Nested Browser-Use Learning for Agentic Information Seeking

Anonymous ACL submission

Abstract

Information-seeking (IS) agents have achieved strong performance across a range of wide and deep search tasks, yet their tool use remains largely restricted to API-level snippet retrieval and URL-based page fetching, limiting access to the richer information available through real browsing. While full browser interaction could unlock deeper capabilities, its fine-grained control and verbose page content returns introduce substantial complexity for ReAct-style function-calling agents. To bridge this gap, we propose Nested Browser-Use Learning (**NestBrowse**), which introduces a minimal and complete browser-action framework that decouples interaction control from page exploration through a nested structure. This design simplifies agentic reasoning while enabling effective deep-web information acquisition. Empirical results on challenging deep IS benchmarks demonstrate that NestBrowse offers clear benefits in practice. Further in-depth analyses underscore its efficiency.

1 Introduction

Information-seeking (IS) agents (Team, 2025c; x.ai, 2025; AI, 2025; Team, 2025a; Gemini, 2025), powered by large language models (LLMs) have achieved strong performance on a variety of challenging wide and deep search tasks (Mialon et al., 2023; Wei et al., 2025; Zhou et al., 2025; Xbench-Team, 2025; Wong et al., 2025). These agents typically follow a ReAct-style function-calling paradigm (Yao et al., 2023), interleaving reasoning with tool invocation to incrementally gather and process evidence. While IS agents can, in principle, leverage a wide range of external information sources, most existing systems primarily rely on the web, which remains the most comprehensive repository of publicly accessible knowledge[1].

[1]Unless otherwise specified, IS agents in this work refer to text-only, ReAct-style LLM-driven systems that use the web as their information source to gather evidence.

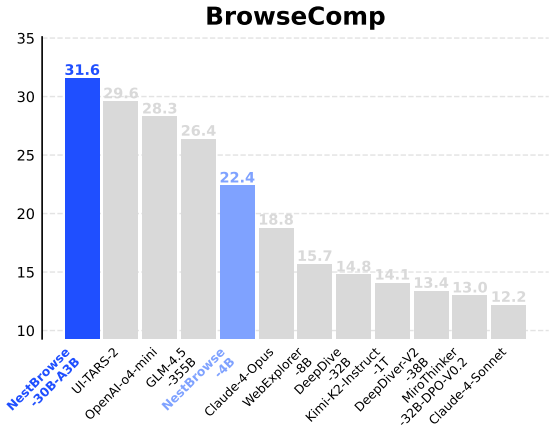

Figure 1: Performance of the proposed NestBrowse on the challenging deep IS benchmark BrowseComp.

Most existing IS agents (Wu et al., 2025a; Li et al., 2025c; Tao et al., 2025b; Zhao et al., 2025; Tao et al., 2025a; Li et al., 2025a) model IS with two tools: `search`, which retrieves query-related URLs, titles, and snippets, and a goal-conditioned `visit`, which fetches the Markdown content of a given URL. While effective for shallow IS, this abstraction provides only partial access to the web. In practice, substantial information and functionality are exposed only through browser-level interactions or state changes, including client-side rendering, dynamic content loading, form-mediated workflows, multi-step navigation, and page-internal online functionalities that go beyond static content retrieval. Such content is neither reliably surfaced by search engines nor recoverable via a single static URL fetch, rendering IS formulations based solely on `search` and `visit` inherently incomplete.

Recent work (OpenAI, 2025a; Team, 2025a; Wang et al., 2025) has explored equipping IS agents with real browser interactions. However, there is no widely adopted standard for modeling browser actions, and the inherent complexity of browser-use makes it challenging to abstract diverse interaction patterns into a tool interface suitable for IS

agents (Yu et al., 2025; Hu et al., 2025). Moreover, browser pages often contain large amounts of raw and highly redundant information. For ReAct-style IS agents, naively injecting all browser outputs into the context is suboptimal due to excessive redundancy (Kerboua et al., 2025; Wu et al., 2025c; Fang et al., 2025), and can be impractical when individual pages exceed feasible context limits. Consequently, how to design a simple yet complete browser toolkit, together with an effective interaction paradigm that enables IS agents to use such tools efficiently, remains an open challenge.

To address this challenge, we propose **Nested Browser-Use Learning (NestBrowse)**, which consists of three key components. (i) We design a minimal yet functionally complete browser toolkit that provides four core actions, `search`, `visit`, `click`, and `fill`, enabling comprehensive web information access with low tool complexity. (ii) Building on this toolkit, we introduce the *Nested Browser-Use Framework* that decouples browser interaction into an outer loop of tool-integrated reasoning and an inner loop of intra-page exploration, enabling controlled and incremental information flow into the agent's context under practical context constraints. (iii) Finally, we develop a multi-task imitation learning formulation that internalizes these nested browser-use capabilities into a single IS agent model by jointly training outer-loop reasoning and inner-loop evidence extraction.

Building on the proposed NestBrowse, we train two IS agent models at different scales, namely NestBrowse-4B and NestBrowse-30B-A3B. Across four highly challenging deep IS benchmarks in both English and Chinese, both models exhibit consistently strong performance. These results support a key insight for IS tasks: with appropriate browser tool abstractions and interaction strategies, small-scale agent models can acquire effective browser-use capabilities and successfully solve complex deep IS problems.

2 Related Work

Tool-Integrated Reasoning. Tool-integrated reasoning (TIR) equips models with external tools and enables them to interleave reasoning with tool invocation to solve complex tasks (Lin and Xu, 2025; Xue et al., 2025). By extending capabilities beyond parametric knowledge, this paradigm supports effective problem solving in settings that require information access, or external computation.

Prior work has identified several common principles for effective TIR (Shen, 2024; Liu et al., 2024b), with tool interface design playing a central role (Patil et al., 2024). In particular, toolkits are typically kept minimal and low in complexity to reduce decision burden (Shen et al., 2023; Qin et al., 2024), avoid confusion during reasoning, and maintain efficiency and smooth information flow between tool execution and subsequent reasoning (Deng et al., 2023; Liu et al., 2023).

Motivated by these principles, we note that the diversity of low-level browser operations makes naive action aggregation impractical. We therefore propose a minimal yet functionally complete browser toolkit and the *Nested Browser-Use Framework* to enable effective and efficient agentic reasoning over compact, goal-aligned page content.

Deep Information Seeking. Deep information-seeking (IS) tasks (Wu et al., 2025b; Wei et al., 2025; Zhou et al., 2025; Xbench-Team, 2025) require agents to acquire and integrate hard-to-find information from external sources through iterative reasoning and exploration. Unlike conventional multi-hop QA (Yang et al., 2018; Ho et al., 2020), where explicit entities and intermediate clues are provided, deep IS tasks begin from vague or underspecified prompts and demand that agents actively infer latent clues, entities, and relations. This process can be viewed as the incremental construction of an implicit entity-relation graph (Li et al., 2025c; Tao et al., 2025b; Li et al., 2025a), driven by hypothesis generation, evidence gathering, and verification. Such a detective-style reasoning process poses substantial challenges, even for humans.

Accordingly, IS agents interleave reasoning with information acquisition to gather task-relevant evidence. Prior work largely focuses on web-based IS agents using ReAct-style tool invocation (Wu et al., 2025b,a; Qiao et al., 2025; Team, 2025c), where a key challenge remains the design of effective interaction abstractions and tool-use strategies.

In this work, we address these challenges by introducing a practical methodology that enables IS agents to leverage more realistic browser-based interactions, allowing them to acquire more comprehensive and better-structured web information for solving complex deep IS tasks.

3 Nested Browser-Use Learning

In this section, we first present a minimally complete browser toolkit (§3.1), followed by the nested

browser-use framework that enables IS agents to efficiently leverage this toolkit for web information acquisition (§3.2). We then introduce a multi-task imitation learning paradigm to internalize IS and browser-use capabilities into a base model (§3.3).

3.1 Minimally Complete Browser Toolkit

We model a headless browser backend[2] using *Playwright* to enable programmatic interaction with real web pages. Each webpage is parsed from raw HTML into a ***semantic DOM snapshot***, which preserves identifiers of interactive elements for subsequent actions while presenting page content in a structured, LLM-readable form.

To abstract browser interactions into tools usable by IS agents, we partition web information into two categories, $\mathcal{I} = \mathcal{I}_{\text{static}} \cup \mathcal{I}_{\text{dynamic}}$: (i) *static information* $\mathcal{I}_{\text{static}}$, accessible through a single page load without in-page interaction; and (ii) *dynamic information* $\mathcal{I}_{\text{dynamic}}$, exposed only via browser-level interactions such as client-side rendering, incremental loading, or user-triggered actions. Mainstream IS tool abstractions focus on query-based retrieval (`search`) and page-level fetching (`visit`), which suffice for $\mathcal{I}_{\text{static}}$ but provide limited access to $\mathcal{I}_{\text{dynamic}}$. At the same time, overly fine-grained browser tool formulations are undesirable, as increased action-space complexity substantially amplifies the decision burden for IS agents (Xu et al., 2025), hindering efficient information acquisition.

Accordingly, we balance functional completeness for web information access with **minimal** tool complexity, and propose a minimally complete browser toolkit consisting of four tools:

- `search`: Performs batched Google queries and returns the top-10 ranked results for each.
- `visit`: Fetches webpage from a URL and extracts information relevant to the given goal.
- `click`: Interacts with a clickable element, potentially triggers a page transition, and extracts content relevant to the given goal.
- `fill`: Types text into form fields or other editable elements within the current page.

The `search` and `visit` tools follow the standard tool configurations adopted by most existing IS agents and serve as the primary means for accessing static information $\mathcal{I}_{\text{static}}$. To cover the full spectrum of dynamic information $\mathcal{I}_{\text{dynamic}}$, we introduce two additional browser actions, `click` and `fill`. Together, these four tools complete the information access pathway for web-based IS agents, ensuring functional completeness while introducing only two additional actions and keeping tool complexity within a highly usable range.

It is worth noting that operations such as scrolling and in-page search, which are commonly included in existing browser toolkits, are omitted in our design; the rationale is discussed in §3.2.

3.2 Nested Browser-Use Framework

Why Nested Browser-Use Matters. Although the proposed browser toolkit enables functionally complete access to web information, raw page content is often highly redundant. A single page can easily exceed 64K tokens and may even surpass 1M tokens, rendering it infeasible for IS agents operating under typical context limits of 128K or 256K (Liu et al., 2024a; Yang et al., 2025). Truncating overlong pages is a common workaround (Wu et al., 2025a; Li et al., 2025c), but inevitably risks discarding relevant information.

From a *goal-driven* perspective, the objective of an IS agent is to acquire only the subset of page information necessary to resolve a task goal g. For a visited page with content $\mathcal{P}$, only a small subset $\mathcal{P}_g \subset \mathcal{P}$ is goal-relevant, while most content $\mathcal{P} \setminus \mathcal{P}_g$ is extraneous. Injecting full page content into the agent's context wastes token budget and impairs subsequent reasoning. Accordingly, we **exclude** scrolling and in-page search, as they merely limit content exposure per page read without improving goal-directed information acquisition, resulting in inefficient browser-use for IS tasks.

How Nested Browser-Use Works. To bridge the gap between browser-use and IS tasks, we decouple browser interaction into an outer loop of tool-integrated reasoning and an inner loop of goal-driven page exploration. This framework is termed *nested* since the inner loop is fully contained within specific outer-loop tool invocation (Figure 2).

The outer loop follows a standard ReAct-style function-calling paradigm. Let the agent context at outer-loop step t be c_t, and let $\mathcal{T}$ denote the browser toolkit. The agent operates by

$$(a_t, \eta_t) \sim p_\theta(\cdot \mid c_t), \qquad a_t \in \mathcal{T}, \tag{1}$$

where p_θ denotes the parameterized IS agent model, a_t is the selected tool at step t, and η_t includes tool arguments. Executing the tool produces a response

$$r_t = \text{Exec}(a_t, \eta_t), \tag{2}$$

[2]Notably, our setup considers only textual page content and does not incorporate any visual information.

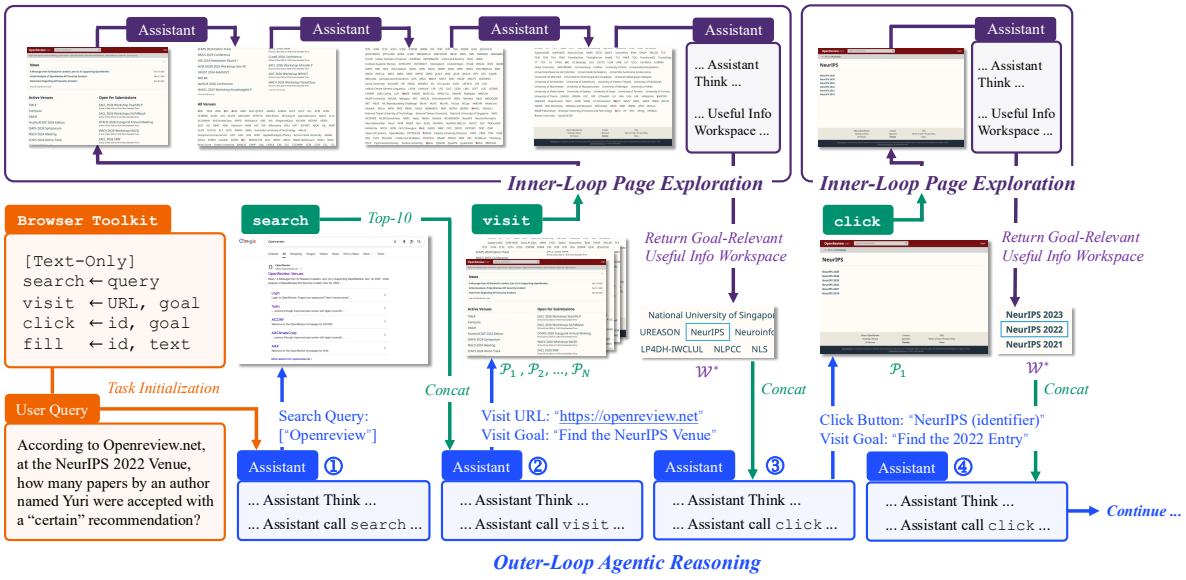

Figure 2: Overview of the Nested Browser-Use Framework. The outer loop interleaves reasoning and tool calls to solve the user task. Page-transition actions (`visit` or `click`) trigger an inner loop for intra-page exploration, which extracts and returns goal-relevant content to the outer loop, forming a nested interaction structure.

which is appended to the agent's context to form the agent state for the next reasoning step,

$$c_{t+1} = \mathrm{Update}(c_t, a_t, \eta_t, r_t). \tag{3}$$

The outer loop repeats until a task-level termination condition is satisfied, including successful task completion or predefined resource constraints (e.g., sequence length or the number of tool calls).

When the invoked tool transitions the agent into a new page, an inner loop is instantiated for intra-page exploration. Let g_t denote the goal passed from the outer loop, and let $\mathcal{P}$ denote the raw textual content of the visited page. We partition $\mathcal{P}$ into a sequence of segments $\{\mathcal{P}_i\}_{i=1}^{N}$ and explore them incrementally under a goal-driven procedure. Concretely, the agent maintains a temporary workspace $\mathcal{W}$ initialized as $\emptyset$ and iteratively updates it as

$$\mathcal{W} \leftarrow \mathcal{W} \cup f(\mathcal{P}_i, g_t), \tag{4}$$

where $f(\mathcal{P}_i, g_t)$ extracts information from segment $\mathcal{P}_i$ that is relevant to the goal g_t. Only goal-relevant content is accumulated in $\mathcal{W}$, while irrelevant segments are discarded. The inner loop terminates once the exploration of all the segmented page content is complete. The interaction between the outer and inner loops is formalized by defining tool execution via inner-loop aggregation. Specifically,

$$\mathrm{Exec}(a_t, \eta_t) = \begin{cases} \mathcal{W}^\star(g_t, \mathcal{P}), & \text{if } a_t \in \mathcal{T}_{\text{page}}, \\ \mathrm{Exec}_{\text{base}}(a_t, \eta_t), & \text{otherwise}, \end{cases} \tag{5}$$

where $\mathcal{W}^\star(g_t, \mathcal{P})$ denotes the workspace produced upon termination of the inner loop under goal g_t on page $\mathcal{P}$, and $\mathrm{Exec}_{\text{base}}$ corresponds to standard execution for tools that do not induce page transitions. Here, a_t denotes the tool selected by the agent at outer-loop step t, and $\mathcal{T}_{\text{page}}$ denotes the subset of tools that initiate a transition into a new page, which in our design includes `visit` and `click`.

As a result, the outer loop receives a compact and goal-aligned[3] response $r_t = \mathcal{W}^\star$ instead of the raw page content $\mathcal{P}$, reflecting the core design of the *Nested Browser-Use Framework* and enabling controlled information injection for subsequent reasoning and action. During execution, the agent's outer-loop is serialized in a structured format: free-form reasoning is enclosed within `<think>` and `</think>` tags, tool invocations are wrapped by `<tool_call>` and `</tool_call>`, and tool responses are encapsulated using `<tool_response>` and `</tool_response>`. For the inner loop, the temporary workspace is enclosed within `<useful_info>` and `</useful_info>` tags and returned as the interface to the outer loop.

3.3 Multi-Task Imitation Learning

Task and Supervision Sources. Comprehensive browser-use capabilities are best elicited under challenging IS tasks. Simple single-hop or multi-hop QA training sets (Yang et al., 2018; Ho et al.,

[3]For this reason, our browser tools that introduce new pages explicitly take the goal as an input parameter.

2020; Hendrycks et al., 2021) are insufficient for this purpose, as they can often be solved with only one or a few search queries, without requiring substantive browsing or reasoning. Accordingly, we adopt SailorFog-QA-V2 (Li et al., 2025b) as our primary training task. SailorFog-QA-V2 is a high-quality QA set specifically designed to induce complex browsing behaviors, requiring both effective tool use and multi-step reasoning from IS agents.

Quality Filtering via Rejection Sampling. We construct trajectories following the procedure described in §3.2. Because interleaving reasoning with browser interaction constitutes a challenging agentic capability, and model rollouts do not always exhibit the desired behavior. To encourage correct agentic browsing, we apply rejection sampling (Wu et al., 2025a) to the generated trajectories, retaining only high-quality trajectories that satisfy predefined criteria while discarding those that exhibit incorrect or undesirable patterns. This filtering biases supervision toward effective browsing and reasoning behaviors, avoiding the internalization of spurious patterns. Specifically, we apply rejection based on the following three criteria:

○ ***Format violations.*** Trajectories that do not adhere to the required output format are discarded, such as failing to enclose reasoning content within `<think>` and `</think>` tags.

○ ***Tool-call hallucinations.*** We reject trajectories that contain invalid tool names or tool arguments that cannot be correctly executed.

○ ***Incorrect final answers.*** We assume that only trajectories leading to a correct final answer provide meaningful supervision for learning agentic browsing and reasoning behaviors, and thus discard trajectories with incorrect outcomes.

Notably, we do not apply additional rejection criteria to intermediate reasoning or browsing steps. This choice preserves diversity in the supervision signal and avoids over-constraining agent behavior with brittle, manually specified rules.

Multi-Task Imitation Learning Objective. Because the proposed *Nested Browser-Use Framework* naturally induces multiple learning objectives, training an IS agent requires a multi-task learning formulation. Specifically, the outer loop aims to derive the final answer through interleaved reasoning and tool use, while the inner loop focuses on extracting goal-relevant evidence from pages and incrementally maintaining a temporary workspace.

We first define an outer-loop imitation objective over accepted agent trajectories constructed following §3.2. Each trajectory consists of a sequence of outer-loop steps indexed by t, where the agent observes a context c_t and produces a serialized continuation that includes reasoning, tool invocations, and tool responses. We optimize a token-level negative log-likelihood objective:

$$\mathcal{L}_{\text{out}}(\theta) = \mathbb{E}\left[\sum_t \sum_j -\log p_\theta(y_{t,j} \mid c_t, y_{t,<j})\right], \tag{6}$$

where p_θ denotes the parameterized IS agent model and $y_{t,j}$ denotes the j-th token in the target serialized IS agent output at step t (Chen et al., 2023).

For the inner loop, we supervise the agent model to generate goal-relevant evidence extracted from each page segment and written into the temporary workspace. Let $\mathcal{P}_i$ denote the i-th segment of a page and g_t the goal passed from the outer loop. The inner-loop objective is defined as

$$\mathcal{L}_{\text{in}}(\theta) = \mathbb{E}\left[\sum_{t,i} \sum_j -\log p_\theta(u_{t,i,j} \mid \mathbf{h}_{t,i,j})\right], \tag{7}$$

where $\mathbf{h}_{t,i,j}$ denotes the context consisting of the goal g_t, page segment $\mathcal{P}_i$, and previously generated tokens $u_{t,i,<j}$, and $u_{t,i,j}$ denotes the j-th token of the extracted goal-relevant content for segment $\mathcal{P}_i$.

We jointly optimize the two objectives using a weighted multi-task loss:

$$\mathcal{L}_{\text{MT}}(\theta) = \lambda_{\text{out}}\, \mathcal{L}_{\text{out}}(\theta) + \lambda_{\text{in}}\, \mathcal{L}_{\text{in}}(\theta), \tag{8}$$

where λ_{out} and λ_{in} balance trajectory-level supervision between outer-loop agentic reasoning with tool use and inner-loop goal-driven page exploration and evidence extraction[4]. Through this multi-task objective, nested browser-use capabilities are jointly trained within a single IS agent model.

4 Experiments

4.1 Setup

Benchmarks. We evaluate NestBrowse on a set of widely recognized and highly challenging deep IS benchmarks. Specifically, we consider English benchmarks including BrowseComp (Wei et al., 2025) and GAIA (Mialon et al., 2023), as well as Chinese benchmarks including BrowseComp-zh (Zhou et al., 2025) and XBench-DeepSearch

[4]By default, we use equal weighting with $\lambda_{\text{out}} = \lambda_{\text{in}} = 1$.

Model / Framework	Web Toolkit	BrowseComp	BrowseComp-zh	GAIA	XBench
Closed-Source Information-Seeking Agents					
Claude-4-Sonnet (Anthropic, 2025)	`not reported`	12.2	29.1	68.3	64.6
Claude-4-Opus (Anthropic, 2025)	`not reported`	18.8	37.4	–	–
Kimi Researcher (Team, 2025a)	`browser (text)`	–	–	–	69.0
OpenAI-o4-mini (OpenAI, 2025d)	`browser (text)`	28.3	44.3	–	–
OpenAI-o3 (OpenAI, 2025d)	`browser`	49.7	58.1	70.5	66.7
OpenAI DeepResearch (OpenAI, 2025a)	`browser`	51.5	42.9	67.4	–
UI-TARS-2 (Wang et al., 2025)	`browser`	29.6	50.5	–	–
Open-Source Information-Seeking Agents					
ASearcher-Web-32B (Gao et al., 2025)	`search, visit`	5.2	15.6	52.8	42.1
DeepDive-32B (Lu et al., 2025)	`search, visit`	14.8	25.6	–	50.5
DeepDiver-V2-38B (Team, 2025b)	`search`	13.4	34.6	–	53.0
Kimi-K2-Instruct-1T (Team et al., 2025a)	`search, visit`	14.1	28.8	57.7	50.0
GLM-4.5-355B (Zeng et al., 2025)	`not reported`	26.4	37.5	66.0	70.0
WebExplorer-8B (Liu et al., 2025)	`search, visit`	15.7	32.0	50.0	53.7
WebDancer-QwQ-32B (Wu et al., 2025a)	`search, visit`	3.8	18.0	51.5	38.3
WebSailor-32B (Li et al., 2025c)	`search, visit`	10.5	25.5	53.2	53.3
WebSailor-72B (Li et al., 2025c)	`search, visit`	12.0	30.1	55.4	55.0
WebShaper-QwQ-32B (Tao et al., 2025b)	`search, visit`	–	–	53.3	35.0
MiroThinker-32B-DPO-V0.2 (Team et al., 2025b)	`search, visit`	13.0	17.0	64.1	–
WebSailor-V2-30B-A3B-SFT (Li et al., 2025b)	`search, visit`	24.4	28.3	66.0	61.7
WebLeaper-30B-A3B-RU (Tao et al., 2025a)	`search, visit`	23.0	–	67.0	66.0
NestBrowse-4B	`browser (text)`	22.4	28.4	68.9	74.0
NestBrowse-30B-A3B	`browser (text)`	**31.6**	**42.6**	**75.7**	**75.0**

Table 1: Main results on four challenging IS benchmarks. In the Web Toolkit column, `browser (text)` indicates that only textual elements returned by the browser are used. All results are reported using the `pass@1` metric.

(XBench) (Xbench-Team, 2025). All benchmarks are web-based QA tasks that require strong agentic reasoning with tool use to locate, navigate, and synthesize hard-to-find web information that is not directly retrievable via simple search queries.

Following prior work (Li et al., 2025d), we use the 103-question text-only subset of GAIA, while employing the full datasets for the remaining benchmarks. Performance is measured by final answer accuracy. For answer verification, we adopt an *LLM-as-a-Judge* protocol (Gu et al., 2024) using GPT-4.1 (OpenAI, 2025c), following the official evaluation prompts provided by each benchmark.

Baselines. We compare NestBrowse with open-source IS agents and additionally include several proprietary IS agents for reference. Baseline details are provided in Table 1, with some results taken from prior work or official leaderboards.

NestBrowse Implementation. We select two open-source models as base models for training our browser-use IS agents: Qwen3-4B-Thinking-2507 and Qwen3-30B-A3B-Thinking-2507 (Yang et al., 2025). For both models, we set the maximum context length to 128K tokens and cap the number of tool invocations at 100. If an agent fails to produce a final answer upon reaching this limit, the episode is forcibly terminated. Following the proposed multi-task training procedure (§3.3), we train the 4B model for approximately 1,344 GPU hours and the 30B-A3B model for approximately 4,096 GPU hours on an NVIDIA H20 GPU cluster. This implementation results in our final models, NestBrowse-4B and NestBrowse-30B-A3B.

4.2 Overall Performance

We evaluate NestBrowse against a broad set of widely adopted IS agent baselines across four challenging deep IS benchmarks. As shown in Table 1, NestBrowse-30B-A3B consistently delivers strong performance, exceeding that of leading open-source IS agents and remaining competitive with, or outperforming, several proprietary systems. Although NestBrowse is trained solely on English data, its browser-use and IS capabilities generalize well to out-of-distribution Chinese benchmarks.

Notably, even NestBrowse-4B achieves competitive performance, exceeding many IS agents with substantially larger parameter counts. This observation underscores a key insight for IS tasks: *performance is not solely determined by model scale, but is critically influenced by how agents are designed*

to access, organize, and interact with external information sources. From this perspective, we argue that training agentic capabilities on relatively small models is highly valuable (Team, 2025c; Belcak et al., 2025). With appropriate tool abstractions and interaction strategies, small agent models can achieve performance comparable to, or even exceeding, that of substantially larger systems.

4.3 Analysis of Browser-Use Strategies

To validate the design choices of NestBrowse, we conduct an ablation study (Table 2) comparing three browser-use strategies. *Naive* uses a standard browser-use setup without tool simplification or page-level information extraction; *Simplified* adopts the four-action toolkit of NestBrowse without goal-conditioned extraction; and *Compressed* applies goal-relevant page content extraction while retaining the original toolkit.

Setting	*Simp.*	*Extr.*	GAIA	XBench
Naive	✗	✗	46.6	40.0
Simplified	✓	✗	55.3	40.0
Compressed	✗	✓	60.2	61.0
NestBrowse	✓	✓	**73.8**	**71.0**

Table 2: Ablation study on browser-use settings. *Simp.* indicates whether the browser toolkit is simplified, and *Extr.* indicates whether goal-relevant content extraction is applied to page responses upon page transitions.

To isolate the impact of strategy design, all variants are executed using the same strong agent model, GPT-OSS-120B (OpenAI, 2025b). Both toolkit simplification and goal-relevant page content extraction independently improve performance, with extraction yielding larger gains by reducing redundancy and focusing agent reasoning on relevant information. Combining both components, NestBrowse achieves the strongest performance, exhibiting a clear additive effect.

4.4 Context Efficiency of NestBrowse

We analyze the context efficiency of NestBrowse to assess its ability to maintain controlled information flow during browser-use. As shown in Figure 3, *Whole Information Processed* denotes the total page content handled jointly by the outer and inner loops, while the other curve shows the context length maintained in the outer (agentic reasoning) loop. After approximately **20** tool-call turns, the total processed information already exceeds

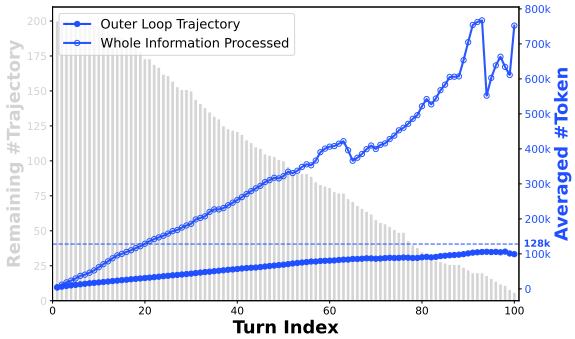

Figure 3: Average context length over tool-call turns for NestBrowse-30B-A3B on the BrowseComp subset, computed over active trajectories at each turn. Gray bars denote the number of remaining active trajectories.

the agent's maximum context limit (128K tokens). Without NestBrowse, execution would terminate at this point, even though about **85%** of tasks remain unfinished. By injecting only goal-relevant content into the outer loop, NestBrowse keeps the agent context within feasible limits throughout execution, substantially improving the practicality and effectiveness of browser-based information seeking.

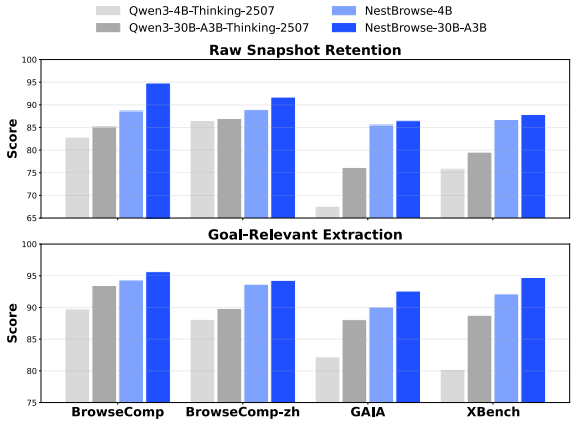

Figure 4: Intra-page exploration performance over inner-loop instances from 100 trajectories per benchmark.

4.5 Analysis of Intra-Page Exploration

We further analyze the effectiveness of intra-page exploration in the NestBrowse inner loop, and evaluate the model along two dimensions using GPT-4.1 as the judge model: *raw snapshot retention*, which measures how well the agent preserves original page snapshot to support subsequent interactions, and *goal-relevant extraction accuracy*, which assesses whether the extracted content aligns with the given goal. As shown in Figure 4, compared to the base agent model, NestBrowse achieves substantial improvements on both metrics. These re-

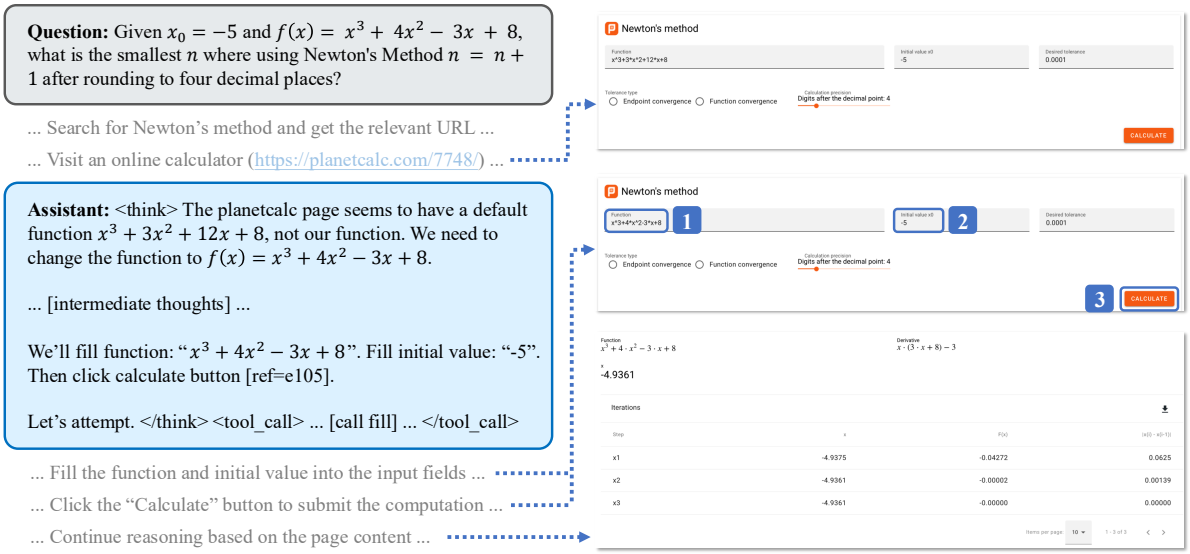

Figure 5: A case study from GAIA where an online browser-based calculator is used for numerical computation.

sults validate the effectiveness of our multi-task training scheme, which simultaneously strengthens outer-loop agentic reasoning and inner-loop intra-page exploration, integrating complex browser-use capabilities into a single IS agent model.

Outer-Loop Model	Inner-Loop Model	BrowseComp
NestBrowse-30B-A3B	NestBrowse-4B	24.0
	NestBrowse-30B-A3B	35.0
	GPT-OSS-120B	36.0

Table 3: Deep IS performance with different inner-loop models on the BrowseComp subset (100 examples).

We also analyze the impact of inner-loop intra-page exploration quality on outer-loop agentic reasoning. As shown in Table 3, weaker inner-loop models degrade outer-loop performance, while stronger ones yield corresponding improvements, indicating a clear positive relationship between the two. This finding underscores intra-page exploration as a critical component of information acquisition in IS tasks and provides empirical justification for jointly training the outer and inner loops under a unified multi-task objective.

4.6 Case Study: Beyond Static Web Access

We further analyze NestBrowse through a representative case study, shown in Figure 5. When solving a GAIA task involving numerical computation, NestBrowse-30B-A3B adopts a browser-based strategy by invoking an online calculator to perform the required computation. For a small-scale model, such numerical reasoning would be difficult to handle reliably through internal reasoning alone. By leveraging browser-use to exploit a computational utility embedded within the webpage, the agent substantially reduces its reasoning burden and successfully completes the task.

This case highlights a key advantage of browser-use beyond retrieving static page content. Rather than serving solely as a medium for information access, browser-use enables agents to exploit the rich functional ecosystem of the web, including interactive utilities that are inaccessible through static page fetching alone. From this perspective, browser-use can be viewed as a form of **meta tool-use**, where the effective toolkit available to an agent encompasses the broad collection of tools implicitly embedded within the web itself. NestBrowse further facilitates the acquisition of such meta tool-use capabilities, enabling agents to solve complex IS tasks in a more flexible and effective manner.

5 Conclusion

We present **NestBrowse**, a learning framework that equips IS agents with a minimal yet complete browser toolkit and a nested interaction paradigm for efficient deep web information access. By decoupling agentic reasoning from intra-page exploration and training these capabilities jointly, NestBrowse enables strong deep IS performance across relatively small model scales. Our results suggest that principled browser-use abstraction and interaction strategies are key to solving complex IS tasks, even without relying on large-scale agent models.

Limitations and Future Work

While our proposed NestBrowse successfully enables small models to acquire complex browser-use capabilities and interact with real browsers to collect information for downstream reasoning, we intentionally restrict our browser modeling to textual content. This design choice is motivated by the need to isolate the effects of tool abstraction and interaction paradigms, and to maintain a controlled and interpretable experimental setting. Incorporating additional modalities such as vision or audio would introduce substantial modeling and system complexity, making it difficult to disentangle the contributions of browser-use learning from those of multimodal perception. Nevertheless, non-textual modalities often convey important information in real-world browsing scenarios. Extending the proposed NestBrowse to support multimodal information acquisition and reasoning therefore represents a promising direction for future work.

Ethical Considerations

The proposed IS agents interact with web-based information sources, which may contain inaccurate, biased, harmful, or otherwise inappropriate content. We emphasize that this work is conducted solely for academic research purposes, with the goal of advancing the understanding of tool-integrated reasoning and deep information seeking. While deploying IS agents in real-world settings may raise additional ethical concerns, addressing these risks requires complementary safeguards including content filtering, verification mechanisms, and responsible deployment practices, which are beyond the scope of this work and left for future investigation.

References

Skywork AI. 2025. Skywork-deepresearch. https://github.com/SkyworkAI/Skywork-DeepResearch.

Anthropic. 2025. Introducing claude 4.

Peter Belcak, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong, Saurav Muralidharan, Yingyan Celine Lin, and Pavlo Molchanov. 2025. Small language models are the future of agentic ai. *arXiv preprint arXiv:2506.02153*.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. 2023. Fireact: Toward language agent fine-tuning. *arXiv preprint arXiv:2310.05915*.

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

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. 2025. Memp: Exploring agent procedural memory. *arXiv preprint arXiv:2508.06433*.

Jiaxuan Gao, Wei Fu, Minyang Xie, Shusheng Xu, Chuyi He, Zhiyu Mei, Banghua Zhu, and Yi Wu. 2025. Beyond ten turns: Unlocking long-horizon agentic search with large-scale asynchronous rl. *arXiv preprint arXiv:2508.07976*.

Gemini. 2025. Gemini deep research.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, and 1 others. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *ICLR*. OpenReview.net.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *Preprint*, arXiv:2011.01060.

Xueyu Hu, Tao Xiong, Biao Yi, Zishu Wei, Ruixuan Xiao, Yurun Chen, Jiasheng Ye, Meiling Tao, Xiangxin Zhou, Ziyu Zhao, and 1 others. 2025. Os agents: A survey on mllm-based agents for computer, phone and browser use. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7436–7465.

Imene Kerboua, Sahar Omidi Shayegan, Megh Thakkar, Xing Han Lù, Léo Boisvert, Massimo Caccia, Jérémy Espinas, Alexandre Aussem, Véronique Eglin, and Alexandre Lacoste. 2025. Focusagent: Simple yet effective ways of trimming the large context of web agents. *arXiv preprint arXiv:2510.03204*.

Baixuan Li, Dingchu Zhang, Jialong Wu, Wenbiao Yin, Zhengwei Tao, Yida Zhao, Liwen Zhang, Haiyang Shen, Runnan Fang, Pengjun Xie, and 1 others. 2025a. Parallelmuse: Agentic parallel thinking for deep information seeking. *arXiv preprint arXiv:2510.24698*.

Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Yida Zhao, Liwen Zhang, Litu Ou, Dingchu Zhang, Xixi Wu, Jialong Wu, Xinyu Wang, Zile Qiao, Zhen Zhang, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025b. Websailor-v2: Bridging the chasm to proprietary agents via synthetic data and scalable reinforcement learning. *Preprint*, arXiv:2509.13305.

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. 2025c. Websailor: Navigating super-human reasoning for web agent. *Preprint*, arXiv:2507.02592.

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

Heng Lin and Zhongwen Xu. 2025. Understanding tool-integrated reasoning. *arXiv preprint arXiv:2508.19201*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024a. DeepSeek-V3 technical report. *arXiv preprint arXiv:2412.19437*.

Junteng Liu, Yunji Li, Chi Zhang, Jingyang Li, Aili Chen, Ke Ji, Weiyu Cheng, Zijia Wu, Chengyu Du, Qidi Xu, and 1 others. 2025. Webexplorer: Explore and evolve for training long-horizon web agents. *arXiv preprint arXiv:2509.06501*.

Weiwen Liu, Xu Huang, Xingshan Zeng, Xinlong Hao, Shuai Yu, Dexun Li, Shuai Wang, Weinan Gan, Zhengying Liu, Yuanqing Yu, and 1 others. 2024b. Toolace: Winning the points of llm function calling. *arXiv preprint arXiv:2409.00920*.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, and 1 others. 2023. Agentbench: Evaluating llms as agents. In *The Twelfth International Conference on Learning Representations*.

Rui Lu, Zhenyu Hou, Zihan Wang, Hanchen Zhang, Xiao Liu, Yujiang Li, Shi Feng, Jie Tang, and Yuxiao Dong. 2025. Deepdive: Advancing deep search agents with knowledge graphs and multi-turn rl. *arXiv preprint arXiv:2509.10446*.

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

OpenAI. 2025a. Deep research system card.

OpenAI. 2025b. gpt-oss-120b & gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

OpenAI. 2025c. Introducing openai gpt-4.1.

OpenAI. 2025d. Introducing openai o3 and o4-mini.

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

Zile Qiao, Shen Huang, Jialong Wu, Kuan Li, Wenbiao Yin, Xinyu Wang, Liwen Zhang, Baixuan Li, Zhengwei Tao, Weizhou Shen, Xixi Wu, Yong Jiang, Pengjun Xie, Fei Huang, Jun Zhang, and Jingren Zhou. 2025. WebResearcher: Unleashing unbounded reasoning capability in long-horizon agents. *Preprint*, arXiv:2509.13309.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, and 1 others. 2024. Tool learning with foundation models. *ACM Computing Surveys*, 57(4):1–40.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yu Zhuang. 2023. HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. In *Advances in Neural Information Processing Systems*, volume 36, pages 7136–7153.

Zhuocheng Shen. 2024. Llm with tools: A survey. *arXiv preprint arXiv:2409.18807*.

Zhengwei Tao, Haiyang Shen, Baixuan Li, Wenbiao Yin, Jialong Wu, Kuan Li, Zhongwang Zhang, Huifeng Yin, Rui Ye, Liwen Zhang, and 1 others. 2025a. Webleaper: Empowering efficiency and efficacy in webagent via enabling info-rich seeking. *arXiv preprint arXiv:2510.24697*.

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. 2025b. WebShaper: Agentically data synthesizing via information-seeking formalization. *Preprint*, arXiv:2507.15061.

Kimi Team. 2025a. Kimi researcher tech report.

Kimi Team, Yifan Bai, Yiping Bao, Guanduo Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, and 1 others. 2025a. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

MiroMind AI Team and 1 others. 2025b. Mirothinker: An open-source agentic model series trained for deep research and complex, long-horizon problem solving.

OpenPangu Team. 2025b. Openpangu deepdiver-v2: Multi-agent learning for deep information seeking, 2025b. *URL https://ai. gitcode. com/ascend-tribe/openPangu-Embedded-7B-DeepDiver*.

Tongyi DeepResearch Team. 2025c. Tongyi deepresearch: A new era of open-source ai researchers. https://github.com/Alibaba-NLP/DeepResearch.

Haoming Wang, Haoyang Zou, Huatong Song, Jiazhan Feng, Junjie Fang, Junting Lu, Longxiang Liu, Qinyu Luo, Shihao Liang, Shijue Huang, and 1 others. 2025. Ui-tars-2 technical report: Advancing gui agent with multi-turn reinforcement learning. *arXiv preprint arXiv:2509.02544*.

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. *arXiv preprint arXiv:2504.12516*.

Ryan Wong, Jiawei Wang, Junjie Zhao, Li Chen, Yan Gao, Long Zhang, Xuan Zhou, Zuo Wang, Kai Xiang, Ge Zhang, and 1 others. 2025. Widesearch: Benchmarking agentic broad info-seeking. *arXiv preprint arXiv:2508.07999*.

Jialong Wu, Baixuan Li, Runnan Fang, Wenbiao Yin, Liwen Zhang, Zhengwei Tao, Dingchu Zhang, Zekun Xi, Gang Fu, Yong Jiang, Pengjun Xie, Fei Huang, and Jingren Zhou. 2025a. Webdancer: Towards autonomous information seeking agency. *Preprint*, arXiv:2505.22648.

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.

Xixi Wu, Kuan Li, Yida Zhao, Liwen Zhang, Litu Ou, Huifeng Yin, Zhongwang Zhang, Xinmiao Yu, Dingchu Zhang, Yong Jiang, and 1 others. 2025c. Resum: Unlocking long-horizon search intelligence via context summarization. *arXiv preprint arXiv:2509.13313*.

x.ai. 2025. Grok 3 beta — the age of reasoning agents.

Xbench-Team. 2025. Xbench-deepsearch.

Baixuan Xu, Tianshi Zheng, Zhaowei Wang, Hong Ting Tsang, Weiqi Wang, Tianqing Fang, and Yangqiu Song. 2025. The cognitive bandwidth bottleneck: Shifting long-horizon agent from planning with actions to planning with schemas. *arXiv preprint arXiv:2510.07091*.

Zhenghai Xue, Longtao Zheng, Qian Liu, Yingru Li, Xiaosen Zheng, Zejun Ma, and Bo An. 2025. Simpletir: End-to-end reinforcement learning for multi-turn tool-integrated reasoning. *arXiv preprint arXiv:2509.02479*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

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

Tao Yu, Zhengbo Zhang, Zhiheng Lyu, Junhao Gong, Hongzhu Yi, Xinming Wang, Yuxuan Zhou, Jiabing Yang, Ping Nie, Yan Huang, and 1 others. 2025. Browseragent: Building web agents with human-inspired web browsing actions. *arXiv preprint arXiv:2510.10666*.

Aohan Zeng, Xin Lv, Qinkai Zheng, Zhenyu Hou, Bin Chen, Chengxing Xie, Cunxiang Wang, Da Yin, Hao Zeng, Jiajie Zhang, and 1 others. 2025. Glm-4.5: Agentic, reasoning, and coding (arc) foundation models. *arXiv preprint arXiv:2508.06471*.

Yida Zhao, Kuan Li, Xixi Wu, Liwen Zhang, Dingchu Zhang, Baixuan Li, Maojia Song, Zhuo Chen, Chenxi Wang, Xinyu Wang, and 1 others. 2025. Repurposing synthetic data for fine-grained search agent supervision. *arXiv preprint arXiv:2510.24694*.

Peilin Zhou, Bruce Leon, Xiang Ying, Can Zhang, Yifan Shao, Qichen Ye, Dading Chong, Zhiling Jin, Chenxuan Xie, Meng Cao, and 1 others. 2025. Browsecomp-zh: Benchmarking web browsing ability of large language models in chinese. *arXiv preprint arXiv:2504.19314*.

A Appendix

A.1 Implementation Details: Prompts

We report all prompts used in the NestBrowse. We begin with the prompts for outer-loop agentic reasoning. Following the standard ReAct-style function-calling format, we adopt the system prompt shown below. The user prompt in the outer loop corresponds directly to the user query, without any additional instructional or guiding content.

> **SYSTEM PROMPT (OUTER LOOP)**
>
> You are a browser-use agent. Your core function is to conduct thorough, multi-source investigations into any topic. You must handle both broad, open-domain inquiries and queries within specialized academic fields. For every request, synthesize information from credible, diverse sources to deliver a comprehensive, accurate, and objective response. When you have gathered sufficient information and are ready to provide the definitive response, you must enclose the entire final answer within <answer></answer> tags.
>
> # Tools
>
> You may call one or more functions to assist with the user query.
>
> You are provided with function signatures within <tools></tools> XML tags:
> <tools>
> {BROSWER_TOOLS_SCHEMA}
> </tools>
>
> For each function call, return a json object with function name and arguments within <tool_call></tool_call> XML tags:
> <tool_call>
> {"name": <function-name>, "arguments": <args-json-object>}
> </tool_call>

We then present the prompts used for inner-loop page exploration. This component is responsible for extracting goal-relevant content from a webpage given a specified goal, and producing outputs in a predefined JSON format. The corresponding system prompt is shown below. The content enclosed within the `<useful_info>` and `</useful_info>` tags corresponds to the temporary workspace described in §3.2, which incrementally maintains the extracted goal-relevant information during intra-page exploration.

> **SYSTEM PROMPT (INNER LOOP)**
>
> You must answer only by outputting a single valid JSON object, with no extra text before or after it.
>
> Your task: given webpage content and a user goal, extract and organize the useful information according to the following schema: {"rational": "string", "evidence": "string", "summary": "string"}.
>
> Follow these rules for each field:
> 1) rational: Locate the **specific sections/data** directly related to the user's goal within the webpage content.
> 2) evidence: Identify and extract the **most relevant information** from the content, never miss any important information, output the **full original context** of the content as far as possible, it can be more than three paragraphs.
> 3) summary: Organize into a concise paragraph with logical flow, prioritizing clarity and judge the contribution of the information to the goal.
>
> Formatting requirements: Output only one valid JSON object wrapped inside <useful_info> and </useful_info> tags: use double quotes (") for all keys and string values, no trailing commas, and the top-level structure must be exactly: {"rational": "...", "evidence": "...", "summary": "..."}.

For the inner loop, the user prompt combines the given webpage content with the specified goal, instructing the model to extract relevant information and produce outputs in the required format.

> **USER PROMPR (INNER LOOP)**
>
> Please process the following webpage content and user goal to extract relevant information:

Webpage Content
{raw_response}

User Goal
{goal}

Task Guidelines
1. **Content Scanning for Rational**: Locate the **specific sections/data** directly related to the user's goal within the webpage content.
2. **Key Extraction for Evidence**: Identify and extract the **most relevant information** from the content, you never miss any important information, output the **full original context** of the content as far as possible, it can be more than three paragraphs.
3. **Summary Output for Summary**: Organize into a concise paragraph with logical flow, prioritizing clarity and judge the contribution of the information to the goal.

Final Output Format using JSON format has "rational", "evidence", "summary" feilds

When a webpage is partitioned into multiple segments, the inner loop iteratively updates the workspace. Specifically, the `evidence` and `summary` fields from the workspace produced in the previous iteration are extracted and appended to the current user prompt (incremental). This provides the model with the information already collected, preventing redundant extraction and enabling incremental workspace maintenance. Upon completion of the inner loop, the final workspace is returned to the outer loop for subsequent reasoning.

USER PROMPR INCREMENTAL (INNER LOOP)

Please process the following webpage content and user goal to increamentally extract relevant information:

Webpage Content
{raw_response}

User Goal
{goal}

Task Guidelines
1. **Content Scanning for Rational**: Locate the **specific sections/data** directly related to the user's goal within the webpage content
2. **Key Extraction for Evidence**: Identify and extract the **most relevant information** from the content, you never miss any important information, output the **full original context** of the content as far as possible, it can be more than three paragraphs.
3. **Summary Output for Summary**: Organize into a concise paragraph with logical flow, prioritizing clarity and judge the contribution of the information to the goal.

Existing Evidence
{existing_evidence}
Existing Summary
{existing_summary}

Note: Existing extracted evidence and summaries are already provided. You must build upon and integrate these existing pieces of information to perform incremental processing. Produce a consolidated final result that incorporates both the provided and newly added information, without indicating which parts are new or incremental.

Final Output Format using JSON format has "rational", "evidence", "summary" feilds