aji, and Amir Saffari. 2022. *Mintaka: A complex, natural, and multilingual dataset for end-to-end question answering*. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1604–1619, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.
- Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, King Zhu, Minghao Yang, Jian Yang, Ge Zhang, Jiaheng Liu, Changwang Zhang, Jun Wang, Yuchen Eleanor Jiang, and Wangchunshu Zhou. 2025. *TaskCraft: Automated Generation of Agentic Tasks*. *arXiv preprint*. ArXiv:2506.10055 [cs].
- Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hananeh Hajishirzi, and Jonathan Berant. 2021. *MULTIMODALQA: COMPLEX QUESTION ANSWERING OVER TEXT, TABLES AND IMAGES*.
- Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025. *Webshaper: Agentic data synthesizing via information-seeking formalization*. *Preprint*, arXiv:2507.15061.
- MiroMind Team, Song Bai, Lidong Bing, Carson Chen, Guanzheng Chen, Yuntao Chen, Zhe Chen, Ziyi Chen, Jifeng Dai, Xuan Dong, and 1 others. 2025. *Miro-thinker: Pushing the performance boundaries of open-source research agents via model, context, and interactive scaling*. *arXiv preprint arXiv:2511.11793*.
- Tongyi. 2025. *Tongyi-deepresearch*. <https://github.com/Alibaba-NLP/DeepResearch>.
- Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. *MuSiQue: Multihop questions via single-hop question composition*. *Transactions of the Association for Computational Linguistics*.
- Yuxuan Wan, Tianqing Fang, Zaitang Li, Yintong Huo, Wenxuan Wang, Haitao Mi, Dong Yu, and Michael R Lyu. 2026. *Inference-time scaling of verification: Self-evolving deep research agents via test-time rubric-guided verification*. *arXiv preprint arXiv:2601.15808*.
- Tianyi Wang, Yixia Li, Long Li, Yibiao Chen, Shaohan Huang, Yun Chen, Peng Li, Yang Liu, and Guanhua Chen. 2026. *Sppo: Sequence-level ppo for long-horizon reasoning tasks*. *Preprint*, arXiv:2604.08865.
- Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. 2025. *Browsecomp: A simple yet challenging benchmark for browsing agents*. *Preprint*, arXiv:2504.12516.
- Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, Wenhao Huang, Yang Wang, and Ke Wang. 2025. *Widesearch: Benchmarking agentic broad information-seeking*. *Preprint*, arXiv:2508.07999.
- Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025a. *WebDancer: Towards Autonomous Information Seeking Agency*. *arXiv preprint*. ArXiv:2505.22648 [cs].
- Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, and Fei Huang. 2025b. *Webwalker: Benchmarking llms in web traversal*. *Preprint*, arXiv:2501.07572.
- Xianjie Wu, Jian Yang, Linzheng Chai, Ge Zhang, Jiaheng Liu, Xeron Du, Di Liang, Daixin Shu, Xi'anfu Cheng, Tianzhen Sun, and 1 others. 2025c. *Tablebench: A comprehensive and complex benchmark for table question answering*. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25497–25506.
- Ziyi Xia, Kun Luo, Hongjin Qian, and Zheng Liu. 2025. *Open data synthesis for deep research*. *Preprint*, arXiv:2509.00375.

- Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. 2025. [Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials](#). *Preprint*, arXiv:2412.09605.
- 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, and 41 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.
- An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, and 22 others. 2024. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*.
- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. [HotpotQA: A dataset for diverse, explainable multi-hop question answering](#). In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.
- Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. [React: Synergizing reasoning and acting in language models](#). *Preprint*, arXiv:2210.03629.
- Jianshu Zhang, Dongyu Yao, Renjie Pi, Paul Pu Liang, and Yi R. Fung. 2025. [Vlm2-bench: A closer look at how well vlms implicitly link explicit matching visual cues](#). In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025, Vienna, Austria, July 27 - August 1, 2025*, pages 7510–7545. Association for Computational Linguistics.
- Zhiwei Zhang, Fei Zhao, Rui Wang, Zezhong Wang, Bin Liang, Jiakang Wang, Yao Hu, Shaosheng Cao, and Kam-Fai Wong. 2026. [Robust tool use via fission-grpo: Learning to recover from execution errors](#). *arXiv preprint arXiv:2601.15625*.

## A Agent Structure

First, we introduce our agent framework. User instructions often require accessing diverse information on the web—plain text, images, or files—some needing specific interactions like button clicks. Thus, the agent must go beyond extracting static page text. We categorize tools by information source: **Search** (Search), **Static Page Parsing** (Visit, StrFind), **Dynamic Interaction** (Input, Click, Scroll, Goback), **File Processing** (FileRead), and **Image Captioning** (ImageCaption).

Our implementation utilizes a code-based ReAct (Yao et al., 2023) agent, built on SmolAgents (Roucher et al., 2025), which outputs natural language thoughts, Python-coded actions, and receives code log as environment observations. Each task runs within a fixed 30-step budget, where a step includes an agent action and its observation. We extend the *opendeepresearch* SmolAgents instance with DOM parsing for finer web interactions (see Table 6). This web agent effectively handles tasks requiring information from multiple sources, as summarized in Table 2.

## B More Details for WebAggregatorQA

### B.1 More Explanations of Quality Control

**QA Alignment-based Filtering** To improve data quality, we implemented a two-stage refinement process for task construction. The first stage uses a self-refinement tool with a checklist (Appendix B.5) to quickly verify and revise questions during creation. Items meeting all criteria are accepted and outputted; those that don’t are revised based on feedback until they comply. The second stage involves a data checking agent that thoroughly reviews all reference URLs to ensure alignment of question, answer, and sources (Appendix B.4). About 11.72% of the data were filtered out for failing to meet these standards.

**Diversity Constraint** We ensure dataset diversity by balancing domain and aggregation operation distributions. First, we annotated anchor URL domains with GPT-4.1 and sampled data to achieve a more balanced domain distribution, shown in Figure 4. Second, we analyzed compositional reasoning types using GPT-4.1 to identify high- and low-level operations (e.g., *math subtraction*). Although not perfectly reliable without solving the questions, common operations like *calculating av-*

*erage GDP* can be accurately detected. We then adjusted prompts to emphasize rare aggregation types, increasing their sample frequency. Figure 8 shows a word cloud illustrating how different high-level aggregation tasks yield diverse specific operations, such as *intersection* for set operations and *table processing* for Science Analysis.

**Data Leakage Avoidance** During proactive web exploration, agents may download and parse existing datasets. To prevent data contamination, we created a website keyword blacklist. Pages matching the blacklist or containing identified datasets were excluded from retrieval and subsequent model evaluation to ensure the fairness of the evaluation.

### B.2 Testset Annotation

#### B.2.1 Data Collection and Verification

The construction of the test set includes the seed tasks collection, several rounds of revision, and cross-validation procedures. Three human annotators are involved, all of them with at least a bachelor’s degree. Each sample requires an average of 3 hours of annotation work, and the whole procedure lasts for more than 4 weeks of part-time work (4 hours a day).

**Seed Tasks Collection** A single human annotator, even if highly specialized in one domain, faces inherent limitations in generating diverse and comprehensive samples across multiple fields. To address this, we engage multiple annotators to revise 200 topic-diverse tasks, uniformly selected from WebAggregatorQA based on their domain labels. These carefully revised seed examples help ensure that the test set attains the desired diversity.

**Task Revision Principles** Our initial analysis, consistent with prior work (Wei et al., 2025), shows that due to high uncertainty in the web environment and an answer-to-question task design, questions are generally well-structured but often lack unique ground truths. While this ambiguity may be tolerable for training, it is unacceptable for testing. Thus, our key revision principle is to ensure each question is unambiguous and has exactly one correct answer.

- **QA** (1) Ensure clarity in the question statements. All claims must be explicitly stated, and if multiple sources of evidence exist, additional constraints should be provided in the question to avoid ambiguity. (2) The reference answer must be the sole feasible and correct one to the question.
- **Reference** Reference information, including URLs and solutions, is vital to the revision pro-

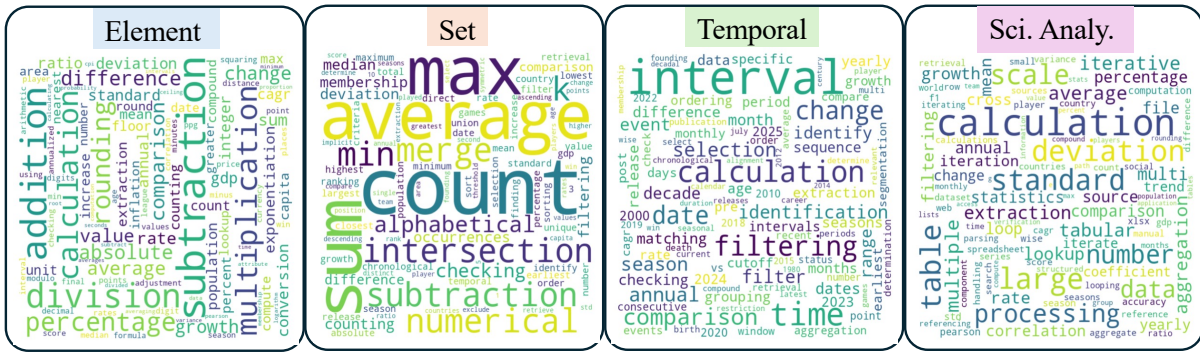


Figure 8: Word cloud of compositional reasoning types extracted from the constructed tasks. High-level logic guidelines are derived into concrete logics.

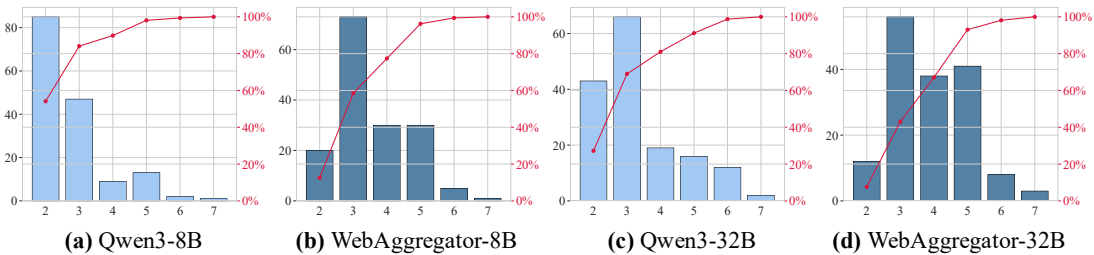


Figure 9: The distribution of unique tools used per trajectory, comparing the model before and after fine-tuning. After fine-tuning, WebAggregator utilizes a more diverse and balanced set of tools within each trajectory. In trajectories that involve only two tools, these are consistently Search and Visit.

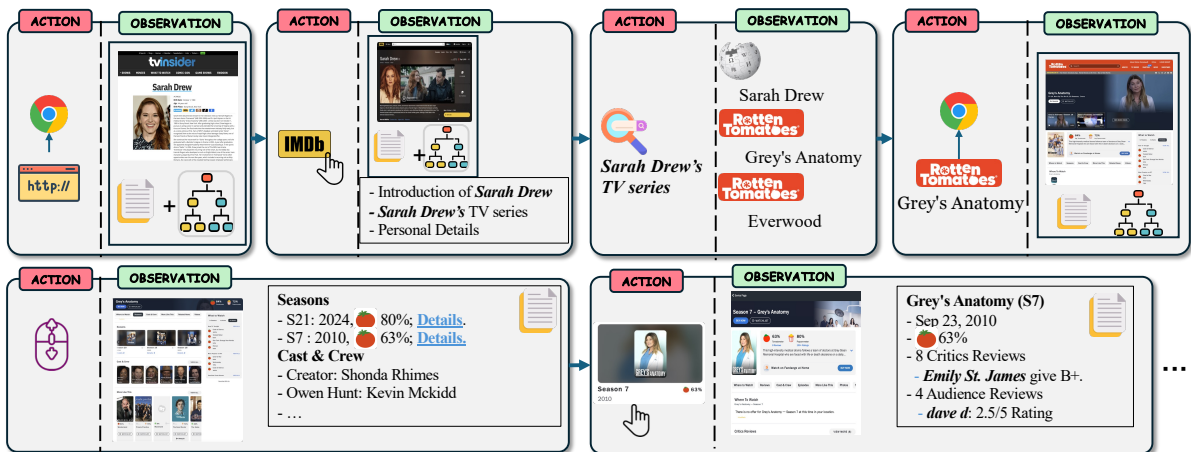


Figure 10: A running example of **Proactive Web Exploring**: a greater variety of interactions fosters a richer diversity of knowledge and introduces more challenges throughout the process, e.g., questions built from file knowledge also test the file-processing abilities of responding agents.

Operations	Questions
<i>Element Operations</i>	<i>Aggregate elements/entities, e.g., numbers, times, names(<math>x, y</math>).</i>
Retrieve ( $x$ )	In Amor: A Recipe for Building Adaptable ... , <b>what hourly pay (in USD) is for the hired NLP expert?</b>
Inverse ( $x$ )	<b>Which American actor won the Academy Award for XXX in the 1990s</b> released their first solo studio album the greatest number of years after their Oscar win?
Math ( $x, y$ )	Among Benedict, Robert Downey, and ..., for the person's first appearance in a Marvel Cinematic Universe film corresponded to <b>the highest ROI</b> for their debut Marvel movie, what is the ROI (three decimals)?
<i>Set Operations</i>	<i>Aggregate elements (<math>x</math>) and sets (<math>Y, Z, \dots</math>).</i>
Filter( $Y$ )	Among the countries that won at least 15 gold medals at the London 2012 Summer Olympics, what is the HDI of the country that had <b>the third highest</b> per-capita GDP (in USD) in 2012?
Existence( $x, Y$ )	For the college that had the most players selected overall in the 2023 NBA Draft, <b>how many</b> of its draftees were picked in the first round?
Compose( $Y, Z$ )	According to the WorldPopulationReview, how many cities among <b>the top 100 most populous cities in 2025</b> have experienced <b>a population decrease compared to 2024?</b>
<i>Temporal Reasoning</i>	<i>Reasoning or calculation related with time.</i>
Change	Between 1990 and 2022, which country had the third largest average <b>annual percentage increase</b> in nominal GDP?
TempCalc	Among Robert De Niro, Al Pacino, Christopher Walken, and Jessica Lange, who has the <b>longest interval</b> between their first and most recent Academy Award nominations without winning, and <b>what is the length of that span in years?</b>
<i>Science Analysis</i>	<i>Coding is a must to improve efficiency or precision for computation-intensive tasks.</i>
CompIntensive( $X$ )	What is <b>the average closure price of Apple.inc from Jan. 2024 to Oct. 2024?</b>
Predict ( $x_1, \dots, x_n$ )	KFF published an article on abortion in Women's Health Policy on Feb 27, 2025. Using single exponential smoothing and MSE, search for the optimal alpha (0.01-0.99, step=0.01) based on the historical data, the MSE loss, and <b>use the alpha to estimate the next data point.</b>
Statistic ( $x_1, \dots, x_n$ )	Among all Cleveland Cavaliers head coaches who have won at least one playoff game with the team, what is <b>the standard deviation</b> of their playoff win percentages?
Correlate ( $X, Y$ )	Between the 2012 to 2022 NBA seasons, what is <b>the Pearson correlation coefficient</b> between Damian Lillard's season average points per game and the Portland Trail Blazers' regular season win percentage?

Table 4: Several representative examples in WebAggregatorQA of information aggregation operations are presented. Note that the operations here are *high-level guidance* that could be derived into a diversified, specific form, rather than low-level constraints.  $x$  means an element or knowledge snippet,  $X$  denotes a list of knowledge snippets that fulfill a certain condition.

cess. When these reference URLs and solutions are properly validated, the quality and reliability of the questions and answers are assured. Accordingly, annotators are required to: (1) verify the reliability of URLs, ensuring they originate from authoritative and reputable sources; (2) ensure consistency: the evidence remains stable and not prone to variation across different websites, contexts, or over time; (3) confirm the fidelity of URLs: each provided reference URL directly and substantively supports the question. Those pages that have a strict CAPTCHA will be replaced with more stable ones. Then the questions and answers are revised accordingly.

The second principle is to **increase task complexity** from the same two perspectives: complex compositional reasoning and diversified information sources. We provide annotators with the compositional reasoning guidance and encourage them to incorporate more reasoning steps into the ques-

tions to enhance their difficulty. They are also advised to leverage various information forms beyond plain webpage text. The answer should not be directly found on the web page.

**Verification** We utilize agents to assist the human validation procedure. Initially, a GPT-4.1-powered agent attempts to solve the questions, facilitating identification of potential ambiguities from the solver's perspective within a realistic web environment. Subsequently, human annotators review the agent's responses, detect any misunderstandings, and revise the questions, solutions, and answers accordingly. This cycle is repeated once more to ensure comprehensive disambiguation.

In the final stage, human annotators independently solve the tasks for cross-validation purposes. Samples that fail to achieve consensus with previously annotated answers are excluded. The independent annotation achieved agreement with 155

Model	% TC Steps (Step-wise)	TC Density (TC/Step)	Total TC (Sum)	Latency (s) (Avg.)	Reasoning % (Type B)
Qwen3-8B	90.27%	1.52	2004	301.1	18.6%
<b>+WebAggregatorQA(Ours)</b>	<b>86.37%</b>	<b>0.78</b>	<b>928</b>	372.7	<b>26.2%</b>
Qwen3-32B	91.55%	1.12	1337	225.4	31.4%
<b>+WebAggregatorQA(Ours)</b>	<b>82.14%</b>	<b>0.71</b>	<b>1248</b>	306.6	<b>32.6%</b>

Table 5: Behavioral analysis on the WebAggregatorQA test set. WebAggregatorQA training leads to a significant reduction in Tool Call (TC) density and total volume, while increasing the proportion of **Type B (Reasoning)** steps. This suggests the models shift from mechanical retrieval-centric patterns to more deliberate, compositional deliberation.

**Question:** Between the game's release month and three months afterward, what was the average monthly percentage change in peak concurrent players for a superhero PvP shooter game released in 2024 (from steamcharts.com)? The game lost the highest average number of players in a month before July 2025. At the beginning of that month, there was a Twitch Drops event where watching for 30 minutes rewarded an item featuring two characters. Before July 2025, how many times was the character on the right buffed and nerfed, respectively? Rounded to two decimals.

**Answer:** -2.78; buffed: 3, nerfed: 7

**Solution:** 1. Get peak concurrent players for Marvel Rivals for Dec 2024, Jan 2025, Feb 2025, and Mar 2025 from statistics (...).  
3. Average the percentages: -2.78%. 4. The month this game lost the highest number of average players is March 2025.  
5. The first twitch drops shown in marvelrivals.com is 20250217.  
The 30mins reward is a spray, the human torch on the right of the spray.  
6. Finally, we can count the ↓ (nerf), 7, and ↑ (buffed), 3.

**URLs:**  
<https://steamcharts.com/app/2767030>  
[https://www.marvelrivals.com/announcements/20250217/40955\\_1212338.html](https://www.marvelrivals.com/announcements/20250217/40955_1212338.html)  
[https://marvelrivals.fandom.com/wiki/Human\\_Torch#Balance\\_Changes](https://marvelrivals.fandom.com/wiki/Human_Torch#Balance_Changes)

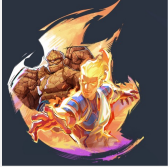


Figure 11: A multimodal sample from the test set of WebAggregatorQA. To solve this task, the agent must extract information from the image to obtain clues for the next step. Since the image is not provided with the question, the agent is required to locate the relevant picture independently.

Action	Observation
Search(Query)	Search results
Visit(URL)	Web Text & DOM
StrFind(Query)	Matched str in text
Input(str, tbox_id)	Web Text & DOM
Click(button_id)	Web Text & DOM
Scroll(Pixels)	Web Text & DOM
Goback()	Web Text & DOM
FileRead(Path)	File content
Screenshot(Path)	Capture screen
ImageCaption(Path)	Image description

Table 6: Action and Observation space.

out of 159 references, with 4 tasks omitted due to excessive complexity, thereby validating the quality of the references.

## B.3 Data Construction Prompt

### *PART-I: Proactive Explorer*

URL:  
{URL}

---

#### Task Overview

- Create a challenging multi-hop question based on the given URL and related information.
- Ensure the quality of the answer when providing a reference answer! Please calculate and verify the reference answer before giving the final data.
- The question should be written in the SAME language as the website content.

---

#### 1. Information Gathering

- Start by thoroughly exploring the given URL and its description.
- Visit and browse at least **{least\_visits}** different websites to collect diverse and relevant information.
- Avoid relying solely on simple search engine queries or Wikipedia. Instead, actively browse, jump between pages, and record your navigation steps and key findings.
- After each browsing action, briefly document what you did and the important information you discovered.

---

#### 2. Question Design

Formulate a **multi-hop question** that **MUST** requires reasoning across multiple sources. The answer should **not** be obtainable by a simple search or from a single page.

The question should be:

- Challenging but natural and concise, as if a real user is seeking to learn or solve a puzzle. Avoid unnatural or arbitrary questions such as summing unrelated numbers.
  - e.g., year \* (number of countries of china) is unacceptable!
- Self-contained.
  - Illustrated with essential clues that guide the respondent to locate the information without explicitly naming the sources or searching queries. The clues must be necessary but precise, avoiding overly broad candidates.
    - BAD EXAMPLES: Some China city has,... (NOT self-contained! Specify the city by specifying the name or providing clues.)
- Based on specific details from at least 5 to 8 different web pages.
- Reflective of the domain's characteristics (e.g., medical: functions, gaming: guidance, players, chemistry, math, puzzles).

## PART2: Compositional Logic Proposer

### 3. Composition Reasoning Operations (Mandatory)

Incorporate at least one of the following reasoning operations in your question:

- Science Analysis

- > Statistical Analysis

- Analyze data from web pages, you may use, but not limited to: calculating the mean, variance, or standard deviation within a specified time period. Some good examples:

1. What is the median winnings for drivers who have driven a Chevrolet car?
2. Which category exhibits the most consistent growth rate across the 5-year period, and what is the average annual percentage increase for that category?
3. Can you calculate the standard deviation of the average comprehension scores across A, B, and C?

- > Correlation Analysis

1. What is the Pearson correlation coefficient (to two decimal places) between China's average annual temperature and its CO2 emissions per capita over the same period?

- > Trend Forecasting

- Based on historical data, predict future data points. Any algorithm can be used, such as linear regression, polynomial regression, logistic regression, EMA, etc. REMEMBER: Clearly specify the basis for prediction to ensure a unique answer. Some good examples:

1. Considering the historical data from 1961 to 1967, what could be the forecasted points of Suzuki in the 50cc and 125cc classes for the upcoming years? Use the average growth rate or the most recent 5-year growth rate for prediction.

2. KFF published an article on abortion in Women's Health Policy on Feb 27, 2025. Using single exponential smoothing and MSE, search for the optimal alpha (0.01-0.99) based on the historical data, the MSE loss, and use the alpha to estimate the next data point.

- > General Computation Intensive Tasks

- Batch Data Analysis Requires Intensive Computation. The need to retrieve and process large lists of numbers makes coding ESSENTIAL.

1. What is the average closure price of Apple.inc from Sep. 2024 to Oct. 2024?
2. Across all NBA seasons where Manu Ginobili's Player Efficiency Rating (PER) exceeded 20 in the regular season, what was the average number of regular season wins by his team?

- > Other Tasks

- Complex Algorithm with high Complexity: Try to design problems that require coding to reduce time complexity.

---

- Element-wise operations

- > Calculation

- Selecting specific elements, performing mathematical operations between elements, e.g., probability, calculation.

- Examples:

1. What is the sum of A's speed and B's speed?
2. By how much does C's value exceed D's value?
3. What is the difference between the population of city X and city Y?

- > Inverse Question

- Formalized as an inverse question about certain information. Avoid direct listing; use indirect clues framed as questions. Ensure your phrasing uniquely identifies the subject without ambiguity.

- Examples:

1. Instead of "Tom is a singer from New York, who was born on 11 Nov 2024, he...", you can use "for the single from New York, who was born on 11 Nov 2024, he...".
2. In June 2022, researchers from Huddersfield University published a paper on the application of YOLO in agriculture. My research primarily focuses on ...

---

- List/Set-wise operations

- > sorting (alphabetical, numerical, top-K), sum, average, counting, intersection, subtraction, merging. Examples:

1. Which is the shortest among XXX?
2. What is the average length of YYY?
3. How many items appear in both set A and set B?
4. What is the total number of Z across all categories?

---

- Element-Set operations

- > checking membership or counting occurrences. Examples:

1. Is element E part of the top 10 ranked items?
2. Exclude all names that were born in 1984 from ...

3. Between 2012 and 2021, was the rate of increase in China's average annual temperature higher or lower than the global average?

4. On the same day that a landmark house on South Main Street in Coeymans Landing, New York, rich with local history, built in the late 1830s, officially entered the National Register of Historic Places listing, how many places entered the list total?

-----  
Note: The numbers or elements used in these operations should be discoverable by reading the web content, not directly provided in the question.  
-----

#### 4. Answer Requirements

- > The answer MUST not be obtained directly from the retrieved text and MUST be derived through reasoning. > Short, Concise and easy to verify.
- > Stable over time (avoid dynamic or real-time data).
- > Of a clear entity type (e.g., person, number, date, place).

#### 5. Output Format

Output your final result in the following JSON format:

```
{
  "topic": "Brief description of the question's domain or topic",
  "question": "The constructed multi-hop question",
  "answer": "The answer X",
  "context": {
    "URLs": [ "url_1", "url_2", "url_3", "url_4", "url_5", ... ]
  }
}
```

## B.4 Prompt of Data Quality Checking Agent

### TASK DESCRIPTION OF DATA QUALITY CHECKING AGENT

{Composition Reasoning Operations Prompt}  
Evidence Checking

- URL Validity:** Verification that all URLs conform to proper syntax and resolve correctly without errors.
- Information Relevance:** Assessment of whether each URL contains information that is necessary and sufficient to address the research question.

Question Checking

- Self-Containment:** The extent to which the question is fully specified and comprehensible without requiring additional external context.
- Retrieval Necessity:** The degree to which answering the question necessitates consulting external sources, while avoiding excessive disclosure of information within the question itself.
- Composition Necessity:** The question must include at least three different aggregation operations, ensuring that the answer cannot be obtained through direct retrieval.
- Clarity:** The precision and unambiguity of the cues or references embedded in the question that facilitate accurate data retrieval. The clues will not lead to multiple feasible answers.
- Temporal Stability:** The property that the correct answer to the question remains consistent over time, unaffected by temporal changes (e.g., "Who was the immediate past president of the United States?").

Answer Quality Assessment

- Information Fidelity:** The extent to which all information presented in the reference answer is fully consistent with the URLs or other provided external information sources.
  - *Example of inconsistency:* The temperature retrieved from the reference URL is 37°C, whereas the solution states 35°C, resulting in an erroneous calculation of the average temperature.
- Ground Truth Validity:** The reference answer must accurately and unambiguously reflect the requirements of the question, conforming to information obtained from authoritative and reliable data sources.
  - The answer should be derived from recognized authoritative channels or verified databases.
  - Ensuring verifiability through reliable sources is especially important for questions involving numerical data, statistics, or other factual information.
  - *Example of invalid answer:* "The moon's distance from Earth is 100,000 km." This contradicts scientific consensus, which states the distance is approximately 384,400 km.
- Uniqueness and Unambiguity:** The reference answer should be uniquely correct, avoiding ambiguity or multiple plausible solutions.
  - Are there conflicting data from multiple sources that lead to multiple possible answers?
  - Are there precision conflicts between different data sources (e.g., 33.2 vs. 33.20987)?

-----  
Based on the above criteria, analyze the following data by solving it:  
Question: {}  
Answer: {}  
Evidence\_URLs: {}  
Please verify whether each item meets the standards.  
-----

Output Format

Return your analysis in the following JSON format:

```
```json
{
  "Evidence Passed": 1 or 0,
  "Question Passed": 1 or 0,
  "Answer Passed": 1 or 0,
  "Domain": "[USE ONLY ONE WORD OF THE FOLLOWING!] Gaming, Sport, TV shows & movies,
Computer Science, Art, History, Music, Geography, Politics, Finance, Medical, Law",
  "Compositional_Operation":
  {
    "type": ["Science Analysis Operations->Informations search->XLSX Processing of ...",
"Element-wise->Math->Addition", "Science Analysis Operations->Batch Data Processing->",
...]
  }
}
```
```

## B.5 Prompt of Intergrated Data Quality Checking Tool

### PROMPT OF EFFICIENT QUESTION CHECKING TOOL

{Composition Reasoning Operations Prompt}  
Question Checking

- Self-Containment:** The extent to which the question is fully specified and comprehensible without requiring additional external context.
- Retrieval Necessity:** The degree to which answering the question necessitates consulting external sources, while avoiding excessive disclosure of information within the question itself.
- Compositional Necessity:** The question must include at least three different aggregation operations, ensuring that the answer cannot be obtained through direct retrieval.
- Clarity:** The precision and unambiguity of the cues or references embedded in the question that facilitate accurate data retrieval. The clues will not lead to multiple feasible answers.
- Temporal Stability:** The property that the correct answer to the question remains consistent over time, unaffected by temporal changes (e.g., "Who was the immediate past president of the United States?").

-----  
Based on the above criteria, analyze the following data:

Question: {}

Answer: {}

Evidence\_URLs: {}

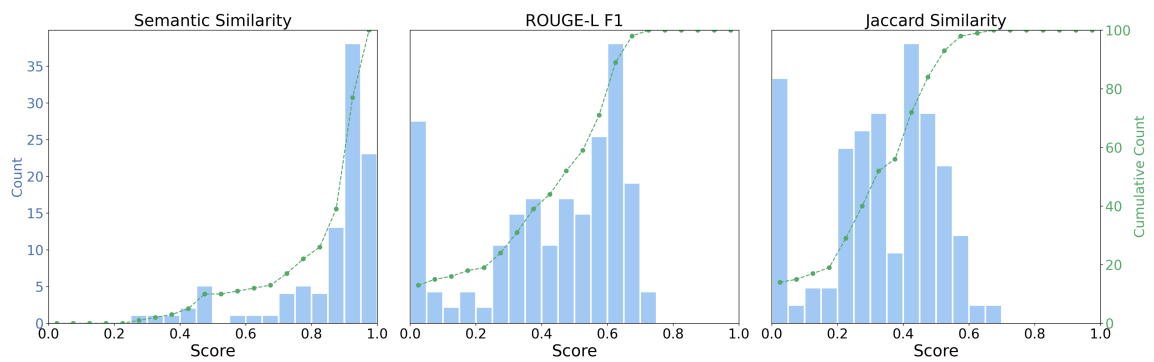
Please verify whether each item meets the standards and provide advice for improvements.

## C Data Contamination Analysis

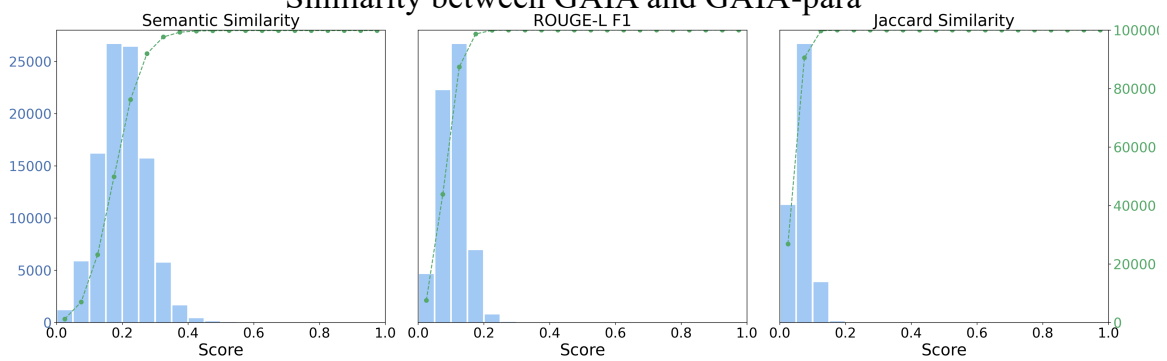
We investigate potential data contamination between WebAggregatorQA and GAIA through two similarity analyses, with the results presented in Figure 12. First, to establish a baseline, we prompt an LLM to paraphrase the GAIA dataset, creating a new version termed GAIA-para. We then measure the similarity between the original and paraphrased GAIA questions. Second, we assess the cross-dataset overlap by sampling 100,000 pairs from the Cartesian product of the WebAggregatorQA training set and GAIA and computing their similarity. For all comparisons, we employ three metrics: a semantic similarity score calculated by Qwen3-Embedding-8B, ROUGE-L, and Jaccard similarity.

Figure 12 reveals a stark contrast: while GAIA shows significant similarity with its paraphrased counterpart, its overlap with WebAggregatorQA is negligible. This provides strong evidence that **our WebAggregatorQA dataset is free from GAIA data contamination.**

## D Close Screening Agent Trajectories



### Similarity between GAIA and GAIA-para



### Similarity between GAIA and WebAggregatorQA

Figure 12: Data Contamination Experiment

## D.1 Example 1

### Example 1

#### User

Between 2014 and 2024, which year has the most new Trustees of the current Embry-Riddle Board of Trustees? And how many of the current Trustees received the Living Legend of Aviation?

#### Agent

First, I need to find the official Embry-Riddle Aeronautical University Board of Trustees page. This will likely contain the current list of Trustees and possibly historical data or links to archives. I will search for the Board of Trustees page and review its contents for the necessary information.

```
1 results = web_search(query="Embry-Riddle Aeronautical University Board of
2 Trustees")
3 print(results)
```

#### User

1. [Embry-Riddle Aeronautical University: Board of Trustees](https://trustees.erau.edu/)
2. [Meet the Trustees](https://trustees.erau.edu/trustees)
3. [Trustee Ken Dufour and Chairman Mori ...](https://news.erau.edu/headlines/trustee-ken-dufour-and-chairman-mori-hosseini-mark-milestones-on-embry-riddles-board)
4. ...

#### Agent

The top search result is the official Embry-Riddle Board of Trustees page: <https://trustees.erau.edu/>.

This page is likely to have a current list of Trustees and possibly links to historical data or news about new appointments.

I will visit this page and review its contents to gather the information needed, especially looking for sections about current Trustees, their appointment years, and any links to archives or news about new appointments.

```
1 board_page = visit_webpage(url="https://trustees.erau.edu/")
2
3 print(board_page)
```

#### User

...

### Agent

Since there is no unified list or page displaying the year each trustee became a trustee, I will open each trustee's profile one by one to check and confirm whether they have received the Living Legend of Aviation award.

```
1 # Visit the pages of the first ten trustees
2 urls=[
3 "https://trustees.erau.edu/trustees/michael-amalfitano",
4 "https://trustees.erau.edu/trustees/michael-amalfitano",
5 "https://trustees.erau.edu/trustees/leanne-caret",
6 ...
7 ]
8 print(len(urls)) # There are {20} pages
9 for url in urls[:10]:
10     print(visit_webpage(url))
```

### User

...

Mori Hosseini is the Chairman and CEO of ICI Homes, one of Florida's largest residential homebuilder/developers. Born in Iran, Mr. Hosseini was educated in London, England, ...

- Mori Hosseini (HonDoc '13; '78, '79, '82, DB)
- Member since **1999**
- Chair, Board of Trustees since 2014
- Chair, Committee on Trustees

...

### Agent

I will continue to access other pages and record the years they became members.

```
1 trustees=[
2 "Michael Amalfitano" :{
3     "member year": "2021",
4     "living legend": False
5 },
6
7 "John Amore" :{
8     "member year": "2010",
9     "living legend": False
10 },
11 ...
12 "Mori Hosseini" :{
13     "member year": "2014",
14     "living legend": False
15 },
16 ...
17 ]
18 for url in urls[10:]:
19     print(visit_webpage(url))
```

Error Analysis: Faulty Composition

Key information extraction error: 2014 v.s. 1999

### User

...

**Agent**

```
1 final_answer=(\{"year":2014, "number of awardees": 3 \})
```

Error Analysis: Faulty Composition

Faulty Composition while *Set->Filter->Counting*

**User**

END» answer: 2014, 3 # true answer: 2014&2018, 2

## D.2 Example 2

### Example 2

**User**

After one game's update introducing the Embrion and Old Bird, calculate the compound monthly CAGR of its average players from the update's release month until (but excluding) the next major patch's month, using steambase.io data. For the same period, calculate the CAGR of another game, which is a 4 player online co-op psychological horror game released in 2020. Present both game name and number as 'game: ±XY.ZT%`.

**Agent**

Step 1: Identify the game that received an update introducing "Embrion" and "Old Bird". I will use a web search to find which game this is and get the update's release month. After that, I will look for the next major patch's release month for that game.

```
1 search_query = 'game update introducing Embrion and Old Bird'
2
3 results = web_search(query=search_query)
4
5 print(results)
```

**User**

...

**Agent**

The update introducing Embrion and Old Bird is Version 50 of Lethal Company, released on April 13, 2024. I now need to find the release month of the next major patch after Version 50. I will search for "Lethal Company next major patch after Version 50" to determine this.

```
1 next_patch_search = web_search(query="Lethal Company next major patch
  after Version 50")
2
3 print(next_patch_search)
```

**User**

...

### Agent

The most relevant sources suggest that after Version 50, **the next major patch is speculated to be Version 70**, but there is no explicit mention of an official release date. Since the question specifies to use steambase.io data, I should check steambase.io for the version history or patch notes for Lethal Company to see if there is a clear next major patch after Version 50. If not, I might have to use the next version listed or clarify if there is a clear, next major patch. I will visit the steambase.io Lethal Company page and look for update/version history.

```
1 lethal_company_steambase = web_search(query="Lethal Company steambase.io")
2
3 print(lethal_company_steambase)
```

**Error Analysis: Faulty Composition**

The next major patch is v60, on August 17, 2024. But the agent takes a guess.

### User

...

### Agent

The search did not provide a direct link to the Lethal Company page on steambase.io, but it did mention steambase.io as a source for player counts. I should visit steambase.io and look for the Lethal Company page, or try to directly access the relevant URL. Once there, I can look for monthly average player counts for the period starting April 2024 (the release month of Version 50) and **check for any next major patch (likely Version 70)** or any other official patch that is listed as "major" after Version 50.

```
1 lethal_company_url = "https://steambase.io/games/lethal-company"
2
3 page_info = visit_webpage(url=lethal_company_url)
4
5 print(page_info)
```

**Error Analysis: Underperformed Iterative Refinement**

The next major patch is 'v60', on August 17, 2024. But the agent no longer checks whether 'v70' is reasonable and proceeds with subsequent steps.

### Agent

The update introducing Embrion and Old Bird (Version 50) was released in April 2024. The next major patch after Version 50 is not explicitly listed, but the next major patch after 50 is Version 70, which is referenced in previous search results as the next major update.

...

Now, I will extract the average player counts for Lethal Company from April 2024 to March 2025 (12 months, not including April 2025), as that's the period from the update's release month until (but not including) the next major patch's month.

```
1 # Monthly average players for Lethal Company from steambase.io (April
  2024 - March 2025)
2 lethal_company_averages = { "2024-04": 24020, "2024-05": 13345,
  "2024-06": 12073, "2024-07": 14299,
3
4 "2024-08": 16523, "2024-09": 16036, "2024-10": 13545, "
  2024-11": 14299,
5
6 "2024-12": 13120, "2025-01": 17462, "2025-02": 15399, "
  2025-03": 8886}
7
8 print(lethal_company_averages)
```

#### Error Analysis: Underperformed Iterative Refinement

The next major patch is 'v60', on August 17, 2024. But the agent no longer checks whether 'v70' is reasonable and proceeds with subsequent steps.

### User

The rest of the trajectory is omitted for readability.

## D.3 Example 3

### User

Using data from the World Bank for 2014-2023, calculate annual averages for: (1) GDP growth rate, (2) Inflation rate (consumer prices), (3) CO2 emissions per capita growth (excluding LULUCF), (4) Internet users (% population), (5) Women in parliament (%). Find the intersection between: (a) top 200 countries in (1),(3),(4),(5) and (b) bottom 200 countries in (2). Rank the countries in ascending order based on their CO2 emissions growth rate, with CO2 emissions growth rate rounded to 5 decimals. For the growth rate conditions mentioned above, if the World Bank provides growth rate data, prioritize using the World Bank's growth rate data. Otherwise, calculate it using the following formula: Growth Rate (%) =  $(\text{data}[n] - \text{data}[n-1]) / (\text{data}[n-1] + 0.00001) \times 100$  The small constant 0.00001 is added to avoid division by zero in cases where  $\text{data}[n-1] = 0$ . Give me the answer as 'Country: CO2 emissions growth rate'

### Agent

This is a comprehensive data-rich task involving 5 World Bank indicators for all countries from 2014-2023. I need to:

1. Identify the exact World Bank indicator codes for each metric.
  2. Download the datasets for each indicator (preferably from the World Bank API or CSV download links).
  3. For each country and indicator, compute the annual averages for the period 2014-2023.
  4. For indicators (1), (3), and (2): if WB has growth rate data, use it; if not, compute it using the formula provided.
  5. Select the top 200 countries in (1) GDP growth, (3) CO2 per capita growth, (4) Internet users, (5) Women in parliament, and bottom 200 in (2) inflation. Find the intersection countries among top 200 in 1/3/4/5 and bottom 200 in 2.
  6. Rank those by ascending order of their CO2 emissions growth rate, rounded to 5 decimals.
- This is a multi-step data task. First, I will search for the official World Bank indicator codes for each variable.

```
1 queries = [  
2  
3 "World Bank indicator for GDP growth rate",  
4  
5 "World Bank indicator for inflation rate (consumer prices)",  
6  
7 "World Bank indicator for CO2 emissions per capita growth (excluding  
8   LULUCF)",  
9 "World Bank indicator for Internet users percent the population",  
10  
11 "World Bank indicator for women in parliament percent"  
12  
13 ]  
14  
15 for q in queries:  
16     print(f"{q}:", web_search(query=q))
```

### User

...

## Agent

I'll process the data **\*\*directly in memory\*\***, reading from the zipfile into a 'csv.reader' and printing the header/first lines for inspection. approach requires no file system operations and should work given the allowed modules.

```
1 def average(values):
2     nums = [float(v) for v in values if v not in ('', None)]
3     return sum(nums)/len(nums) if nums else None
4
5 def CO2_growth_rate(vals):
6     # compute per-year growth rates, then average them
7     result = []
8     # Need at least two years to compute one growth rate
9     vals_num = [float(v) if v not in ('', None) else None for v in vals]
10    for i in range(1, len(vals_num)):
11        if vals_num[i] is not None and vals_num[i-1] is not None:
12            prev = vals_num[i-1]
13            curr = vals_num[i]
14            # formula with epsilon to avoid division by zero
15            rate = ((curr - prev) / (prev + 0.00001)) * 100
16            result.append(rate)
17    return sum(result)/len(result) if result else None
18
19 indicator_avgs = {}
20 for key in indicator_data:
21     indicator_avgs[key] = {}
22     for code, vals in indicator_data[key].items():
23         if key == "CO2_pc":
24             avg = CO2_growth_rate(vals)
25         else:
26             avg = average(vals)
27         indicator_avgs[key][code] = avg
28
29 # Print samples for all indicators to verify
30 for k in indicator_avgs:
31     print(f"Averages for {k}:", list(indicator_avgs[k].items())[:3])
```

## User

...

## Agent

I will obtain the countries with the highest CO2 emissions growth rates. Then I will begin calculating other metrics and finding their intersections.

```
1 # Prepare (country, rounded growth rate) pairs for intersection countries
2 results = []
3 for code in intersection:
4     val = indicator_avgs['CO2_pc'][code]
5     if val is not None:
6         results.append((code, round(val, 5)))
7
8 # Sort by CO2 emissions growth rate, ascending
9 results_sorted = sorted(results, key=lambda x: x[1])
10
11 # Obtain the top 200 countries and growth rate
12 country_rate=results_sorted[:200]
13 print(country_rate)
```

Error Analysis: Error Accumulation in Composition

Due to **the pre-reduction of the country list in this step**, the final intersection differed from the actual results, causing **this trajectory to ultimately return an empty list**.