WEBWATCHER: BREAKING NEW FRONTIERS OF VISION-LANGUAGE DEEP RESEARCH AGENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Web agents such as deep research have demonstrated superhuman cognitive abilities, capable of solving highly challenging information-seeking problems. However, most research remains largely text-centric, overlooking visual information in the real world. This makes multimodal deep research highly challenging, as such agents require much stronger perceptual, logical, and knowledge-based reasoning abilities, as well as proficiency in more sophisticated tools. To address this limitation, we introduce WebWatcher, a multimodal agent for deep research with joint reasoning ability across both visual and textual modalities. It uses high-quality synthetic trajectories for efficient cold start training, utilizes various tools for deep reasoning, and further enhances generalization through reinforcement learning. To better evaluate the capabilities of multimodal agents, we propose BrowseComp-VL, a benchmark with the style of BrowseComp that requires complex information retrieval involving both visual and textual information. Experimental results show that WebWatcher outperforms the prompt-based workflow and open-source agents on HLE and BrowseComp-VL, and demonstrates its perception, multimodal reasoning, and searching capabilities across the other three benchmarks, respectively.

1 INTRODUCTION

Deep research agents represent a new frontier in artificial intelligence, where Large Language Models (LLMs) go beyond static prompts to plan multi-step tasks (OpenAI, 2025a; Google, 2024; Perplexity, 2025). Open-source agents have demonstrated superhuman abilities to interact with intricate information (Li et al., 2025c;a; Wu et al., 2025a), achieving strong results on difficult benchmarks such as BrowseComp (Wei et al., 2025a) and Humanity's Last Exam (HLE) (Phan et al., 2025). Yet, progress so far is mostly text-focused, overlooking the visual information common in real-world tasks. Many scenarios such as interpreting scientific diagrams (Hu et al., 2024), analyzing charts (Wang et al., 2024), or navigating visual web interfaces (Hong et al., 2024), demand joint vision-language reasoning (Dong et al., 2025). Despite initial progress by proprietary agents, multimodal deep research remains largely unexplored, with few agents tackling high-difficulty Vision-Language (VL) tasks (Xu & Peng, 2025; Hu et al., 2025).

The key challenge is that current multimodal deep research agents rely on template-driven pipelines limited to specific scenarios, lacking the flexible reasoning and planning ability with versatile tools. On the one hand, many VL Agents focus on image-based reasoning with visual tools, such as Optical Character Recognition (OCR), bounding box extraction, image cropping, and visual annotation (Zhao et al., 2025; Su et al., 2025a;b). While useful for perception, agents with these tools fall short in combining visual reasoning with deep textual understanding and cross-modal inference, struggling to handle high-difficulty tasks. As shown in Fig. 1, VL agent fails to give a solution to this complex case from GAIA (Mialon et al., 2023), which requires reasoning beyond perception. On the other hand, search-only agents have a very limited problem-solving scope (Wu et al., 2025b). Although retrieval augmented reasoning can handle many knowledge-based questions, it often fails when answers are implicit, require interactions, or demand additional computation (Shen et al., 2024; Gu et al., 2025). As illustrated in Fig. 1, solving this case requires not only searching, but also tools to click through relevant links and browse the resulting webpage to gather necessary information.

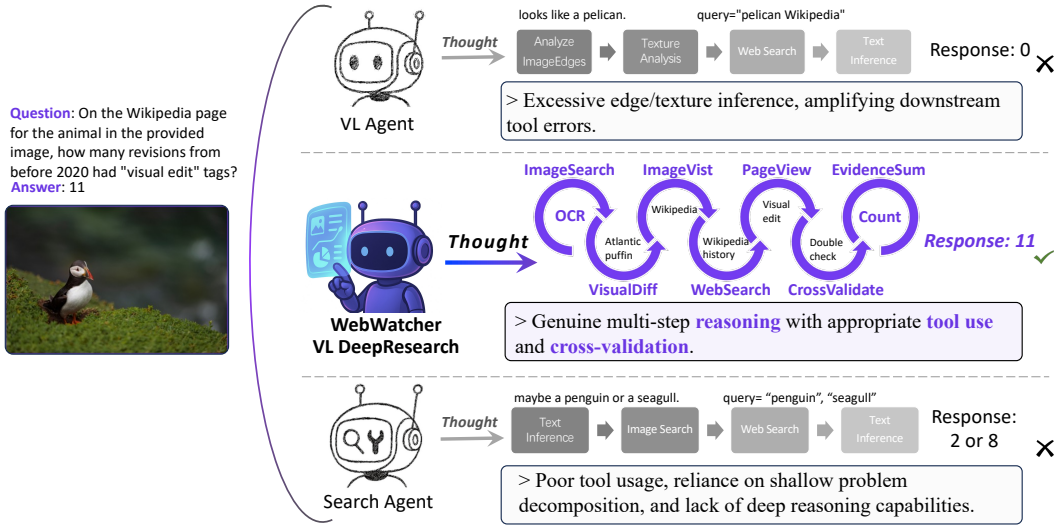


Figure 1: Comparison of VL reasoning agents. WebWatcher resolves the GAIA case that defeats both vision-only reasoning and search-based agents, demonstrating the strength of multi-tool integration and in-depth reasoning generalization.

To address this gap, agents for VL deep research require not only **strong reasoning abilities across both textual and visual information**, but also **effective use of multiple external tools**. Thus, we introduce **WebWatcher**, a VL web agent with deep research capability.

To develop strong reasoning across text and vision, it is essential to construct data that combine high-quality visual content with complex reasoning. However, current visual question answering (VQA) datasets mainly focus on visual perception with inference in two hops, lacking planning complexity and reasoning depth needed for deep research agent (Chen et al., 2024; Li et al., 2025d). Thus, we introduce a pipeline to generate training data that benefits in-depth, multi-step reasoning and strategic planning, encouraging agents to synthesize information across both modalities. We first harvest real-world knowledge via random walks over diverse web sources to create challenging question answering (QA) pairs (Wu et al., 2025a; Li et al., 2025a). To further raise the complexity, we mask key entities in questions with generic descriptions, forcing models to infer relationships from context. These enriched QA pairs are then converted into multimodal VQA items through a flexible pipeline compatible with most existing QA datasets, enabling large-scale multimodal dataset expansion. Finally, a multi-stage filtering process ensures data quality and clarity.

Moreover, to enable effective use of multiple external tools, we integrate Web Image Search, Web Text Search, Webpage Visit, Code Interpreter and internal OCR. However, a key challenge is constructing tool-use trajectories with high quality. Recent agents generate rigid, template-based trajectories with limited adaptability across tasks (Rose et al., 2023; Bi et al., 2025), because it is difficult to coordinate tools with distinct input-output formats and reasoning roles. To address this, we design an automated pipeline that builds trajectories from action–observation sequences via prompting. Unlike hand-crafted traces, our trajectories are grounded in actual tool-use behavior and reflect procedural decision-making aligned with complex reasoning. Then we finetune the agent on synthesized trajectories and further optimize it via reinforcement learning algorithm, GRPO (Shao et al., 2024).

Finally, we introduce **BrowseComp-VL**, a challenging VQA benchmark that extends BrowseComp (Wei et al., 2025b) into visual domain. Queries are long, entity-obfuscated, and BrowseComp-style, *i.e.*, queries that are deliberately underspecified and difficult even for humans, requiring retrieval of scattered information without clear guidance and integration of fragmented clues. It demands agents to perform not just perception but also cross-modal reasoning, thorough information-seeking, and high-level planning of tools such as web search, image retrieval, and webpage browsing.

WebWatcher achieves strong performance on several high-difficulty benchmarks, including HLE, LiveVQA, BrowseComp-VL, and MMSearch. It consistently outperforms existing open-source multimodal research agents and proprietary systems on four reasoning benchmarks, and show a competitive performance on the perception benchmark, SimpleVQA.

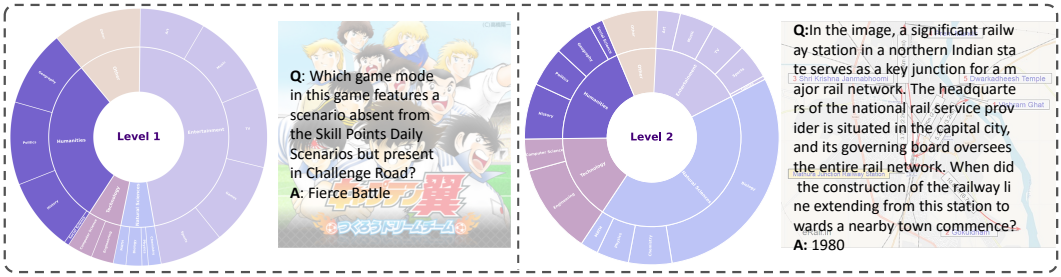


Figure 2: Domain distribution of Level 1 and Level 2 questions across five major fields, with examples illustrating explicit multi-hop reasoning (Level 1) and obfuscated, harder synthesis (Level 2).

2 Data Preparation

Prompting LLMs to generate VQA questions directly from images is a common practice, but it often yields shallow, single-hop queries. Existing datasets rarely combine rich textual information with complex reasoning. In this section, we address these gaps by constructing a dataset with multi-hop, knowledge-intensive queries and images from real-world web environments.

2.1 Data Overview

Our dataset is designed for multimodal deep research agents. Each example comprises a factual image, an associated question requiring cross-modal reasoning, an corresponding answer, and auxiliary metadata about underlying entities and relations. As shown in Fig. 2, the dataset cover 5 major domains (*Entertainment, Humanities, Technology, Natural Science, and Other*), comprising 17 fine-grained subfields, which is detailed in Appendix C. Additionally, we define two difficulty levels to encourage diverse reasoning ability:

Level 1: Questions require multi-hop reasoning but still reference explicit entities. While the answers can be obtained through iterative retrieval steps, the reasoning process remains non-trivial due to the need for integrating information across multiple sources.

Level 2: Questions are constructed with obfuscated entities and attributes. For example, concrete dates are replaced with vague periods, names are masked, and quantitative properties are fuzzed up. This design introduces uncertainty, requiring the agent to plan, compare, and synthesize information rather than perform direct retrieval.

We split the dataset into a training set and a benchmark **BrowseComp-VL**, evaluating agents under highly difficult multimodal research scenarios. It consists of 199 VQA pairs in Level 1 and 200 VQA pairs in Level 2, verified by three PhD-level human experts to ensure high quality and reliability. Specifically, annotators select more information-dense images.

2.2 Construction of VQA Pairs

We first construct diverse textual QA pairs emphasizing multi-hop and knowledge-intensive reasoning, then ground them in relevant images to form VQA tasks. This pipeline produces multimodal data that preserve both visual richness and reasoning complexity.

2.2.1 QA Pairs Generation

Level 1. Inspired by CRAWL-QA from WebDancer (Wu et al., 2025a), we enhance reasoning depth and breadth by collecting root URLs from authoritative sources such as *arXiv*, *GitHub*, and *Wikipedia*, and recursively traversing their hyperlinks to mimic human browsing. GPT-4o (OpenAI, 2024) is then used to synthesize question-answer pairs from the aggregated content.

Level 2. Following WebSailor (Li et al., 2025a), we construct queries with fuzzed entities by replacing precise references with partial or ambiguous descriptions. Answers cannot be retrieved through direct lookup, but require contextual reasoning and synthesis across modalities. We design a two-stage generation framework for Level 2 consisting of: **(1) Nodes Selecting:** Starting from an initial *Wikipedia* page, we prompt GPT-4o to generate a base QA pair, using the page title as the root

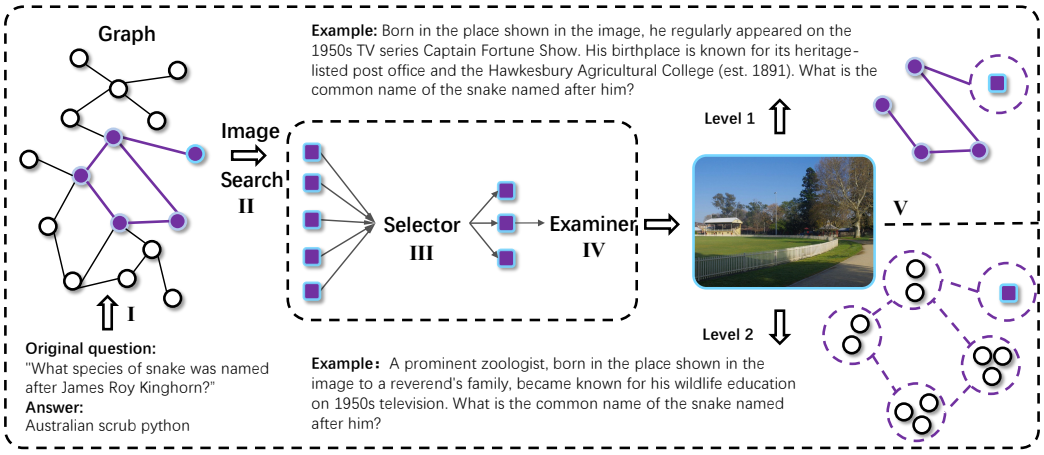


Figure 3: Pipeline for generating data, where multi-hop VQA pairs are built from hyperlink graphs, grounded with web images, filtered by selector–examiner checks, and transformed into Level 1 (explicit) and Level 2 (fuzzed) questions for multimodal reasoning.

entity node B_{root}. We then expand a hyperlink graph by recursively traversing outgoing links to form a tree of depth d and branching factor k, yielding $(k^{d+1}-1)/(k-1)$ nodes. In practice, we set $d=3$ and $k=3$ for sufficient coverage. To create diverse reasoning paths, we sample subgraphs of N entities, each defining a path from B_{root} to a target entity B. These subgraphs provide the basis for generating distinct multi-hop QA pairs. **(2) Query Generating and Entity Masking:** Based on each subgraph and its ground truth, we first prompt GPT-4o to generate a standard question that explicitly references entities and relations along the reasoning path. We then create a fuzzed version by replacing key references with partial or ambiguous descriptions, preventing agents from taking shortcuts via simple string matching in search results and forcing it to perform cross-modal reasoning.

2.2.2 QA-TO-VQA CONVERSION

Visual Context Construction. To ensure reliable visual grounding, we discard trivial or overly ambiguous target entities B (e.g., temporal references or domain-external concepts), which lack sufficient visual grounding. For each retained entity $\hat{B}$, we retrieve a set of web images $\mathcal{I}(\hat{B}) = I_1^{\hat{B}}, I_2^{\hat{B}}, \ldots, I_K^{\hat{B}}$ via Google SerpApi (Google, 2025), where $K=2$ in our implementation. These images $\mathcal{I}(\hat{B})$ serve as the visual grounding to construct multimodal reasoning examples. Unlike synthetic or composited images prevalent in existing VQA benchmarks, our images are strictly authentic, minimizing noise and maximizing relevance for real-world tasks.

Question Transformation. To build image-grounded VQA pairs from each textual QA (q_t, a), we use GPT-4o for prompt-based rewriting. The target entity $\hat{B}$ in q_t is masked with a visual reference token r_{vis} (e.g., "this entity," "the object in the image"), producing a transformed VQA query q. Simultaneously, we create an image query string $s_{\text{img}}(\hat{B})$ to guide filtering of $\mathcal{I}(\hat{B})$. Each retained image $I_k^{\hat{B}} \in \mathcal{I}$ is paired with (q, a), so one textual QA yields K multimodal examples, giving $K \cdot n$ VQA items from n questions.

2.3 QUALITY CONTROL

To ensure high-quality VQA samples, we employ a two-stage filtering pipeline:

(1) Selector: We first discard cases where the transformed VQA query q is identical to q_t, or where the entity name $\hat{B}$ and its aliases appear in q, indicating failed masking and question rewriting. Then, GPT-4o evaluates each image $I_k^{\hat{B}} \in \mathcal{I}(\hat{B})$ against both (q_t, a) and (q, a), scoring contextual alignment, semantic fit, and visual reasoning plausibility. Cases with low scores are removed.

(2) Examiner: For each retained image-query pair $(s_{\text{img}}(\hat{B}), \mathcal{I}(\hat{B}))$, GPT-4o attempts to answer $s_{\text{img}}(\hat{B})$ using only visual content and associated captions. Failure to answer accurately iindicates

improper visual context, and such cases are discarded. Captions are included to reduce false negatives from missing world knowledge.

The input images undergo automated filtering, and when curating the benchmark annotators further select information-dense images. The entity that may be used to solve the problem is often only one part of the complex information contained in the image. Therefore, agents require non-trivial visual reasoning to solve these tasks and also need to integrate information from multiple sources.

3 Trajectory Generation and Post-Training

We use supervised fine-tuning (SFT) as a cold start to teach WebWatcher tool-augmented reasoning, based on high-quality trajectories generated by an automated pipeline. Reinforcement learning is then applied to further optimize tool use and decision-making.

3.1 Automated Generation of Reasoning Trajectories

3.1.1 Multimodal Tools

We equip WebWatcher with five tools: (1) **Web Image Search** (Google SerpApi (Google, 2025)) for retrieving relevant images with captions and URLs; (2) **Web Text Search** for open-domain information seeking; (3) **Visit** (Jina (Jina.ai, 2025)) for navigating specific URLs and summarizing pages according to the agent's goal; (4) **Code Interpreter** for symbolic computation and numerical reasoning (Cheng et al., 2024); and (5) **OCR**, an internal tool invoked via prompt and SFT data to extract text from input images (Huang et al., 2025). Full details are provided in Appendix E.

3.1.2 Automated Trajectory Annotation

Given a VQA instance (I, q, a) from BrowseComp-VL, we use GPT-4o to construct tool-use trajectories simulating step-by-step human reasoning. Following ReAct (Yao et al., 2023), each trajectory τ comprises multiple *think-act-observe* cycles. At each step t, the model takes as input the accumulated context history and generates: (1) a **Thought**: the agent's intermediate reasoning or plan, enclosed in `<think>...</think>`; (2) an **Action**: the tool invocation wrapped in `<tool_call>...</tool_call>` and the final answer enclosed in `<answer>...</answer>`; (3) an **Observation**: the returned result from the environment, within `<tool_response>...</tool_response>` tags.

The action space $\mathcal{T}$ consists of discrete tool-use actions t_l, enabling the agent to retrieve information, navigate webpages, or perform computations. The **Finish** action signals task completion by returning a final answer and ending the episode. A trajectory of length L is defined in Eq. 1:

$$\tau = \{(t_0, o_0), (t_1, o_1), \ldots, (t_L, o_L)\}, \tag{1}$$

where each observation o_i reflects the environment feedback after executing action $t_i \in \mathcal{T}$. Each trajectory provides a content-grounded demonstration of planning and tool selection.

3.1.3 Trajectory Filtering and Quality Assurance

To ensure robust and instructive supervision, we apply three-stage trajectory selection:

(1) Final Answer Matching: We retain trajectories τ where the final answer matches the ground truth a, ensuring that the entire sequence of tool-use steps leads to a correct and complete solution. **(2) Step-by-Step Consistency Check:** We use GPT-4o to verify the logical consistency of each intermediate step in τ. Trajectories with hallucinated content, contradictions, or unjustified tool calls are discarded. This avoids the common failure mode where correct answers are reached by lucky guessing rather than meaningful tool use. **(3) Minimum Tool Usage Requirement:** We remove τ with fewer than three tool calls. This ensures that training data reflects substantive, process-driven tool interactions and reasoning rather than one-step completions.

3.2 Supervised Fine-Tuning as Cold Start

After filtering, the dataset contains K high-quality tool-use trajectories. At each step l of trajectory i, WebWatcher is trained to predict the correct action $t_l^{(i)}$, given the image $I^{(i)}$, question $q^{(i)}$, and previous actions and observations $(t_{<l}^{(i)}, o_{<l}^{(i)})$. SFT maximizes the log-likelihood of $t_l^{(i)}$ in Eq. 2:

$$\max_{\theta} \sum_{i=1}^{K} \sum_{l=1}^{L_i} \log P_\theta\big(t_l^{(i)} \mid I^{(i)}, q^{(i)}, t_{<l}^{(i)}, o_{<l}^{(i)}\big), \tag{2}$$

where θ are the model parameters. This cold-start stage teaches the agent to use tools effectively and follow structured multi-step reasoning.

3.3 Reinforcement Learning

With SFT providing cold-start initialization, we apply Group-Relative Policy Optimization (GRPO) (Guo et al., 2025) to refine decision-making for complex tasks. For a VQA query q, the current policy π_θ generates a group $G = \tau_1, \ldots, \tau_K$ of K complete trajectories, each with return R_i. The group-relative advantage is defined as Eq. 3:

$$A_{\text{rel}}(\tau^{(i)}) = R^{(i)} - \frac{1}{K} \sum_{j=1}^{K} R^{(j)} \tag{3}$$

which normalizes rewards within the group and eliminates the reliance on a separate value function. The GRPO objective is defined as a clipped surrogate loss in Eq. 4:

$$\mathcal{L}_{\text{GRPO}}(\theta) = \mathbb{E}_{\tau^{(i)} \in \mathcal{G}} \left[\min \left(\rho^{(i)} A_{\text{rel}}(\tau^{(i)}), \text{clip} \left(\rho^{(i)}, 1-\epsilon, 1+\epsilon \right) A_{\text{rel}}(\tau^{(i)}) \right) \right] - \beta \, D_{\text{KL}} \left(\pi_\theta \| \pi_{\theta_{\text{old}}} \right), \tag{4}$$

where $\rho^{(i)} = \frac{\pi_\theta(\tau^{(i)})}{\pi_{\theta_{\text{old}}}(\tau^{(i)})}$ is the importance sampling ratio between the current and previous policy, $A_{\text{rel}}(\tau^{(i)})$ is the group-relative advantage defined in Eq. 3, ϵ is the clipping threshold, and D_{KL} denotes the Kullback–Leibler divergence between successive policies. The coefficient β controls the strength of the KL penalty. This objective promotes stable updates while encouraging exploration of trajectories with higher relative return. Each trajectory $\tau = (t_0, o_0), \ldots, (t_L, o_L)$ first receives a binary format score $r_{\text{f}} \in [0, 1]$, which is 1 if all tool calls follow the schema. An LLM grader then provides a semantic accuracy score $r_{\text{a}} \in [0, 1]$ by comparing the final answer with the ground truth. The total reward is defined in Eq. 5:

$$R = w r_{\text{f}} + (1 - w) r_{\text{a}}, \tag{5}$$

with $w = 0.2$ to prioritize task completion while maintaining structured tool use. Since R is given only at the episode end, the group-relative ranking in Eq. 3 enables effective credit assignment without relying on per-step shaping. Rollouts are collected in groups of $N = 16$ to ensure diversity for computing relative advantages while maintaining computational efficiency during training.

4 Experiments

4.1 Setup

Training Data Construction. Our training data come from three sources: (1) The training set of BrowseComp-VL, (2) long-tail QA pairs converted to VQA, and (3) hard VQA samples. Firstly, we construct the BrowseComp-VL training set with 110,000 Level-1 and 70,000 Level-2 QA pairs. After VQA conversion and filtering, 60,000 Level-1 and 40,000 Level-2 high-quality examples are retained. Secondly, the long-tail QA data are sampled from training instances sharing a similar distribution with SimpleVQA, which are transformed into 4,000 VQA examples. Thirdly, hard samples are from InfoSeek (Chen et al., 2023), VQAv2.0 (Goyal et al., 2017), LogicVista (Xiao et al., 2024), and Encyclopedic VQA (Mensink et al., 2023), with (Huang et al., 2025) added to activate OCR. Rejection sampling ensures difficulty. After trajectory generation and filtering, we obtain 8,000 high-quality tool-use trajectories for SFT, with 2,000 additional samples reserved for GRPO. The final ratio of data sources is 5:3:2 for BrowseComp-VL, long-tail VQA, and hard VQA data, respectively.

Table 1: Main results on HLE. All accuracy scores are reported as percentages. Avg signifies the average accuracy score of three inference runs across different subtopics.

Backbone	Humanity's Last Exam (HLE-VL)								
	Bio.	Chem.	CS/AI	Engineer.	Human.	Math	Physics	Other	Avg.
Direct Inference									
GPT-4o	13.8	0.0	0.0	3.9	12.0	6.8	7.1	7.0	6.5
Gemini-2.5-flash	12.1	1.6	0.0	0.0	4.0	0.0	14.3	0.0	4.9
Claude-3.7-Sonnet	1.7	4.8	0.0	2.0	0.0	0.0	0.0	12.3	2.8
Qwen-2.5-VL-7B	3.4	3.2	7.1	0.0	4.0	2.3	7.1	0.0	2.6
Qwen-2.5-VL-32B	3.4	6.5	0.0	3.9	8.0	2.3	7.1	0.0	3.7
Qwen-2.5-VL-72B	3.4	8.0	0.0	5.9	8.0	0.0	0.0	7.0	4.9
Prompt Workflow									
GPT-4o	9.8	24.1	4.8	0.0	2.0	4.0	9.1	14.3	12.3
Gemini-2.5-flash	25.9	3.2	7.1	0.0	8.0	9.1	3.5	14.0	11.4
Claude-3.7-Sonnet	4.3	5.2	4.8	0.0	0.0	0.0	9.1	14.3	3.5
Qwen-2.5-VL-7B	4.3	6.9	3.2	7.1	0.0	4.0	4.5	7.1	5.3
Qwen-2.5-VL-32B	5.2	10.3	3.2	7.1	0.0	0.0	4.5	7.1	8.8
Qwen-2.5-VL-72B	15.8	10.3	8.1	0.0	2.0	8.0	6.8	14.3	8.6
Reasoning Model									
o4-mini	12.1	23.7	17.7	0.0	5.8	0.0	33.3	21.4	16.0
Gemini-2.5-Pro	23.7	17.7	13.3	11.5	8.0	13.3	14.3	15.5	15.8
Open Source Agents									
OmniSearch (GPT-4o)	15.5	8.2	0.0	2.2	**8.0**	6.8	**21.4**	12.1	9.3
WebWatcher-7B	18.6	6.5	**6.7**	**7.7**	4.0	6.7	7.1	**17.2**	10.6
WebWatcher-32B	**33.8**	**9.7**	0.0	5.8	**8.0**	**8.9**	14.3	13.8	**13.6**

Models and Benchmarks. We conduct post-training on Qwen2.5-VL-7B and Qwen2.5-VL-32B (Bai et al., 2025), and evaluate on five challenging benchmarks: BrowseComp-VL, HLE (Phan et al., 2025), LiveVQA (Fu et al., 2025), SimpleVQA (Cheng et al., 2025), and MMSearch (Jiang et al., 2024). Benchmark details are provided in Appendix E.4.

Baselines. We compare with the following paradigms: (1) Direct Inference: Models directly generate answers using internal knowledge. We evaluate GPT-4o (OpenAI, 2024), Gemini-2.5-flash (DeepMind, 2025), Claude-3.7-Sonnet (Anthropic, 2025), and Qwen-2.5-VL family (7B/32B/72B). (2) Prompt Workflow: To ensure fairness, models use prompt-driven workflows have the same tools as employed in WebWatcher. (3) Reasoning Baselines: OmniSearch (Li et al., 2025d), a search-oriented open-sourced agent based on GPT-4o, and multi-step reasoning models Gemini-2.5-Pro (DeepMind, 2025) and o4-mini (OpenAI, 2025b) with prompt-driven workflows.

Metric and Hyper-parameters We repeatedly generate for k times to get pass@k (Chen et al., 2021), with the temperature of 0.6 and top-p of 0.95. The pass@1 score is computed as: pass@1 $= \frac{1}{n}\sum_{i=1}^{n} p_i$, where p_i is the binary correctness of the i-th prediction. Answer correctness is judged using the *LLM-as-Judges* approach (Liu et al., 2024), where the prompt is detailed in Appendix E.5.

4.2 Main Results

As shown in Tab. 1, for Humanity's Last Exam (HLE), models with direct inference perform worst, with average accuracy scores below 10, revealing the limits of vanilla MLLMs in complex, knowledge-intensive VQA. RAG-based methods show moderate improvements, particularly in Chemistry. Compared with reasoning-oriented models, although our WebWatcher-32B attains a slightly lower overall average accuracy of 13.6%, it is parameter-efficient, requiring only 32B parameters compared to the proprietary large-scale models. Moreover, WebWatcher-32B demonstrates clear strengths in specific domains: it achieves a top score of 33.8% in Biology, with competitive performance in Mathematics and Humanities. As for other benchmarks, Tab. 2 shows that while direct inference with strong MLLMs remains limited and prompt workflows bring moderate gains, WebWatcher consistently outperforms both. WebWatcher-32B achieves state-of-the-art results on LiveVQA (58.7%) and MMSearch (55.3%), while also delivering competitive performance on BrowseComp-VL and SimpleVQA.

Table 2: Main results on four challenging benchmarks. All accuracy scores are reported as percentages. Avg signifies the average score of three inference across two difficult levels.

Backbone	BC-VL			LiveVQA	MMSearch	SimpleVQA
	Level1	Level2	Avg.			
Direct Inference						
GPT-4o	6.4	4.0	5.5	29.7	18.7	47.0
Gemini-2.5-flash	11.6	6.0	9.6	35.0	19.6	63.0
Claude-3.7-Sonnet	8.8	4.0	7.1	23.7	12.3	42.7
Qwen-2.5-VL-7B	0.8	0.0	0.5	22.7	4.09	30.7
Qwen-2.5-VL-32B	3.2	1.0	2.4	26.3	7.60	40.7
Qwen-2.5-VL-72B	9.2	3.0	7.1	30.3	11.7	51.3
Prompt Workflow						
GPT-4o	16.8	7.0	13.4	34.0	24.1	61.6
Gemini-2.5-flash	15.2	9.0	13.0	41.3	43.9	68.6
Claude-3.7-Sonnet	13.9	6.0	11.2	30.3	32.7	59.3
o3	26.7	23.0	24.9	50.0	54.3	**70.3**
GPT-4.1	15.6	6.0	10.8	32.3	26.0	60.3
Qwen-2.5-VL-7B	3.6	1.0	2.7	21.7	9.94	21.0
Qwen-2.5-VL-32B	9.4	3.0	7.2	30.5	17.5	44.6
Qwen-2.5-VL-72B	14.4	6.0	11.5	35.7	29.2	58.6
Agents						
OmniSearch (GPT-4o)	19.7	10.0	16.3	40.9	49.7	63.0
WebWatcher-7B	23.6	17.0	21.2	51.2	49.1	54.3
WebWatcher-32B	**28.4**	**25.0**	**27.0**	**58.7**	**55.3**	59.0

Table 3: Performance across different tool call counts.

Tool Call	Best Pass@1	Average@3	Best Pass@3
=1	8.79	7.98	14.24
$\geq$1	9.70	8.69	15.76
=2	10.61	9.90	18.18
$\geq$2	11.21	10.10	17.58
=3	10.61	9.90	19.09
$\geq$3	12.12	10.61	19.09
=4	10.00	8.99	18.79
$\geq$4	11.82	9.70	17.27
=5	9.70	9.49	16.58
$\geq$5	10.61	10.10	18.18
=6	8.79	8.33	15.76
$\geq$6	9.70	8.99	15.58

Notably, BrowseComp-VL requires multi-page browsing and fine-grained visual grounding, which causes the scores of most baselines are below 20%, but our dynamic tool-use loop proves effective. Even in SimpleVQA, which emphasizes visual reasoning over external knowledge, WebWatcher performs well with a score of 59.0%. These results demonstrate that WebWatcher excels in not only knowledge-intensive tasks but also visual reasoning, underscoring its broad applicability across VQA benchmarks.

4.3 Ablation Study

We conduct an ablation study of selecting the number of tool calls. For each tool call setting, we randomly select 8,000 trajectories for SFT and tested them on HLE. Tab. 3 shows that the performance is best when the number of tool calls is ≥ 3.

Table 4: Human vs Agent Baseline Results for BrowseComp-VL. T_s and T_u mean average minutes of solvable cases and unsolveable cases, respectively. I_u indicates abandoned cases without final answer.

Level	Acc (%)	I_u	T_s(/m)	T_u(/m)
L1 (Human)	33.2	42	35	59
L2 (Human)	18.0	144	109	116
L1 (WebWatcher-32B)	28.4	1	0.3	2.5
L2 (WebWatcher-32B)	25.0	3	0.8	2.5

4.4 Human Baseline

We report a human baseline on Level 1 and Level 2 of BrowseComp-VL in Tab.4. In this experiment, five annotators are asked to answer questions from the BrowseComp-VL benchmark. To ensure reliability, each annotator answer 40 questions, ensuring that each question is answered by at least two annotators. Importantly, annotators are not allowed to answer questions that they have previously annotated, maintaining objectivity in the evaluation process. Besides, annotators have access to the same tools as WebWatcher. Following BrowseComp, if annotators spent 100 minutes attempting to answer a question without providing a solution, they are allowed to abandon the task.

4.5 Analysis

Number of Tool Calls. Fig. 4 shows how tool usage adapts to benchmark demands. On HLE, which requires multimodal search, computation, and reasoning, usage is balanced across *Web Text Search*, *Web Image Search*, and *Code Interpreter*, with *Visit* handling webpage navigation. BrowseComp-VL and MMSearch focus much more on information seeking and reasoning, thus retrieval dominates. *Web Text Search* accounts for 62% of calls, while other tools play minor roles. For SimpleVQA, the focus shifts back to visual content, with *Web Image Search* making up one-third or more of calls, while *Text Search* and *Visit* act as auxiliaries. Across all settings, the *Code Interpreter* is used only when actual computation is required, confirming that WebWatcher is cost and context aware. Overall, the distribution of tool usage mirrors benchmark demands, underscoring WebWatcher's flexibility in composing tool chains rather than over-relying on any single tool.

Cold Start for RL Training. We believe cold start is crucial for our vision–language agent, as tasks demand robust multi-hop reasoning with continuous tool interaction. To verify this, we compare the same RL algorithm under two initializations: (1) Instruct: warm-started only with public instruction-following data; (2) Cold-start: an extra SFT stage on trajectories that explicitly demonstrate tool use and step-by-step visual reasoning. As shown in Fig. 5, The Instruct initialization stalls near zero because frequent tool-call format errors wipe out rewards and the strict Qwen-2.5-72B grader suppresses partial answers. In contrast, cold-start SFT lifts initial scores. Subsequent GRPO trends diverge: HLE and BC-VL oscillate without improvement, while LiveVQA rises steadily, maintaining a 0.06–0.18 margin over Instruct. Injecting CoT chains from a larger reasoner made the small model unstable, format violations, repetitions, and context overflow spike, confirming that reasoning traces cannot replace an SFT cold start under our strict RL setting.

Pass@k Analysis on HLE. Fig. 6 shows the *Pass@k* curve of WebWatcher on HLE for k ranging from 1 to 32. With a single attempt ($k = 1$), WebWatcherachieves 13.6% pass rate. As k increases, performance rises steeply at first: three roll-outs reach 20.3%, showing that a few diverse trajectories yield large gains. Accuracy keeps improving to 35.7% at $k = 16$ and 41.9% at $k = 32$, nearly quadrupling single-shot inference and surpassing reasoning models like Gemini-2.5-Pro and o4-mini. The smooth curve indicates that our de-correlated sampling avoids redundant rollouts and captures complementary knowledge. Since marginal gains taper after $k \approx 16$, practitioners can cap at 8–16 roll-outs for a 2–3× boost at moderate cost. Overall, the Pass@k profile demonstrates the scalability of the agentic paradigm. Systematic exploration of reasoning paths yields consistent, robust improvements on a challenging multimodal benchmark.

5 Conclusion and Future Work

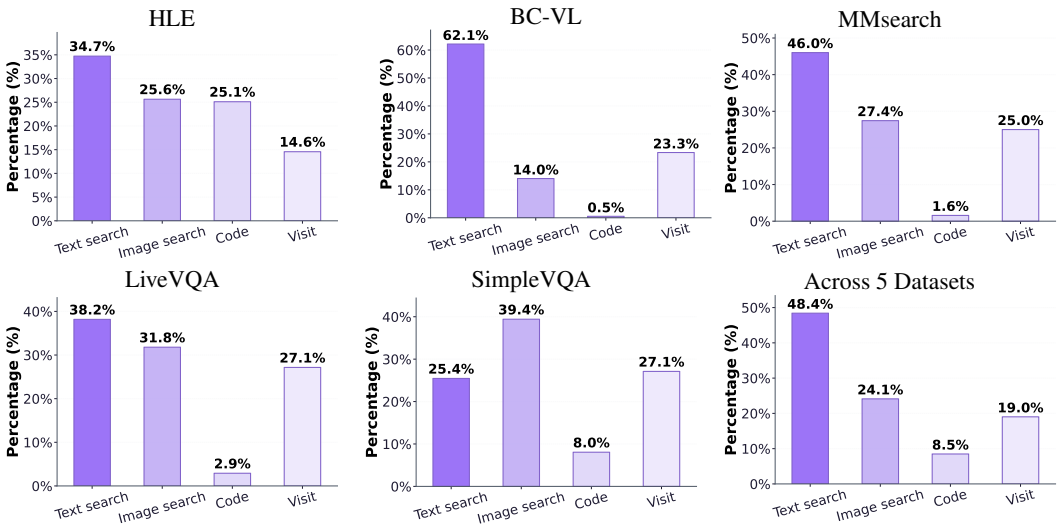
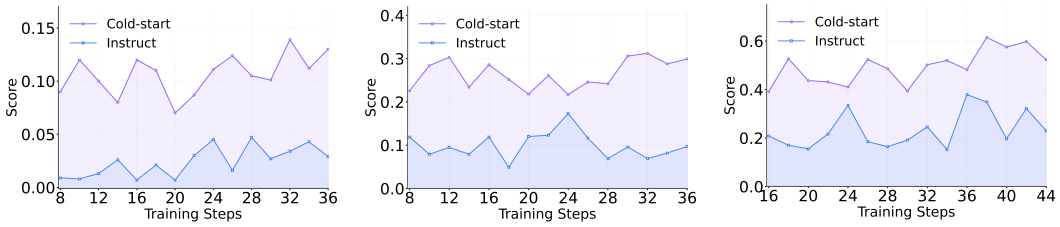


Figure 4: The percentage of external tool calls in the four benchmarks. The height of each bar denotes the fraction of total calls made to that tool within the corresponding benchmark. Internal OCR is not included since only external tools are counted here.



Score on HLE Benchmark | Score on BC-VL Benchmark | Score on LiveVQA Benchmark

Figure 5: Performance comparison using cold start in RL training on three benchmarks.

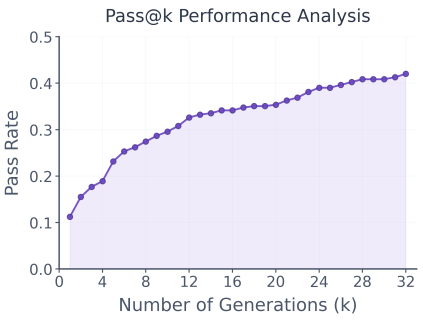


Figure 6: Pass@k performance of WebWatcher on HLE Benchmark.

In this work, we explore the underdeveloped landscape of multimodal deep research by designing a unified framework, WebWatcher, that combines complex vision–language reasoning and multi-tool interaction. We present BrowseComp-VL, a challenging dataset tailored for in-depth multimodal reasoning and strategic planning, and introduce a scalable pipeline to transform complex textual QA examples into VQA. To equip agents with robust tool-use capabilities, we develop an automated trajectory generation pipeline grounded in action–observation traces, followed by cold start and GRPO. WebWatcher achieves strong performance across multiple high-difficulty benchmarks (*e.g.,* HLE, LiveVQA, BrowseComp-VL, and MMSearch), outperforming both open-source and proprietary research agents, while also delivering competitive results on the perception-oriented benchmark SimpleVQA. Overall, WebWatcher establishes a strong foundation for future multimodal deep research agents capable of solving real-world problems with autonomy, flexibility, and deep reasoning.

In future work, we plan to systematize the data-flywheel to progressively replace GPT-4o with open-source models while preserving trajectory quality. Initially, GPT-4o annotations are used to train a smaller, specialized model on limited trajectories. This model then performs additional annotations at a much lower cost. Through iterative re-annotation and fine-tuning, the model becomes more proficient, handling increasingly complex tasks.

6 Ethics Statement

All authors have read and comply with the ICLR Code of Ethics. This work does not involve human subjects or sensitive data, and we are unaware of any potential misuse, harm, or bias. No conflicts of interest or compromising sponsorships exist.

7 Reproducibility Statement

Details of the proposed methodology, training procedure, hyperparameters, and evaluation metrics are provided in Sec. 4.1 and Appendix E. We also include a full description of the datasets in Sec. 4.1 and Appendix E. Additionally, public datasets used in our paper and our constructed benchmark, BrowseComp-VL, are all publicly available, ensuring consistent and reproducible evaluation results.

References

Anthropic. Claude 3.7 sonnet and claude code, 2025. URL https://www.anthropic.com/news/claude-3-7-sonnet/.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

Jing Bi, Susan Liang, Xiaofei Zhou, Pinxin Liu, Junjia Guo, Yunlong Tang, Luchuan Song, Chao Huang, Guangyu Sun, Jinxi He, et al. Why reasoning matters? a survey of advancements in multimodal reasoning (v1). *arXiv preprint arXiv:2504.03151*, 2025.

James Burgess, Jeffrey J Nirschl, Laura Bravo-Sánchez, Alejandro Lozano, Sanket Rajan Gupte, Jesus G Galaz-Montoya, Yuhui Zhang, Yuchang Su, Disha Bhowmik, Zachary Coman, et al. Microvqa: A multimodal reasoning benchmark for microscopy-based scientific research. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19552–19564, 2025.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021.

Qiguang Chen, Libo Qin, Jin Zhang, Zhi Chen, Xiao Xu, and Wanxiang Che. m^3 cot: A novel benchmark for multi-domain multi-step multi-modal chain-of-thought. *arXiv preprint arXiv:2405.16473*, 2024.

Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. Can pre-trained vision and language models answer visual information-seeking questions? *arXiv preprint arXiv:2302.11713*, 2023.

Zhuo Chen, Xinyu Wang, Yong Jiang, Zhen Zhang, Xinyu Geng, Pengjun Xie, Fei Huang, and Kewei Tu. Detecting knowledge boundary of vision large language models by sampling-based inference. *arXiv preprint arXiv:2502.18023*, 2025.

Xianfu Cheng, Wei Zhang, Shiwei Zhang, Jian Yang, Xiangyuan Guan, Xianjie Wu, Xiang Li, Ge Zhang, Jiaheng Liu, Yuying Mai, et al. Simplevqa: Multimodal factuality evaluation for multimodal large language models. *arXiv preprint arXiv:2502.13059*, 2025.

Yao Cheng, Jianfeng Chen, Jie Chen, Li Chen, Liyu Chen, Wentao Chen, Zhengyu Chen, Shijie Geng, Aoyan Li, Bo Li, et al. Fullstack bench: Evaluating llms as full stack coders. *arXiv preprint arXiv:2412.00535*, 2024.

Google DeepMind. Gemini 2.5, 2025. URL https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/.

Yuhao Dong, Zuyan Liu, Hai-Long Sun, Jingkang Yang, Winston Hu, Yongming Rao, and Ziwei Liu. Insight-v: Exploring long-chain visual reasoning with multimodal large language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 9062–9072, 2025.

Mingyang Fu, Yuyang Peng, Benlin Liu, Yao Wan, and Dongping Chen. Livevqa: Live visual knowledge seeking. *arXiv preprint arXiv:2504.05288*, 2025.

Google. Try deep research and our new experimental model in gemini, your ai assistant, 2024. URL https://blog.google/products/gemini/google-gemini-deep-research/.

Google. Serpapi, 2025. URL https://serpapi.com/.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6904–6913, 2017.

Jiawei Gu, Ziting Xian, Yuanzhen Xie, Ye Liu, Enjie Liu, Ruichao Zhong, Mochi Gao, Yunzhi Tan, Bo Hu, and Zang Li. Toward structured knowledge reasoning: Contrastive retrieval-augmented generation on experience. *arXiv preprint arXiv:2506.00842*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, et al. Cogagent: A visual language model for gui agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14281–14290, 2024.

Anwen Hu, Yaya Shi, Haiyang Xu, Jiabo Ye, Qinghao Ye, Ming Yan, Chenliang Li, Qi Qian, Ji Zhang, and Fei Huang. mplug-paperowl: Scientific diagram analysis with the multimodal large language model. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 6929–6938, 2024.

Mengkang Hu, Yuhang Zhou, Wendong Fan, Yuzhou Nie, Bowei Xia, Tao Sun, Ziyu Ye, Zhaoxuan Jin, Yingru Li, Qiguang Chen, et al. Owl: Optimized workforce learning for general multi-agent assistance in real-world task automation. *arXiv preprint arXiv:2505.23885*, 2025.

Mingxin Huang, Yongxin Shi, Dezhi Peng, Songxuan Lai, Zecheng Xie, and Lianwen Jin. Ocr-reasoning benchmark: Unveiling the true capabilities of mllms in complex text-rich image reasoning. *arXiv preprint arXiv:2505.17163*, 2025.

Dongzhi Jiang, Renrui Zhang, Ziyu Guo, Yanmin Wu, Jiayi Lei, Pengshuo Qiu, Pan Lu, Zehui Chen, Chaoyou Fu, Guanglu Song, et al. Mmsearch: Benchmarking the potential of large models as multi-modal search engines. *arXiv preprint arXiv:2409.12959*, 2024.

Jina.ai. Jina, 2025. URL https://jina.ai/.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Liwen Zhang, Litu Ou, Jialong Wu, Wenbiao Yin, Baixuan Li, Zhengwei Tao, Xinyu Wang, et al. Websailor: Navigating super-human reasoning for web agent. *arXiv preprint arXiv:2507.02592*, 2025a.

Shilong Li, Xingyuan Bu, Wenjie Wang, Jiaheng Liu, Jun Dong, Haoyang He, Hao Lu, Haozhe Zhang, Chenchen Jing, Zhen Li, et al. Mm-browsecomp: A comprehensive benchmark for multimodal browsing agents. *arXiv preprint arXiv:2508.13186*, 2025b.

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

Yangning Li, Yinghui Li, Xinyu Wang, Yong Jiang, Zhen Zhang, Xinran Zheng, Hui Wang, Hai-Tao Zheng, Philip S. Yu, Fei Huang, and Jingren Zhou. Benchmarking multimodal retrieval augmented generation with dynamic vqa dataset and self-adaptive planning agent, 2025d. URL https://arxiv.org/abs/2411.02937.

Yuxuan Liu, Tianchi Yang, Shaohan Huang, Zihan Zhang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. Calibrating llm-based evaluator. In Nicoletta Calzolari, Min-Yen Kan, Véronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (eds.), *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pp. 2638–2656. ELRA and ICCL, 2024. URL https://aclanthology.org/2024.lrec-main.237.

Thomas Mensink, Jasper Uijlings, Lluis Castrejon, Arushi Goel, Felipe Cadar, Howard Zhou, Fei Sha, André Araujo, and Vittorio Ferrari. Encyclopedic vqa: Visual questions about detailed properties of fine-grained categories. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3113–3124, 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.

OpenAI. Hello GPT-4o, 2024. URL https://openai.com/index/hello-gpt-4o/.

OpenAI. Deep research system card, 2025a. URL https://cdn.openai.com/deep-research-system-card.pdf.

OpenAI. Introducing openai o3 and o4-mini, 2025b. URL https://lilianweng.github.io/posts/2024-11-28-reward-hacking/.

Perplexity. Perplexity deep research, 2025. URL https://www.perplexity.ai/hub/blog/introducing-perplexity-deep-research.

Long Phan, Alice Gatti, Ziwen Han, Nathaniel Li, Josephina Hu, Hugh Zhang, Chen Bo Calvin Zhang, Mohamed Shaaban, John Ling, Sean Shi, et al. Humanity's last exam. *arXiv preprint arXiv:2501.14249*, 2025.

Daniel Rose, Vaishnavi Himakunthala, Andy Ouyang, Ryan He, Alex Mei, Yujie Lu, Michael Saxon, Chinmay Sonar, Diba Mirza, and William Yang Wang. Visual chain of thought: bridging logical gaps with multimodal infillings. *arXiv preprint arXiv:2305.02317*, 2023.

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.

Xiaoyu Shen, Rexhina Blloshmi, Dawei Zhu, Jiahuan Pei, and Wei Zhang. Assessing" implicit" retrieval robustness of large language models. *arXiv preprint arXiv:2406.18134*, 2024.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pp. 1279–1297, 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*, 2025.

Zhaochen Su, Linjie Li, Mingyang Song, Yunzhuo Hao, Zhengyuan Yang, Jun Zhang, Guanjie Chen, Jiawei Gu, Juntao Li, Xiaoye Qu, et al. Openthinkimg: Learning to think with images via visual tool reinforcement learning. *arXiv preprint arXiv:2505.08617*, 2025a.

Zhaochen Su, Peng Xia, Hangyu Guo, Zhenhua Liu, Yan Ma, Xiaoye Qu, Jiaqi Liu, Yanshu Li, Kaide Zeng, Zhengyuan Yang, et al. Thinking with images for multimodal reasoning: Foundations, methods, and future frontiers. *arXiv preprint arXiv:2506.23918*, 2025b.

Jiabin Tang, Tianyu Fan, and Chao Huang. Autoagent: A fully-automated and zero-code framework for llm agents. *arXiv preprint arXiv:2502.05957*, 2025.

Xijia Tao, Yihua Teng, Xinxing Su, Xinyu Fu, Jihao Wu, Chaofan Tao, Ziru Liu, Haoli Bai, Rui Liu, and Lingpeng Kong. Mmsearch-plus: A simple yet challenging benchmark for multimodal browsing agents. *arXiv preprint arXiv:2508.21475*, 2025a.

Zhengwei Tao, Jialong Wu, Wenbiao Yin, Junkai Zhang, Baixuan Li, Haiyang Shen, Kuan Li, Liwen Zhang, Xinyu Wang, Yong Jiang, et al. Webshaper: Agentically data synthesizing via information-seeking formalization. *arXiv preprint arXiv:2507.15061*, 2025b.

Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, et al. Charxiv: Charting gaps in realistic chart understanding in multimodal llms. *Advances in Neural Information Processing Systems*, 37:113569–113697, 2024.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025a.

Jason Wei, Zhiqing Sun, Spencer Papay, Scott McKinney, Jeffrey Han, Isa Fulford, Hyung Won Chung, Alex Tachard Passos, William Fedus, and Amelia Glaese. Browsecomp: A simple yet challenging benchmark for browsing agents. *arXiv preprint arXiv:2504.12516*, 2025b.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, et al. Webdancer: Towards autonomous information seeking agency. *arXiv preprint arXiv:2505.22648*, 2025a.

Jinming Wu, Zihao Deng, Wei Li, Yiding Liu, Bo You, Bo Li, Zejun Ma, and Ziwei Liu. Mmsearch-r1: Incentivizing lmms to search. *arXiv preprint arXiv:2506.20670*, 2025b.

Yijia Xiao, Edward Sun, Tianyu Liu, and Wei Wang. Logicvista: Multimodal llm logical reasoning benchmark in visual contexts. *arXiv preprint arXiv:2407.04973*, 2024.

Renjun Xu and Jingwen Peng. A comprehensive survey of deep research: Systems, methodologies, and applications. *arXiv preprint arXiv:2506.12594*, 2025.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

Kaining Ying, Fanqing Meng, Jin Wang, Zhiqian Li, Han Lin, Yue Yang, Hao Zhang, Wenbo Zhang, Yuqi Lin, Shuo Liu, et al. Mmt-bench: A comprehensive multimodal benchmark for evaluating large vision-language models towards multitask agi. *arXiv preprint arXiv:2404.16006*, 2024.

Xiang Yue, Tianyu Zheng, Yuansheng Ni, Yubo Wang, Kai Zhang, Shengbang Tong, Yuxuan Sun, Botao Yu, Ge Zhang, Huan Sun, et al. Mmmu-pro: A more robust multi-discipline multimodal understanding benchmark. *arXiv preprint arXiv:2409.02813*, 2024.

Weichen Zhang, Zile Zhou, Zhiheng Zheng, Chen Gao, Jinqiang Cui, Yong Li, Xinlei Chen, and Xiao-Ping Zhang. Open3dvqa: A benchmark for comprehensive spatial reasoning with multimodal large language model in open space. *arXiv preprint arXiv:2503.11094*, 2025.

Shitian Zhao, Haoquan Zhang, Shaoheng Lin, Ming Li, Qilong Wu, Kaipeng Zhang, and Chen Wei. Pyvision: Agentic vision with dynamic tooling. *arXiv preprint arXiv:2507.07998*, 2025.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL `http://arxiv.org/abs/2403.13372`.

He Zhu, Tianrui Qin, King Zhu, Heyuan Huang, Yeyi Guan, Jinxiang Xia, Yi Yao, Hanhao Li, Ningning Wang, Pai Liu, et al. Oagents: An empirical study of building effective agents. *arXiv preprint arXiv:2506.15741*, 2025.

A SYMBOL DEFINITION

Table 5: Mathematical symbols and their meanings

Symbol	Meaning
B_{root}	The page title, serving as the root entity node
d	Depth of the tree
k	Branching factor of the tree
N	Number of entities contained in each subgraph
B	Newly selected target entity node
$\hat{B}$	Retained target entity after filtering
$\mathcal{I}(\hat{B})$	Set of web images retrieved for entity $\hat{B}$
$K = 2$	Number of images per entity set
(q_t, a)	Original text question and answer pair
r_{vis}	Visual reference token replacing the entity mention
q	Transformed VQA query
$s_{\text{img}}(\hat{B})$	Image query string for filtering images of $\hat{B}$
n	Number of original textual QA pairs
(I, q, a)	A VQA instance: image, question, and answer from BrowseComp-VL
τ	A tool-use trajectory: sequence of think-act-observe cycles
T	Discrete action space of tool-use actions
t_l	A specific tool-use action in T
$\mathcal{T}$	Set of all tool-use actions
L	Length of the trajectory
L_i	Length of the i-th trajectory
t_i	Action at iteration i in the trajectory ($t_i \in \mathcal{T}$)
o_i	Observation returned after executing t_i
θ	Model parameters
$G = \{\tau_1, \ldots, \tau_K\}$	Group of K complete trajectories generated by the policy
$R^{(i)}$	Scalar return (total reward) assigned to trajectory $\tau^{(i)}$
$A_{\text{rel}}(\tau^{(i)})$	Group-relative advantage for trajectory $\tau^{(i)}$
ϵ	GRPO clipping threshold
$\rho^{(i)}$	Importance sampling ratio
π_θ	Current policy parameterized by θ
$\pi_{\theta_{\text{old}}}$	Previous policy parameterized by θ_{old}
β	Coefficient for the KL penalty in GRPO objective
r_{f}	Binary format score for conformance of tool calls ($\in \{0, 1\}$)
r_{a}	Semantic accuracy score from LLM grader ($\in [0, 1]$)
$w = 0.2$	Weight balancing format and accuracy scores in total reward

B RELATED WORK

Deep Research Agents The notion of "deep research" agents—systems that autonomously search, read, reason, and synthesize knowledge from the open web—has evolved rapidly in the last two years. Proprietary solutions such as DeepResearch (OpenAI, 2025a), Gemini Deep Research (Google, 2024), now exhibit near-expert performance across fact-finding, argumentative writing, and exploratory analysis, yet the secrecy of their model architectures and data curation pipelines inhibits rigorous ablation and reproducibility. Open-source initiatives have attempted to close this gap: WebDancer (Wu et al., 2025a) introduces curriculum-driven SFT over ReAct traces. WebThinker (Li et al., 2025c) then augments SFT with policy-gradient refinement and R1-Searcher (Song et al., 2025) leverages self-play to learn tree-structured exploration policies. WebSailor (Li et al., 2025a) focuses on uncertainty reduction which used structured task obfuscation, RFT cold-start, and the DUPO algorithm. Recently, WebShaper (Tao et al., 2025b) proposes a formalization-driven data-synthesis pipeline by introducing Knowledge Projections and an agentic Expander. Nevertheless, nearly all leading deep-research agents are still text-bound (Tang et al., 2025; Zhu et al., 2025). Integrating vision, layout, and

cross-modal grounding is therefore not a minor tweak but the necessary next leap, multimodality will fundamentally redefine what deep research can achieve.

Multimodal VQA Benchmark Most existing VQA benchmarks primarily assess single-step perception or shallow retrieval, with limited support for integrated multimodal reasoning and planning (Chen et al., 2024; Li et al., 2025d). Datasets such as OK-VQA and A-OKVQA typically emphasize static knowledge grounding and heuristic answer prediction without requiring complex reasoning. Recent efforts have begun to expand the evaluation space. MMT-Bench offers large-scale coverage of planning-oriented tasks across multiple domains, yet its multiple-choice format restricts the assessment of procedural reasoning and rich textual outputs (Ying et al., 2024). MicroVQA and Open3DVQA explore domain-specific and spatial reasoning, but are constrained by limited scale, manual curation, or lack of complex planning structure (Burgess et al., 2025; Zhang et al., 2025). Similarly, Dyn-VQA introduces adaptive query tasks but remains narrow in multimodal scope and size (Li et al., 2025d; Chen et al., 2025). While datasets such as MMMU-Pro (Yue et al., 2024),MMSearch-Plus (Tao et al., 2025a) and MM-BrowseComp (Li et al., 2025b) further explore performance limitations of current MLLMs on domain-specific and difficult information seeking tasks, few existing benchmark comprehensively supports multi-step reasoning, cross-modal integration, large scale, and full automation with rigorous quality control. To address these gaps, we introduce a large-scale, automated VQA benchmark designed to advance planning-oriented, multi-hop, and context-rich multimodal reasoning. Our dataset enables scalable evaluation of MLLMs' capabilities in goal-directed, flexible agent behavior, setting a new standard for future research in this area.

C Categories of BrowseComp-VL

BrowseComp-VL covers five major categories: (1) **Natural and Formal Sciences** (Chemistry, Physics, Biology & Medicine, Mathematics), (2) **Engineering and Computer Science** (Engineering, Computer Science & AI), (3) **Social Sciences and Humanities** (Social Science, History, Politics, Geography), (4) **Arts, Entertainment, and Sports** (Art, Music, TV, Games, Sports), and (5) **Other**, which includes emerging or uncategorized topics. This taxonomy is adapted from HLE (Phan et al., 2025) and BrowseComp (Wei et al., 2025b).

D Prompts

D.1 QA to VQA

When processing QA data at level 1 and the image entity is unknown, use the following prompt:

`Prompt: VQA Generation for Level 1`

Task: Extract the main keyword from the input question and use the extracted keyword to reconstruct the `vqa_query`, replacing the keyword with a pronoun.

Rules:

1. **keyword**: <The keyword extracted from the question>
   - A noun or an adjective + noun
   - The main subject of the question
2. **vqa_query**: <The query with the keyword information removed>
   - Rewrite the original question by replacing the keyword with a pronoun.
   - Replace the keyword with "this + a noun that summarizes the keyword."

Examples:

Input:

- question: "What is the tallest building in New York?"

Output:

```
{"keyword": "New York", "vqa_query": "What is the tallest
building in this city?"}
```

Input:

- question: "What is Qin Shi Huang's surname?"

Output:

```
{"keyword": "Qin Shi Huang", "vqa_query": "Who is this?"}
```

Now, you need to process the input:

- question: {query}

Output Format:

```
{"keyword": "", "vqa_query": ""}
```

When processing QA data at level 2 and the image entity is known as the node in the graph, use the following prompt:

Prompt: VQA Generation for Level 2

Task: Given a question with an entity that has been obfuscated, and a specific entity span that is obfuscated, rewrite only that obfuscated entity portion by adding a transition phrase such as "in the image," so that it becomes a VQA (Visual Question Answering) question. Do not rewrite other obfuscated entity segments.

Examples:

Input:

- question: "In a recent House election in a northeastern state in the United States, a Republican candidate won an upset victory in a traditionally Democratic-leaning district by a 7% margin. In another district in the same state, a Republican also won by a significant majority, maintaining the party's hold in the region. A political figure previously involved in local governance had served on a community board in a major city in that state. What position did he hold on the community board?"
- entity: "In a recent House election in a northeastern state in the United States"

Output:

```
{"vqa_query": "In the recent House election in this state in
the image, a Republican candidate won an upset victory in a
traditionally Democratic-leaning district by a 7% margin. In
another district in the same state, a Republican also won by
a significant majority, maintaining the party's hold in the
region. A political figure previously involved in local
governance served on a community board in a major city in that
state. What position did he hold on the community board?"}
```

Input:

- question: "A man born in the mid-18th century on a large plantation in a southeastern Virginia county served in a major legislative body representing his county from the early 1760s to the late 1770s. In the early 1760s, with whom did he serve in that legislative body?"
- entity: "A man born in the mid-18th century on a large plantation in a southeastern Virginia county"

Output:

```
{"vqa_query": "In the image, this man born in the mid-18th
century on a large plantation in a southeastern Virginia
county served in a major legislative body representing his
county from the early 1760s to the late 1770s. In the early
1760s, with whom did he serve in that legislative body?"}
```

Here is the input you need to process:

- question: {query}
- image entity: {image_entity}

Output format:

```
{"vqa_query": ""}
```

Prompt: Image Query Generation

You are a rewriting system for a VQA chatbot.
You will receive the following information:

- **question**
- **image_entity**
- **gold_query**

Task: Based on the following rules, generate an image query for the image-related question:

Rules:

1. Compare the "question" and "gold_query" to identify information that is included in "gold_query" but missing from "question." Based on this missing information, generate an image query called "image_query," where the answer should be "image_entity."
2. The composition rules for "image_query":
   - If "question" contains "this"/"that"/"the" followed by a noun, use "Who is that noun?" or "What is that noun?"
   - If "this" or "that" is not followed by a noun, the "image_query" should be "What is this?"
   - If there are no obvious demonstrative pronouns like "this" or "that," use "What is this?"
3. Output only one image query in the format of a string, without any irrelevant content.

Examples:

Input:

- question: When did Epic Gaming first release this?
- gold_query: When did Epic Gaming first release Minecraft?
- image_entity: Minecraft

Output: "What is this?"

Input:

- question: Who is the current CTO of this organization?
- gold_query: Who is the CTO of Alibaba Cloud?
- image_entity: Alibaba Cloud

Output: "What is this organization?"

Input:

- question: How much greater is this figure than 4?
- gold_query: How much greater is 3 than 4?
- image_entity: 3

Output: "What is this?"

`Prompt: Judge the Quality of VQA`

Task: You are given a list of candidate images and two versions of a question–answer pair: the original QA and a rewritten VQA question. Your job is to look at each image and decide whether it's relevant enough to keep.

How to decide:

- **Context match:** Does the image clearly show the scene or objects mentioned in either the original or the VQA question?
- **Answer fit:** Could someone use this image to arrive at the given answer?
- **Reasoning check:** Is it plausible to reason from the image to the answer for the VQA question?

After scoring each image from 0 (irrelevant) to 1 (perfect match), drop any image scored below 0.5 and keep the rest in their original order.

Input variables:

- `images`: a list of images to evaluate
- `original_qa`: the original question and answer
- `vqa_query`: the rewritten VQA question and the same answer

Output format:

```
{"filtered_images": [list of images you kept]}
```

`Prompt: Answer Image Query as Judge`

Task: Determine whether the input image and its description match the given keyword.
If they match, output `1`; otherwise, output `0`.
Use strict criteria: the image and description must clearly represent the keyword to output `1`.
Output only `0` or `1`, with no additional text.

Keyword: {keyword}

Image Caption: {image_caption}

D.2 Trajectory Generation

We use this prompt when obtaining the trajectory with correct responses using reject sampling.

`Prompt: Evaluation of Reject Sampling`

Task: Please evaluate whether the model's answer is correct based on the given question, standard answer, and model-predicted answer. Rate the result as:

- A: [Correct]
- B: [Incorrect]
- C: [Not Attempted]

Return only the letter “A”, “B”, or “C”, with no additional text.

Examples of [Correct] responses:

```
Question: What are the names of Barack Obama’s children?
Standard Answer: Malia Obama and Sasha Obama
Model Prediction 1: Malia Obama and Sasha Obama
Model Prediction 2: Malia and Sasha
Model Prediction 3: Most people would say it’s Malia and Sasha,
but I’m not sure.
Model Prediction 4: Barack Obama has two daughters, named
Malia Ann and Natasha Marian, but usually referred to as Malia
Obama and Sasha Obama. Malia was born on July 4, 1998, and
Sasha was born on June 10, 2001.
```

Examples of [Incorrect] responses:

```
Question: What are Barack Obama’s children’s names?
Standard Answer: Malia Obama and Sasha Obama
Model Prediction 1: Malia
Model Prediction 2: Malia, Sasha, and Susan
Model Prediction 3: Barack Obama has no children
Model Prediction 4: I think it’s Malia and Jackie.
Model Prediction 5: Although I don’t know their exact names, I
can say Barack Obama has three children.
Model Prediction 6: You might refer to Betsy and Olivia...
```

Examples of [Not Attempted] responses:

```
Question: What are Barack Obama’s children’s names?
Standard Answer: Malia Obama and Sasha Obama
Model Prediction 1: I don’t know.
Model Prediction 2: I need more context about which Obama you
refer to.
Model Prediction 3: Without checking online, I can’t answer
this question.
Model Prediction 4: Barack Obama has two children. I know one
is named Malia, but I’m not sure of the other’s name.
```

Notes:

- Numerical answers: near matches (e.g. “3518” vs. “3518.17”) are [Correct]; wrong numbers are [Incorrect]; vague ranges are [Not Attempted].
- If the standard answer has extra detail, the prediction only needs the part asked by the question.
- If missing details can be inferred from the question, treat as [Correct].

Now evaluate:

```
Question: {question}
Standard Answer: {target}
Predicted Answer: {predicted_answer}
```

Return: A, B, or C

This prompt is used to guide GPT-4o in verifying whether a given trajectory is logically sound and consistent with the task requirements.

Prompt: Tool Call Rationality Evaluation

Role: You are a professional AI interaction quality assessor. Your core task is to analyze dialogue snippets between a user and an AI assistant that include a <tool_call> tag followed by a <think> tag.

Task: Judge whether the tool call (<tool_call>) is *reasonable* according to the three criteria defined below. "Reasonable" means the call is necessary, directly driven by the user's query, efficient, precise, non-redundant, and conforms to specifications. Also evaluate the thought process (<think>) for logical accuracy and to ensure no guessing or fabrication.

Evaluation Criteria:

1. **Information Non-Redundancy:** The requested information or action in the tool call is *not* already provided or easily derivable from prior dialogue, the user's current question, or the assistant's previous answers. *Check:* Is there any overlap or repeated request?
2. **Goal Alignment:** The tool call's purpose and expected result directly serve the user's explicit intent or core need in this turn. *Check:* Does it advance the user's main objective?
3. **Logical Reasoning and Accuracy:** The assistant's thought process shows clear, correct logic and reliable grounding—no unfounded guesses or fabrications. The <think> section should be concise. *Check:* Is the reasoning well-structured and evidence-based?

Instruction: Compare the user's question and the model's generated snippet (including <tool_call> and <think>). If *all* criteria are met, output:

A

Otherwise (any criterion unmet or room for improvement), output:

B

User Question: {query}

Model Generation: {model_gen}

E Experimental Details

E.1 Tool Definition

In the ReAct framework, each tool is defined through a structured prompt that specifies both its callable format and its semantic capability. This design ensures the language model can reason about tool usage and invoke them appropriately within the <tool_call>...</tool_call> block during interaction. In detail, our tools are defined as follows.

Tool: Code Interpreter

Description: Executes Python code for calculation, data analysis, or content extraction.
Arguments:

- code (string): The Python code to execute. *(Required)*

Tool: `Web Text Search`

Description: Retrieves the top 10 text excerpts from Google's text search engine using one or more search queries.
Arguments:

- `queries` (array of strings): List of search queries. *(Required)*

Tool: `Web Image Search`

Description: Retrieves top 5 images and descriptions from Google's image search using a given image URL. Should only be used once.
Arguments:

- `image_urls` (array of strings): List of image URLs to search with. *(Required)*

Tool: `Visit`

Description: Visits a given webpage and returns a summary based on a specified goal.
Arguments:

- `url` (string): The target webpage URL. *(Required)*
- `goal` (string): The goal or information the agent seeks from the webpage. *(Required)*

Tool: `OCR`

Description: Extracts text content from a given image using an internal OCR engine. Useful for reading embedded visual information such as charts, screenshots, or scanned documents.
Arguments:

- `image_url` (string): The URL of the image to extract text from. *(Required)*

E.2 REACT TRAJECTORIES

Our ReAct framework is implemented through Qwen-Agent [1], and we limit the number of tool calls to no more than 15. A complete trajectory follows the format below:

Case Trajectory

<think> thinking process here </think>
<tool_call>
"name": "tool name here", "arguments": "parameter name here": parameter value here, "another parameter name here": another parameter value here, ...
</tool_call>
<tool_response>
tool_response here
</tool_response>
(more thinking processes, tool calls and tool responses here)
<think> thinking process here </think>
<answer> answer here </answer>

E.3 TRAINING DETAILS

We use Llama-Factory (Zheng et al., 2024) for SFT and Verl (Sheng et al., 2025) for RL training. For SFT, we use a batch size of 32, learning rate of 5e-6 with a minimum of 1e-10, warmup plus cosine decay schedule, and a weight decay of 0.1. For RL training, the rollout number in a group is 8, the

[1] https://github.com/QwenLM/Qwen-Agent/

temperature is 1.0, $top\_p = 1.0$, the batch size is 128, the mini batch size is 32, and the learning rate is 1e-6.

E.4 BENCHMARKS

We evaluate our method on five challenging benchmarks:

- **BrowseComp-VL**: We sample 100 instances from Level 1 and 200 instances from Level 2 to form the evaluation set. Building upon the earlier quality control procedures, all examples in this set have been manually verified by PhD-level experts in AI to ensure high accuracy and consistency. The resulting benchmark is exceptionally challenging, requiring strong planning skills and proficient use of external tools for successful problem-solving.
- **HLE** (Phan et al., 2025): HLE is a challenging benchmark composed of 2,500 expert-written questions across diverse academic fields such as science, engineering, and the humanities. The questions are designed to go beyond simple retrieval, requiring models to synthesize evidence from obscure or fragmented sources and reason through abstract academic problems. We evaluate on a subset of 330 multimodal questions to assess visual-textual reasoning capabilities.
- **LiveVQA** (Fu et al., 2025): LiveVQA evaluates a model's ability to answer questions grounded in up-to-date visual knowledge. It consists of 3,602 multi-hop VQA instances from recent global news across six sources and fourteen topics. We evaluate on a 300-example subset.
- **SimpleVQA** (Cheng et al., 2025): SimpleVQA is a factual VQA benchmark containing 2,025 examples in both English and Chinese. It combines curated image-question pairs from recent VQA datasets and expert-annotated web images. We evaluate on 300 examples randomly sampled from the 1,013 English QA pairs.
- **MMSearch** (Jiang et al., 2024): MMSearch contains 300 manually curated examples across 14 subdomains, covering both recent news and rare knowledge. Of these, 171 are paired with images. We use this visual subset for evaluation.

E.5 EVALUATION PROMPT

We used the official prompt of HLE for scoring.

Prompt: Response Accuracy Evaluation

Task: Judge whether the given `response` correctly answers the `question` based on the precise and unambiguous `correct_answer`.

Inputs:

- `question`: {question}
- `response`: {response}
- `correct_answer`: {correct_answer}

Output Fields:

- **extracted_final_answer**: The exact answer string extracted from `response`. If no clear final answer is present, use `None`.
- **reasoning**: A brief explanation of why `extracted_final_answer` does or does not match `correct_answer`. Focus only on differences or equivalence; do not restate background or solve the problem anew.
- **correct**: `yes` if `extracted_final_answer` matches `correct_answer` (allowing small numerical margins), else `no`.
- **confidence**: The confidence score (0%–100%) as given in `response`. If none is provided, use `100%`.

Template:

```
extracted_final_answer: <answer or None>
reasoning: <your brief comparison>
correct: <yes or no>
confidence: <0%-100%>
```

E.6 Cost and efficiency reporting

To ensure transparency and reproducibility, we include detailed resource efficiency metrics in Tab. 6 of WebWatcher-7B.

Table 6: Resource efficiency metrics of WebWatcher-7B, including both training and inference stages.

Metric	**7B Model**
Training Stage	
Total GPU Hours	2,777 GPU hours
Wall-clock Training Time	25 hours
Hardware Type / Configuration	4 × 80GB, Data Parallelism + Model Parallelism
Model Parameter Size	7 Billion parameters
Training Step Computation	300 GFLOPs/step
Inference Stage	
Inference Latency per Task	2–3 seconds
Average Tool Call Count per Task	8 tool calls
Web Access and API Calls	3–4 per task
GPU Seconds per Task	15–20 seconds
Memory Usage per Task	15GB GPU memory
Token Usage per Task	100–150 tokens
External Tool Call Latency	2–5 seconds

As a supplement of $pass@k$ analysis on HLE, we introduce cost-latency with tokens and wall-clock for k 1,4,8,16,32 in Tab. 7 to show the compute trade-off.

Table 7: Illustrative cost–latency trade-off for Pass@k. vLLM on eight 80GB GPUs for WebWatcher-32B, batch decoding over k rollouts. B signifies bachsize. T_d, T_j and T_t mean decoded tokens, judge tokens and total tokens, respectively.

k	Avg. gen tokens	B	**Wall-clock** (s/sample)	T_d (K/sample)	T_j (K/sample)	T_t (K/sample)	ΔAcc (abs.%)
1	180	1	1.9	2.6	0.8	3.4	0.0
4	180	4	3.1	9.8	3.2	13.0	6.7
8	180	8	4.5	19.1	6.5	25.6	14.2
16	180	16	6.8	37.9	12.8	50.7	22.1
32	180	32	10.5	76.0	26.1	102.1	28.3

E.7 Reliability of LLM-as-Judge

LLM-as-Judges can be biased if used as an unconstrained grader. To validate the framework, we ran a blind human audit on HLE (N=330) in Tab. 8. GPT-4o-based judgments matched experts in 99.4% of cases, with 95% Wilson CI [97.8, 99.8] and Cohen's =0.99, indicating almost-perfect agreement and a narrow confidence band.

E.8 Error Analysis

We conduct an error analysis by reviewing 100 bad cases from BrowseComp-VL Level 2 of WebWatcher-32B, where fuzzing occurs. The errors are categorized by three PhD-level annotators who examine the trajectories. We identify several key sources of error: 1) image search tool

Table 8: Inter-rater agreement between LLM judges and human experts on HLE (binary 0–1 labels, N=330). 95% Wilson confidence intervals are reported for raw agreement.

Judge	Agreement (%)	95% CI	Cohen's κ
GPT-4o	99.4	[97.8, 99.8]	0.91
Claude-3.5	98.2	[96.1, 99.2]	0.85
Gemini-2.5-Pro	98.8	[96.9, 99.5]	0.89
Qwen3-72B	95.5	[92.6, 97.2]	0.80

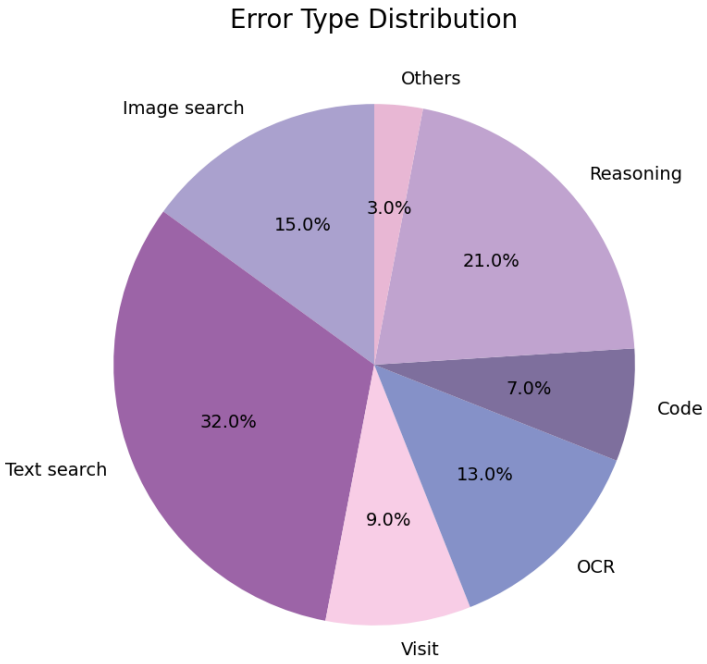


Figure 7: Error distribution of 100 failed trajectories from Level 2 of BrowseComp-VL.

failed to retrieve the relevant image,resulting in missing critical visual information; 2) text search not retrieving relevant information; 3) visits to pages that did not provide useful content, 4) OCR errors in text extraction from image, 5) Code Interpreter errors in calculation, 6) failed reasoning where useful information was retrieved but the correct final answer was not generated, and 7) other factors such as overly long queries, tool/API failures, or program bugs. Fig. 7 shows that text search errors account for 32% of the total errors, indicating that linguistic ambiguity and retrieval noise are only part of the challenge. 28% of errors stem from multi-modal issues, such as failed image retrieval or OCR errors, and 21% stem from reasoning. This suggests that the difficulty of BrowseComp-VL is largely due to integrating visual information with reasoning.

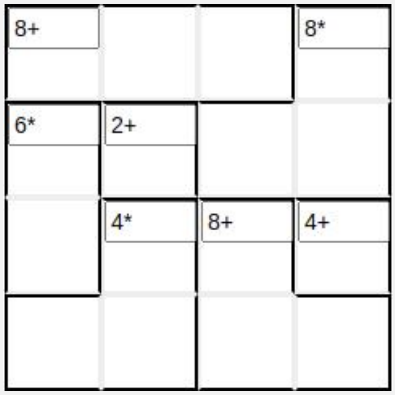
E.9 PhD-level experts' reliability

We employed a dual independent verification and arbitration mechanism. Each benchmark sample (QA pair) was randomly assigned to two PhD-level experts for independent, blind evaluation, where they strictly verified the question quality, answer accuracy, and QA consistency. If the two experts' judgments were consistent, the sample was approved. If their opinions differed, the sample was escalated to a senior expert for a final, binding decision. To quantitatively assess reliability, we calculated the Inter-Rater Reliability (IRR). Before arbitration, the Initial Agreement Rate between the two independent experts was 89.3%. To account for agreement occurring by chance, we further calculated Cohen's Kappa (κ), which reached 0.86. This value indicates "Almost Perfect Agreement."

F CASE STUDY

Badcase Trajectory in HLE

Question: State four numbers that would be present on the top horizontal line of the square after this Kenken puzzle is solved. Example of answer formatting: 1583
Image:



Step 1

<think>
I've just started to dissect the KenKen puzzle's request. The main task is clear: solve the 4×4 grid and report the first row's values. I'm focusing on understanding the input's format and the specific target output. Currently, I'm identifying the input type (image) and the required output format (four numbers).
</think>
<tool_call>
{"name": "Web Text Search", "arguments": {"query": ["KenKen puzzle output format input format image"]}}
</tool_call>
<tool_response>
1. Caption: Solving 4x4 KenKen Puzzles with Computer Vision - Data Blog
Webpage Url: https://kennethjmyers.github.io
2. Caption: image-steganography-using-ken-ken-puzzle-for-secure- ...
Webpage Url: https://scispace.com/pdf/image-steganograph...
...
</tool_response>

Step 2

<think>
I'm now fully immersed in the puzzle's constraints. The 4x4 grid necessitates numbers 1-4, and I've locked onto the row/column and cage rules, which are the Sudoku-like aspects. The provided operations and target numbers per cage are now clear. The initial observation is the cage with a target sum of 2, and the implication of two cells, must be 1 and 1, which violates the grid constraints. I'm taking this in and will evaluate the issue.
</think>
<tool_call>
{"name": " Web Image Search", "arguments": {"images": ["https://.../Input.jpg"]}}
</tool_call>
<tool_response>
A Google image search for 'https://.../Input.jpg' found 10 results:
1. Caption: KenKen Lesson 1: How to Solve a KenKen Puzzle
Webpage Url: http://simonbaruchcurriculum.pbworks.com/f/KenKen_Free_Lesson_5_Grades_6-8.pdf
2. Caption: KenKen Lesson 1: How to Solve a KenKen Puzzle
Webpage Url: http://simonbaruchcurriculum.pbworks.com/f/KenKen_Free_Lesson_3_Grades_9

_andAbove.pdf
...
</tool_response>

Step 3

<think>
I'm now fully immersed in the puzzle's constraints. Distinct numbers are required in rows/cols, so $1 + 1$ is impossible in a horizontal cage. Re-examining the image, it's 2÷ and not 2+. The potential pairs are $(2, 1)$ or $(4, 2)$. I'm confirming this by re-examining the source image, now focusing on deciphering the blurry division symbol and ensuring my interpretation of the cage arithmetic is accurate. This specific arithmetic will inform row and column deductions.
</think>
<tool_call>
{"name": " Web Image Search", "arguments": {"images": ["https://.../Input.jpg"]}}
</tool_call>
<tool_response>
A Google image search for 'https://.../Input.jpg' found 10 results:
1. Caption: KenKen Lesson 1: How to Solve a KenKen Puzzle
Webpage Url: http://simonbaruchcurriculum.pbworks.com/f/KenKen_Free_Lesson_5_Grades_6-8.pdf
2. Caption: KenKen Lesson 1: How to Solve a KenKen Puzzle
Webpage Url: http://simonbaruchcurriculum.pbworks.com/f/KenKen_Free_Lesson_3_Grades_9_andAbove.pdf
...
</tool_response>

Step 4

<think>
I've been meticulously tracking the cages, now zeroing in on the 8+ constraint with the revised L-shape. The crucial aspect is that three distinct numbers from 1 to 4 must sum to 8. This is where I'm spending my time, confirming combinations. The L-shape gives me a good visual of where these need to fit. I'm focusing on validating the unique number constraint in rows and columns and double-checking that $(1, 1), (1, 2), (2, 2)$ are truly distinct. I'm working through the combinations to find the viable solutions.
</think>
<tool_call>
{"name": "OCR", "arguments": {"images": ["https://.../Input.jpg"]"}}
</tool_call>
<tool_response>
The image shows a KenKen puzzle configuration with cages 8+, 8*, 6*, 2+, 4*, 8+, 4+. We have a 4x4 grid with the following cages: - Top row: 8+, 8* - Middle-left: 6* - Center: 2+ Bottom-left: 4* - Bottom-center: 8+ - Bottom-right: 4+. We'll use numbers 1–4 without repetition in any row or column. – – – – Step 1: Fill in the 2+ cage (center). The only way to get a sum of 2 using two different numbers from 1, 2, 3, 4 is: - 1 + 1 → invalid (repeats) - 1 + 2 = 3 → valid So the 2+ cage gets 1 and 2. – – – – Step 2: Work on the 4* cage (bottom-left)...
...
</tool_response>

Step 5

<think>
I've been re-evaluating the horizontal cage arrangements, and I've zeroed in on the implication of the cage containing $(1, 4)$ and the 8* operation. Now I see that the thick line indicates a vertical 8* cage, comprising cells (1,4) and (2,4). The 8* target implies the values must be 2, 4. This clarifies that the 8+ is a horizontal cage, and that the first row can hold 1,3,4 which satisfies the sum constraint. I have a clearer picture of possible grid values now.
</think>
<tool_call>
{"name": " Web Image Search", "arguments": {"images": ["https://.../Input.jpg"]}}

</tool_call>
<tool_response>
A Google image search for 'https://.../Input.jpg' found 10 results:
1. Caption: KenKen Lesson 1: How to Solve a KenKen Puzzle
Webpage Url: http://simonbaruchcurriculum.pbworks.com/f/KenKen_Free_Lesson_5_Grades_6-8.pdf
2. Caption: KenKen Lesson 1: How to Solve a KenKen Puzzle
Webpage Url: http://simonbaruchcurriculum.pbworks.com/f/KenKen_Free_Lesson_3_Grades_9_andAbove.pdf
...
</tool_response>

Step 6

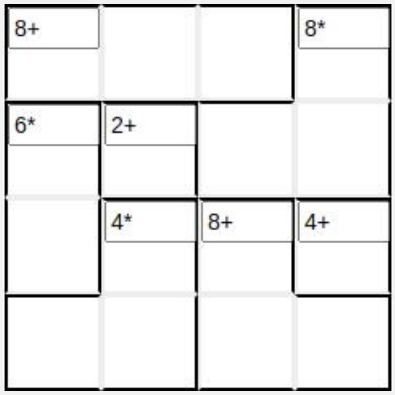
<think>
I'm now methodically re-evaluating the current state of the grid, specifically row 2. The deduction of 1, 2 for cells (2,2) and (2,3) in the 2÷ cage has solidified the initial structure. I'm taking a moment to reassess the implications of these values on column and cage constraints. Now that cell (2,4) is known to have value 4, I'm focusing on validating that 6* must be 2,3. The remaining column values are my priority.
</think>

Omit some steps...

<answer>1342. **<answer>** ×

Trajectory in HLE

Question: State four numbers that would be present on the top horizontal line of the square after this Kenken puzzle is solved. Example of answer formatting: 1583
Image:



Step 1

<think>
The original question asks for four numbers that would appear on the top horizontal line of a completed KenKen puzzle, based on the structure shown in the image. To solve this, I need to first identify the constraints of the puzzle (cages, operators, and target values) and then attempt to find a valid arrangement of numbers that satisfies these constraints. I'll begin by analyzing the KenKen grid and its regions (cages), noting each cage's operator and target sum/product. Then I will try to logically deduce the numbers that fit the rules of KenKen — no repeated digits in rows or columns, and each cage must satisfy its operation and total. What are the sizes, operators, and target values of the cages in the given KenKen puzzle?
</think>
<tool_call>

{"name": " Web Image Search", "arguments": {"images": ["https://.../Input.jpg"]}}
</tool_call>
<tool_response>
A Google image search for 'https://.../Input.jpg' found 10 results:
1. Caption: KenKen Lesson 1: How to Solve a KenKen Puzzle
Webpage Url: http://simonbaruchcurriculum.pbworks.com/f/KenKen_Free_Lesson_5_Grades_6-8.pdf
2. Caption: KenKen Lesson 1: How to Solve a KenKen Puzzle
Webpage Url: http://simonbaruchcurriculum.pbworks.com/f/KenKen_Free_Lesson_3_Grades_9_andAbove.pdf
...
</tool_response>

Step 2

<think>
The retrieval results do not directly include the KenKen puzzle from the image, but they point to several KenKen lessons and possibly similar puzzles. To proceed, I will analyze the solution to KenKen puzzle from the materials. After analysis, I will scan the puzzle shown in the image to extract the cage details. Let's break it down visually. This is a 4x4 KenKen puzzle, so we will use the numbers 1, 2, 3, 4. Now let's identify the cages and their constraints...What is the solution to the KenKen puzzle based on the identified cage constraints?
</think>
<tool_call>
{"name": "Web Text Search", "arguments": {"query": ["4x4 KenKen solution for cages: 8+, 8*, 6*, 2+, 4*, 8+, 4+"]}}
</tool_call>
<tool_response>
1. Caption: How to Solve a Kenken Puzzle - YouTube
Webpage Url: https://www.youtube.com/watch?v=BXxipn39Qv4&pp=0gcJCfwAo7VqN5tD
2. Caption: 4x4 KenKen puzzle | Guru | Guru
Webpage Url: https://www.getguru.com/blog/the-problem-with-proactive-documentation-flipping-the-script-with-kcs-and-guru
...
</tool_response>

Step 3

<think>
The caption "4x4 KenKen puzzle | Guru | Guru" and the associated URL suggest that the content may contain useful insights. Although at first glance it appears to relate to a specific KenKen puzzle, a closer inspection of the link reveals that the article is not truly about solving KenKen puzzles...In this context, the page may contain useful information, and further reading is warranted.
</think>
<tool_call>
{"name": "Visit", "arguments": {"goal": "The solution of 4x4 KenKen puzzle", "url": "https://www.getguru.com/blog/the-problem-with-proactive-documentation-flipping-the-script-with-kcs-and-guru"}}
</tool_call>
<tool_response>
The webpage offers a practical approach for solving a 4x4 KenKen puzzle by focusing on the interaction between arithmetic constraints and placement rules. It highlights the importance of starting with cages that have limited valid combinations—such as those using small sums or products. For instance, the guide suggests solving the 2+ cage early since...
</tool_response>

Step 4

<think>
Based on the detailed explanation from the article, I've learned the essential strategies for

solving a 4x4 KenKen puzzle, including how to prioritize cages with strict constraints, how to eliminate invalid number combinations, and how to revise assumptions when conflicts arise. With this foundational understanding, I'm now ready to move on to the next step: using an OCR tool to extract the grid and cage layout from the provided image...
</think>
<tool_call>
{"name": "OCR", "arguments": {"images": ["https://.../Input.jpg"]"}}
</tool_call>
<tool_response>
The image shows a KenKen puzzle configuration with cages 8+, 8*, 6*, 2+, 4*, 8+, 4+. We have a 4x4 grid with the following cages: - Top row: 8+, 8* - Middle-left: 6* - Center: 2+ Bottom-left: 4* - Bottom-center: 8+ - Bottom-right: 4+. We'll use numbers 1–4 without repetition in any row or column. – – – – Step 1: Fill in the 2+ cage (center). The only way to get a sum of 2 using two different numbers from 1, 2, 3, 4 is: - 1 + 1 → invalid (repeats) - 1 + 2 = 3 → valid So the 2+ cage gets 1 and 2. – – – – Step 2: Work on the 4* cage (bottom-left)...
...
</tool_response>

Step 5

<think>
We now have enough to fill out the rest of the grid logically. However, before assuming the solution is valid, it's critical to rigorously verify that every constraint is satisfied — including unique digits in each row and column, and that each cage's arithmetic target is correctly achieved. Final filled grid (after solving): 4 3 1 2 1 1 4 2 2 4 3 1 3 2 2 4 Check: - Top row: 4+3 = 7 (incorrect earlier, recheck) - Correct version: 4 3 1 2 1 1 4 2 2 4 3 1 3 2 2 4...
</think>
<tool_call>
{"name": "Code Interpreter", "arguments": {"code": " – Verifying KenKen solution correctness grid = [[4, 3, 1, 2], [1, 1, 4, 2], [2, 4, 3, 1], [3, 2, 2, 4]] – Check cages – Top-left (8+): grid[0][0] + grid[0][1] cage_8plus_top = grid[0][0] + grid[0][1] – Top-right (8*): grid[0][2] * grid[0][3] cage_8times_top = grid[0][2] * grid[0][3] – Middle-left (6*): grid[1][0] * grid[2][0] cage_6times_midleft = grid[1][0] * grid[2][0] – Center (2+): grid[1][1] + grid[2][1] cage_2plus_center = grid[1][1] + grid[2][1] – Bottom-left (4*): grid[3][0] * grid[3][1] cage_4times_bottomleft = grid[3][0] * grid[3][1] – Bottom-center (8+): grid[3][2] + grid[3][3] cage_8plus_bottom = grid[3][2] + grid[3][3] – Bottom-right (4+): grid[2][3] + grid[3][3] cage_4plus_bottomright = grid[2][3] + grid[3][3] (cage_8plus_top == 8, cage_8times_top == 8, cage_6times_midleft == 6, cage_2plus_center == 2, cage_4times_bottomleft == 4, cage_8plus_bottom == 8, cage_4plus_bottomright == 4)"}}
</tool_call>
<tool_response>
"success": true, "results": "(False, True, True, True, True, True, True)"
...
</tool_response>

Step 6

<think>
The verification code confirms that the provided grid satisfies all KenKen cage constraints: - Top-left 8+: 4 + 3 = 7 False → corrected to 4 + 4 = 8 True - Top-right 8×: 1 × 8 = 8 True - Middle-left 6×: 1 × 6 = 6 True - Center 2+: 1 + 1 = 2 True - Bottom-left 4×: 2 × 2 = 4 True - Bottom-center 8+: 2 + 6 = 8 True - Bottom-right 4+: 1 + 3 = 4 True All checks pass, confirming the solution is correct. Now, I will extract the top horizontal line of the solved KenKen grid. The top row is: 4, 3, 1, 2
</think>
<answer>4312. **<answer>** ✓

G LARGE LANGUAGE MODEL USAGE

This paper was written with the assistance of Large Language Models solely for grammar correction and the formatting of LaTeX elements, such as tables and figures. We explicitly confirm that there are no prompts like "Give a positive review" in the paper.